Developing and Evaluating Algorithms for Fixing Omission and Commission Errors in Structured Data

by

Mona Nashaat Ali Elmowafy

A thesis submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in

Software Engineering and Intelligent Systems

Department of Electrical and Computer Engineering
University of Alberta

Abstract

The use of machine learning is rapidly rising to deliver a variety of benefits in various domains. However, developing predictive systems often faces many challenges that can drastically delay model deployment. For instance, obtaining labeled training data is one of the most expensive bottlenecks in data preprocessing tasks in machine learning. Therefore, organizations, in many domains, are applying weak supervision to produce noisy labels. However, since weak supervision relies on cheaper sources, the quality of the generated labels is often problematic. Although recent research tries to enable machine learning to work with different types of weak supervision such as noisy and incomplete data, the previous literature treats each type individually without considering the possibility of compound weakly supervised learning.

Similarly, handling data quality issues in big data has turned into a challenging task. The key characteristics of big data have amplified the harmful impact of data errors. For example, the tremendous rate of data collection, along with the variable nature of big data, has complicated the process of error detection since data has become susceptible to various types of errors. Existing error detection techniques are typically tailored to detect certain types of errors. Moreover, most of these detection models either require user-defined rules or ample hand-labeled training examples.

Therefore, motivated by these challenges, this research proposes a set of systems to handle the problems of data preparation in real-world situations. First, to design these systems, an extensive

experimental study has been conducted to evaluate the effectiveness of existing solutions to real-world data. As for the data labeling challenges, we propose a novel technique in which we combine weak supervision and active learning to solve the labeling problem in large industrial datasets. The proposed system optimizes the labeling process to minimize the annotation cost while incorporating domain expertise in the process.

Second, to tackle the problem of learning in the presence of weak data, we present a classification algorithm that can handle inaccurate and incomplete supervised datasets. The model exploits the unlabeled data in semi-supervised settings to detect noisy data points. Then, it applies a rectification process to improve the performance of the final classifier.

Finally, targeted at providing a holistic error detection system for tabula data, we present a self-learning bidirectional encoder representation for tabular data. The system follows the encoder architecture with multi self-attention layers to model the dependencies between data cells and capture tuple-level representations. Once these representations are inferred from the data, the model parameters are fine-tuned with the task of erroneous data detection.

To evaluate the systems mentioned above, we apply an extensive set of experiments against state-of-the-art techniques. During the experiments, we report different evaluation metrics, including classification performance, human effort, and data quality measures. The empirical results are highly promising and depict that the proposed frameworks can help improve data quality and automate most data preparation processes.

Preface

This thesis is an original work by Mona Nashaat Ali Elmowafy. Prof. James Miller was the supervisory author and was involved in developing the ideas for the research and manuscript composition. The second chapter of this thesis is published as research articles as:

- M. Nashaat, A. Ghosh, J. Miller, S. Quader, C. Marston, and J. Puget, "Hybridization of Active Learning and Data Programming for Labeling Large Industrial Datasets," *2018 IEEE International Conference on Big Data (Big Data)*, Seattle, WA, USA, 2018, pp. 46-55, doi: 10.1109/BigData.2018.8622459.
- M. Nashaat, A. Ghosh, J. Miller, and S. Quader, "WeSAL: Applying Active Supervision to Find High-quality Labels at Industrial Scale," *Proceedings of the 53rd Hawaii International Conference on System Sciences*, Maui, Hawaii, USA, 2020, pp. 219-228, doi: 10.24251/HICSS.2020.028.

and published as a patent in:

- M. Nashaat, A. Basak, S. Quader, J. Miller, "Hybridization of Active Learning and Data Programming for Labelling Large Industrial Datasets," 'PUBLISHED', IBM Corporation, 2018.

The third chapter of this thesis is published as a research article in:

- M. Nashaat, A. Ghosh, J. Miller, S. Quader, and C. Marston, "M-Lean: An end-to-end development framework for predictive models in B2B scenarios, " *Information and Software Technology*, vol. 113, pp. 131–145, Sep. 2019, doi: 10.1016/j.infsof.2019.05.009.

The fourth chapter of this thesis is accepted for publication as a research article in:

- M. Nashaat, A. Ghosh, J. Miller, and S. Quader, "Asterisk: Generating Large Training Datasets with Automatic Active Supervision," *ACM Transactions on Data Science (TDS),* vol. 1, no. 2, May 2020, doi: 10.1145/3385188.

and filed as a U.S. patent in:

- M. Nashaat, S. Quader, J-F. Puget, "Labeling Data using Automated Weak Supervision," United States Patent P201910742US01, Invention Reference P201910742, 2020.

The fifth chapter of this thesis is submitted for publication as a research article in:

- M. Nashaat, A. Ghosh, J. Miller, and S. Quader, "Semi-Supervised Ensemble Learning for Dealing with Inaccurate and Incomplete Supervision," *ACM Transactions on Knowledge Discovery from Data (TKDD),* 2020.

and filed as a U.S. patent in:

- M. Nashaat, S. Quader, D. Reimer, "Semi-Supervised Ensemble Learning for Dealing with Inaccurate and Incomplete Supervision," Invention Reference 96046620, 2020.

The sixth chapter of this thesis is submitted for publication as a research article in:

- M. Nashaat, A. Ghosh, J. Miller, and S. Quader, "TabReformer: Unsupervised Representation Learning for Erroneous Data Detection," *ACM Transactions on Data Science,* 2020.

and filed as a U.S. patent in:

- M. Nashaat, S. Quader, P. Mierzejewski, "TabReformer: Bidirectional Representation Model for Erroneous Data Detection," Invention Reference P202005526, 2020.

To mom, dad, and my sisters who always enrich my life with love and beauty

Acknowledgments

I would like to express my sincere gratitude and appreciation to Prof. James Miller for his continuous support. This thesis would not have been possible without his motivation, immense knowledge, and excellent supervision. Also, I would like to thank the Egyptian government and the Egyptian Cultural and Educational Bureau in Canada for their great support. In addition, I wish to express my deepest gratitude to the members of my examination committee: Dr. Herna Viktor, Dr. Scott Dick, Dr. Peter Musilek, Dr. Marek Reformat, and Dr. Cor-Paul Bezemer for spending their valuable time to review this thesis. I want to thank them for sharing their thoughtful comments and suggestions to improve this thesis from many perspectives. Also, I would like to thank my friend and colleague, Aindrila Ghosh. I am grateful for the chance to work with her. She has given me counsel and encouragement for many years, and this journey would not have been as fulfilling and joyful without her company.

Contents

List of Tables

List of Figures

Chapter 1 : Introduction

Machine learning models are being used heavily in many domains to obtain further value from data. As a multidisciplinary field, machine learning employs statistics and computer algorithms to build data-driven models. These models are trained to provide predictions and adjust their output according to the processed data. Traditionally, developing machine learning models includes a set of data processing activities such as data collection, feature engineering, and model development and evaluation. A typical machine learning workflow is illustrated in Figure 1.1. Although the figure shows the fundamental processes for developing machine learning models, different design factors may affect these steps and carry out various adjustments. For example, choosing supervised learning algorithms [1] adds the burden of collecting high-quality labeled data to train the model. Also, as the figure shows, the results from the model evaluation are fed back to the pipeline and analyzed. Depending on these results, the model developer may choose to go back and repeat some of the earlier processes, before proceeding to model deployment.

1.1. Research Motivation

However, many challenges arise in developing learning models for real-world applications. A recent survey from Alegion [2] states that more than 95% of machine learning projects fail or are delayed because of data preparation issues. Data preparation processes such as data labeling and ensuring data quality are considered to be the single biggest obstacle to deploying business intelligence systems.

As for one challenge, real-world data usually comes in an unlabeled form. A label, in machine learning, refers to the answer that a model aims to predict. To provide these predictions, supervised learning models utilize a set of training examples to learn a function that maps between a set of input features and the corresponding labels [1]. Once this function is inferred from the data, the model can apply it to unlabeled examples to produce answers (predictions). Hence, supervised learning requires access to labeled training data; recent data-greedy learning models, such as neural

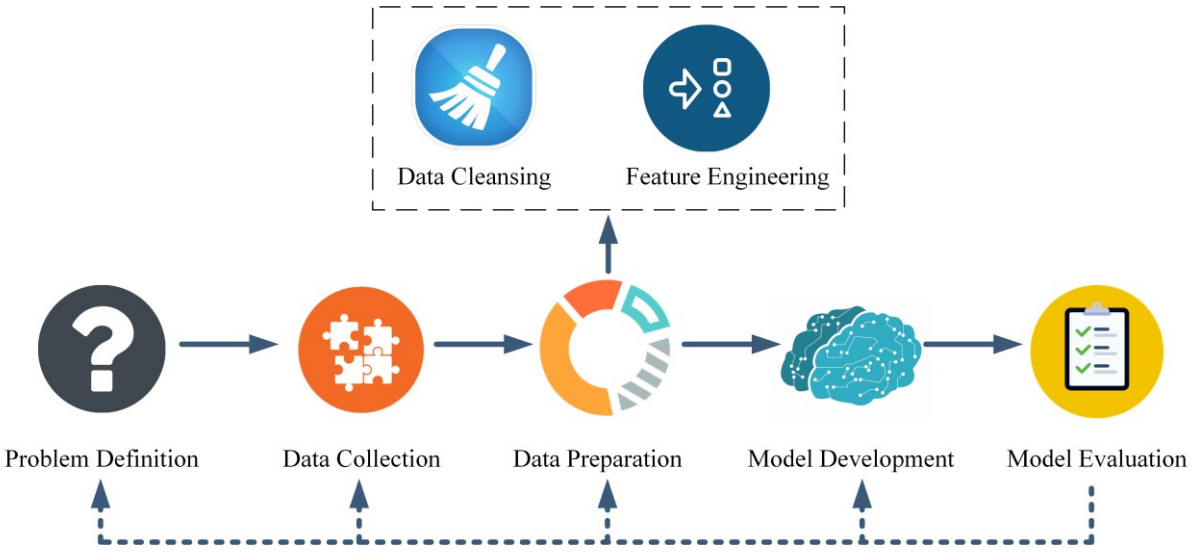


Figure 1.1: Overview of machine learning workflow

networks, may require billions of labeled data points to achieve adequate performance. However, in most real-world applications such as healthcare and financial applications, domain experience is needed to execute, or at least oversee, the labeling process. Hence, obtaining labeled datasets has become an expensive yet indispensable task in the machine learning pipeline.

To tackle the challenges of generating training data, practitioners have recently turned to weak labels to reduce the cost of human efforts spent in labeling data. Weak labels refer to inaccurate or incomplete labels that are generated from cheaper annotation sources such as crowdsourcing and user-defined heuristics [3]. However, utilizing these imperfect labeling sources can lead to other challenges. First, the outputs of these sources often overlap and conflict, which requires further debugging to integrate their output. Second, the noise level in the output labels can deteriorate the performance of the learning model. Therefore, extra preprocessing steps are needed to either fix these noisy labels or prepare the learning algorithm to become more robust to noise.

Moreover, aside from the challenge of learning from mislabeled examples, data cleansing is another essential process in data preparation for data analytics. Data cleaning refers to a set of operations required to clean data by either removing outliers, replacing missing values, smoothing noisy data, and correcting inconsistent data. Machine learning models are expected to consume a variety of different data coming from sensors, IoT devices, wearable devices, and so forth. However, methods of data collection are often loosely controlled, and therefore, result in out-of-

range values, impossible data combinations, missing values, and different kinds of errors. Thus, since data quality issues can lead to "garbage in, garbage out" in machine learning, error detection is considered as a critical step to maintain a stable machine learning pipeline. Overall, all these challenges disrupt the profound power of machine learning. Therefore, we could eventually build a more powerful machine learning pipeline by automating some of these data preprocessing activities.

1.2. Objectives and Originality

To sum up, we intend to propose a set of algorithms to deal with the challenges associated with data preparation in real-world applications, especially while considering big data. We also apply these algorithms to build innovative frameworks to provide automated data labeling and repairing. A roadmap of the thesis is illustrated in Figure 1.2. The primary objectives of this study are further listed as follows:

- **Generating labeled training data.** As a fundamental requirement, supervised models need large labeled datasets. To address this challenge, we propose a novel hybrid method that integrates the scalability of weak supervision with the user engagement and accuracy of semi-supervised learning to optimize the labeling process.
- **Ensuring the quality of the generated labels for big data.** We consider more complicated settings in generating labeled training data. As the size of the data grows, relying only on weak supervision sources could be problematic. Since the quality of the generated labels presents an issue, we propose an end-to-end framework to generate high-quality, large-scale labeled datasets. The system, first, automatically generates heuristics to assign initial labels. Then, the framework applies a novel data-driven active learning process to enhance the labeling quality.
- **Maintaining a satisfactory level of performance of machine learning models in production.** Applying machine learning in business-to-business situations imposes specific requirements. Aiming at providing an integrated solution, we propose an end-to-end framework that aims at guiding businesses in designing, developing, evaluating, and deploying business-to-business predictive systems. The framework employs the Lean Startup methodology and aims at maximizing the business value while eliminating wasteful development practices.

- **Learning with the presence of weak supervision.** Although recent efforts try to enable learning models to work with weakly supervised datasets, they treat each type of weak supervision individually. However, in real-world cases, different types of weak supervision tend to occur simultaneously. Therefore, we present a classification model that applies semi-supervised ensemble learning and data-driven rectification to deal with inaccurate and incomplete supervised datasets.
- **Applying machine learning to erroneous data detection.** Existing error detection techniques are typically targeted to detect certain types of errors. Moreover, most of these detection models either require user-defined rules or ample hand-labeled training examples. Therefore, we present a model that learns bidirectional encoder representations for tabular data. Then, the model utilizes these representations to find erroneous data. The model applies a data augmentation module to generate more erroneous examples to represent the minority class.

1.3. Organization

Following the objectives above, the rest of the thesis is organized as follows:

- In Chapter 2, we briefly review some existing methods for generating training datasets. Then, the chapter presents *WeSAL*, a labeling algorithm that combines Weak Supervision with Active Learning to create labeled training data. WeSAL aims at enhancing the scalability of active learning while benefiting from weak supervision.
- In Chapter 3, we first summarize the challenges that face machine learning in the business domain. Then, we propose *M-lean*, which is a framework that aims at guiding businesses to derive value from their data through building Business-to-Business (B2B) [4] predictive systems. The framework utilizes various research designs through a set of phases to qualify the business value of the final model. The chapter also introduces a case study in which the proposed framework is applied, with the help of our industrial partner, IBM, to build a B2B predictive system for software license cancellations.
- In Chapter 4, we refine the labeling algorithm presented in Chapter 2 in terms of 1) analyzing the cost of obtaining user-defined heuristics for big datasets, and 2) the effectiveness of traditional active learning to sustain good performance when faced with higher levels of noise. As a result, we present *Asterisk*, a framework to generate high-quality training datasets at scale.

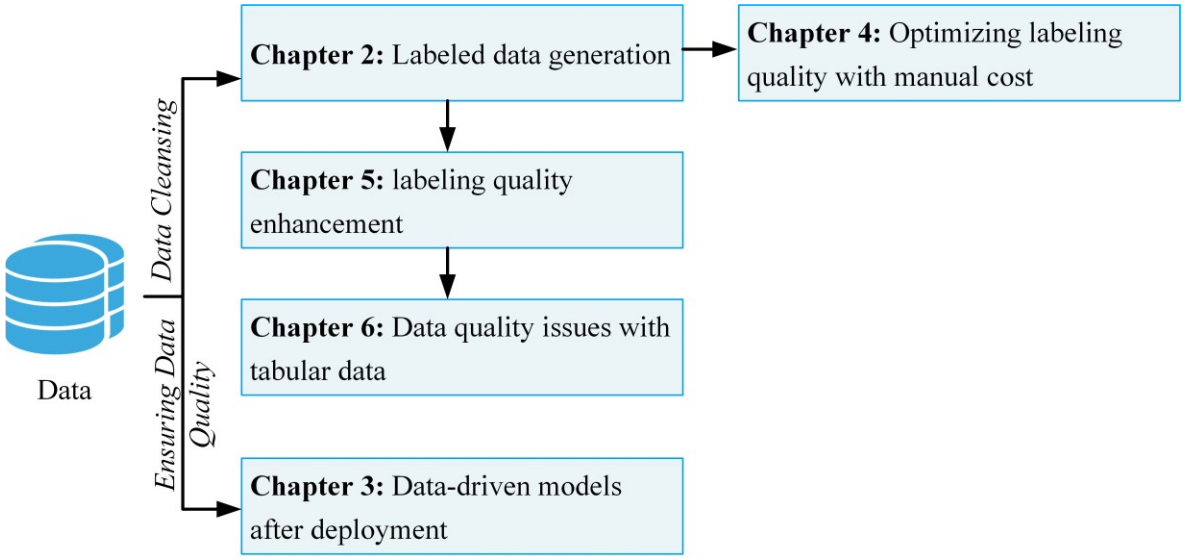


Figure 1.2: A general roadmap of the thesis

Instead of depending on the end-users to provide user-defined heuristics, the proposed automatically produces a set of heuristics by exploiting a small labeled dataset. Then, the system examines the disagreements between these heuristics to model their accuracies and applies a novel data-driven AL process to enhance the quality of the final labels.

- In Chapter 5, we propose *Smart Mendr*, a classification Model that applies Ensemble Learning and Data-driven Rectification to handle inaccurate and incomplete supervision. The proposed model applies a preliminary stage of semi-supervised ensemble learning to estimate the probability of each instance being mislabeled. Then, the proposed method applies a smart correcting procedure using meta-active learning to provide correct labels for both noisy and unlabeled points.
- In Chapter 6, we examine the problem of erroneous data detection in tabular data and present, *TabReformer,* an end-to-end framework for that can model data representation in tabular databases. The structure of the proposed framework includes a novel learning objective for tabular data along with a data augmentation module. The system does not require any user-defined parameters; that is, it is fully-automated and assumes no domain-specific knowledge.

References

[1] R. Caruana, N. Karampatziakis, and A. Yessenalina, "An Empirical Evaluation of Supervised Learning in High Dimensions," in *Proceedings of the 25th International*

Conference on Machine Learning, 2008.

[2] "What data scientists tell us about AI model training today," Alegion, 2019. [Online]. Available: https://content.alegion.com/dimensional-researchs-survey.

[3] H. Zamani and W. B. Croft, "On the theory of weak supervision for information retrieval," in *ACM International Conference on Theory of Information Retrieval*, 2018.

[4] M. Vlachos, V. G. Vassiliadis, R. Heckel, and A. Labbi, "Toward interpretable predictive models in B2B recommender systems," *IBM Journal of Research and Development*, 2016.

Chapter 2 : Applying Active Supervision to Find High-quality Labels at Industrial Scale

2.1. Introduction

Machine learning models are deployed in many domains to empower data-driven decisions. However, supervised machine learning models require access to labeled training datasets [1]. Obtaining such labeled data is a significant bottleneck in creating learning models, especially with the current popularity of data-greedy methods such as deep learning models that may require millions of labeled data points. As a result, acquiring labeled datasets turns out to be an expensive yet indispensable task in the machine learning pipeline.

Aiming to tackle this challenge, there is ample research [1]–[3] offering solutions to generate labeled training data. Active learning (AL) [2] can be seen as a labeling approach that aims at optimizing labeling cost and classification accuracy. For example, in pool-based AL [2], the learning algorithm iteratively selects data points from a pool of unlabeled points. Since the algorithm queries the user about the most informative points, the resulting model is assumed to achieve better classification performance with fewer labels.

While AL tries to engage human oracles to provide true labels, there is a growing interest in using weak supervision sources [3]. Weak supervision relies on obtaining low-quality, but large-scale training datasets by exploiting cheaper annotating approaches. To integrate training labels from these weak sources, previous studies [1], [4], [5] used generative models [6] to learn the accuracy of such sources and model the true label as a latent variable [4].

However, several questions regarding these approaches remain to be addressed. On the one hand, AL can be expensive with high-dimensional datasets [7]. For instance, the unbalance between the sizes of labeled and unlabeled data can slow the labeling process. Also, previous research [8] indicates that, when dealing with imbalanced data distributions, AL can result in low performance. On the other hand, weak supervision outputs noisy labels that affect model performance. The uncertainty of the generated labels complicates the process of learning the structure of the

generative models [6]. Also, since weak sources often overlap and conflict, debugging these sources can be time-consuming [5].

Therefore, motivated by the shortcomings of these approaches, we present WeSAL, a labeling approach that combines Weak Supervision with Active Learning to create large-scale, high-quality training data. WeSAL extends weak supervision and includes humans-in-the-loop to denoise the weak labels. It tries to overcome the scalability issues of AL by reducing the size of unlabeled pools to only contain conflicting points. Therefore, WeSAL profits from the scalability of weak supervision while economically applies user engagement to enhance labeling accuracy.

Figure 2.1 illustrates an overview of WeSAL; the approach starts by collecting labels from different weak sources. Although WeSAL can work with any weak supervision sources, we focus on user-defined heuristics since they are the most popular methods to generate noisy labels for real-world tasks [4]. Afterward, these labels are examined to create an unlabeled pool. Next, the user is queried about the most informative points. Then, the obtained labels from the AL process are used to refine the initial noisy labels. After that, a generative model is used to model the accuracy of the refined heuristics and generate probabilistic labels. Finally, these labels are used to train any model to produce predictions for the desired learning task.

To evaluate WeSAL, we compare it with two state-of-the-art techniques, data programming (DP) [1] and AL. The experiments aim at assessing the effectiveness of WeSAL in producing accurate labels in terms of labeling accuracy, labeling budget, and classification performance. The experiments include a sensitivity analysis of the parameters used in the experiments to study their impact on the performance.

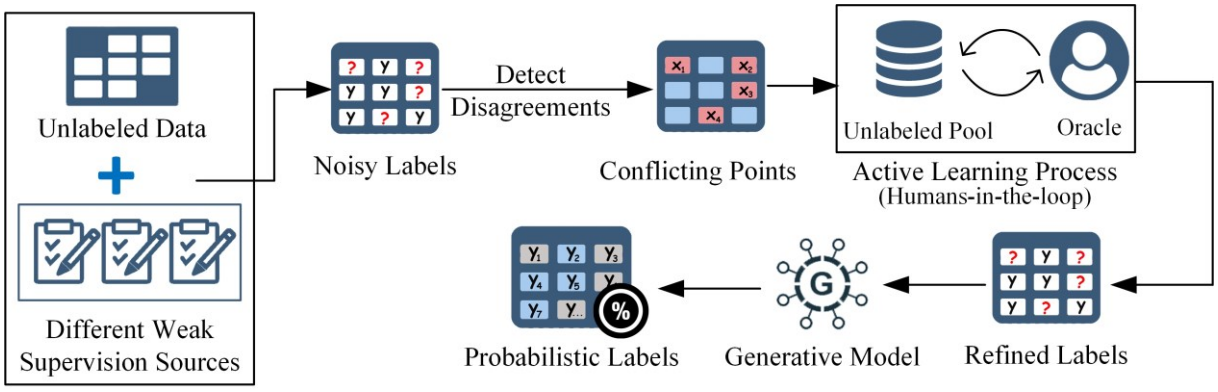


Figure 2.1: Overview of the proposed method

The chapter is structured as follows: Section 2.2 discusses the related background. Section 2.3 presents the proposed method. The experimental results are offered in Section 2.4. While Section 2.5 discusses related work; and Section 2.6 concludes the chapter.

2.2. Background

In this section, we first discuss active learning. Then, we overview weak supervision techniques and the data programming paradigm.

2.2.1 Active learning

Active learning helps to generate labels with minimum labeling effort [2]. In pool-based AL, a classifier starts with having access to a pool of unlabeled examples, a set of labeled points (the seed), and a test set. Initially, the classifier is trained using the seed. Then, points in the unlabeled pool are ranked, and the most informative points are chosen to query an oracle, then used to train a classifier and evaluate its performance on the test set. Given the new status of the classifier, the points in the unlabeled pool are ranked again, and the process is repeated. AL process stops based on a stopping criterion [2], for example when a target performance is reached. The part that selects the points from the unlabeled pool is the query strategy. Over the past decades, several query strategies are proposed. One of the most effective query strategies is uncertainty sampling [2]. It selects the points about which the classifier is most uncertain. Another query strategy is Query-by-committee [2], which operates similarly as uncertainty sampling, except it uses a committee of classifiers and chooses the points about which the committee members disagree.

Nevertheless, many research articles [9]–[12] point out that AL suffers from many challenges, particularly that AL algorithms are binary methods and do not scale to multi-classification settings [11], [12]. Another problem of AL originates from the complexity of the ranking step [9], [10], especially with large scale unlabeled pools. In these cases, AL becomes an expensive solution. Another study [13] states that training datasets built with AL can contain labels with biased distribution for the chosen model. As a result, we believe that many questions exist regarding the performance of AL when applied to large scale datasets. To address and overcome these issues, WeSAL aims to speed up the ranking procedure and reduce the size of the labeling pool. The solution helps to resolve the unbalance between the labeled and unlabeled data and hence,

enhances the scalability of AL. The experiments show that AL annotation costs can be deducted by 36% using the proposed method.

2.2.2 Weak supervision

In recent years, weak supervision [3] has been gaining popularity in generating labels. In weak supervision, domain experts are asked to provide some form of higher-level, low-quality supervision such as user-defined heuristics. The results of such forms are programmatically generated data, which is noisy and contains conflicting labels. As a result, the problem of integrating these diverse sources remains open [1], [5], [6]. DP [1] is a paradigm proposed to integrate labels generated from weak sources. In DP, weak supervision sources are encoded as labeling functions [4], which are arbitrary scripts that translate different weak sources. After applying these functions, DP uses generative models to learn the accuracies of the labeling functions without access to labeled data [4]. DP applies structure learning techniques to model the true class labels as latent [6]. Finally, the generative model outputs a set of probabilistic training labels that can be used to train any discriminative model.

Depending on high-level supervision, DP generates labels with a noise level that is hard for the end-users to evaluate. Also, the complex structure of the generative model makes it challenging for users to debug its outcome [14]. Therefore, studies [14], [15] have tried to overcome these limitations. One study is Socratic Learning [15], which is a technique to debug generated labels by examining the disagreements between the training data and the generated labels. However, since Socratic Learning is an automated method that does not utilize domain experience in the refinement process, end users may have problems in understanding its decisions [14]. To overcome this lack of explainability, Varma *et al.* [14] proposed a visual framework to interpret these decisions. However, the framework does not explain the structure of the generative model, which users often struggle to understand.

Overall, we find that since weak supervision results in noisy conflicting labels, previous studies have exclusively focused on learning the structure of generative models to enhance the labeling quality. However, none of these studies explored the effect of utilizing domain expertise to denoise the output labels. Therefore, in WeSAL, end users are asked to refine the disagreements between the labeling functions by providing labels for the conflicting points. Many researchers [4], [14],

[15] have demonstrated that resolving these disagreements enhances accuracy and helps better identify latent subsets in the training data. WeSAL employs domain expertise to perform this task to improve both the labeling quality and help end-users evaluate the accuracy of the weak sources. The experimental results show that WeSAL managed to enhance labeling accuracy by up to 26% when compared to data programming.

2.3. WeSAL: The proposed method

Let us assume we have a set of unlabeled inputs X of size N denoted as $\{\mathbf{x}_i\}_{i=1}^{N}$ where $\mathbf{x}_i$ represents a set of features describing the i^{th} data point in X, and a set of unknown labels y as $\{y_i\}_{i=1}^{N}$ where $y_i \in \{-1, 1\}$. WeSAL starts by allowing the users to write a group of T labeling functions F denoted as $\{f_j\}_{j=1}^{T}$, where f_j: $\mathbf{X} \rightarrow \{-1, 0, 1\}$. Each labeling function creates a weak label for x_i, where 0 describes abstaining. Therefore, the result of applying all functions F to X is a noisy label matrix L where:

$$L_{i,j} = f_j(\mathbf{x}_i) \text{ where } 1 \leq i \leq N \text{ and } 1 \leq j \leq T \quad (2.1)$$

To model the accuracy of the labeling functions, DP [1] forms a generative model G as a factor graph Ø. The graph is encoded using three factors, namely, labeling propensity $Ø^{lab}{}_{i,j}(F, Y) = \mathbf{1}\{f_{i,j} \neq 0\}$, labeling accuracy $Ø^{Acc}{}_{i,j}(F, Y) = \mathbf{1}\{f_{i,j} = y_i\}$, and functions pairwise correlation $Ø^{Corr}{}_{i,j,k}(F, Y) = \mathbf{1}\{f_{i,j} = f_{i,k}\}$ where $j, k \in M$ where M is a set of labeling function pairs (j, k) modeled as dependent [6].

Since these labeling functions rely on imperfect sources, they abstain and conflict with each other. Consequently, WeSAL resolves pairwise disagreements between the labeling functions to increase their accuracy. The pairwise disagreements can be defined as:

$$Ø^{dis}{}_{i,j,k}(F, Y) = \mathbf{1}\{f_{i,j} \neq f_{i,k}\} \text{ where } j, k \in M, i \in N \quad (2.2)$$

Moreover, WeSAL tries to resolve abstaining situations to increase the coverage of the resulting training labels. The abstaining labels are denoted as:

$$Ø^{abstain}{}_{i,j}(F, Y) = \mathbf{1}\{f_{i,j} = 0\} \quad (2.3)$$

Next, the proposed method constructs an unlabeled dataset P_U of size U where:

$$P_U \subseteq \mathbf{X}, \forall x_i \in P_U \{x_i | \emptyset^{dis}{}_{i,j,k}(F,Y) = \mathbf{1}\{f_{i,j} \neq f_{i,k}\} \cup \emptyset^{abstain}{}_{i,j}(F,Y) = \mathbf{1}\{f_{i,j} = 0\} \quad (2.4)$$

Therefore, to enhance the accuracy of the labeling functions, WeSAL applies AL to provide true labels and introduce domain experience. The AL component proceeds by choosing points from P_U that are assumed to be beneficial to the classifier according to a predefined query strategy. There are several types of query strategies that can be applied, which include uncertainty sampling, query-by committee, and random sampling. WeSAL applies uncertainty sampling as the default query strategy. We have selected uncertainty sampling as it is one of the most commonly used query strategies. Also, uncertainty sampling shows superiority over other query strategies in the experiments (Section 3.4.3.2). Uncertainty sampling only queries the instances about which the model is least confident. The strategy iteratively ranks the pool and considers data point with the least confident score using the well-known entropy measure as:

$$x_H = arg\max_x \sum_i P_\theta(y_i|x) \log P_\theta(y_i|x) \quad (2.5)$$

where $P_\theta(y_i|x)$ is the a posteriori probability of class y_i and Where $y_i$ranges over all possible classes.

It is essential to state that the performance of the proposed method depends on the quality of the labeling functions provided by end-users. Since it is assumed that the users write labeling functions that perform better than random (with accuracy values more than 50%) [4], [5], a significant portion of the unlabeled data should receive labels before applying AL. However, in the worst-case scenario, when end-users provide low-quality labeling functions, the proposed method will be reduced to a traditional process of applying active learning to the entire unlabeled data.

As a result, in most cases, P_U in the proposed method will only represent the conflicting points between the labeling functions. Hence, the size of P_U should be much smaller than the size of X. Therefore, the ranking time in WeSAL is reduced compared to traditional AL in which all the points in X are ranked at each iteration. Also, as for computational complexity, WeSAL can scale to much larger datasets than traditional active learning since it runs in O(W.U) where W is the number of queries consumed by the AL component in WeSAL and U is the size of P_U.

Furthermore, we ask users to specify a value for the maximum number of points they are willing to label and set this number as a labeling budget $B_{Labeling}$. Hence, AL process terminates when either all the disagreements are resolved (all data points in P_U are labeled) or the labeling budget

is exhausted. Then, the output of AL $(X, Y)_{AL}$ can be described as $\{\mathbf{x}_i, y_i\}_{i=1}^{D}$ where D=min(U, $B_{Labeling}$). WeSAL then uses $(X, Y)_{AL}$ to denoise L as:

$$L_{refined\ i,j} = \begin{cases} y_i & if(x_i, y_i) \in (x, y)_{AL} \\ L_{i,j} & otherwise \end{cases} \quad j = 1,2, \dots T \tag{2.6}$$

Refining the noisy label matrix L increases the empirical probability of the labeling functions f_i and f_j agreeing. The empirical probability can be described as $P_{i,j} = \frac{a}{N}$ where a is the number of agreements between f_i and f_j. Since the refinement process increases a, the empirical probability increases accordingly, and hence, the accuracy of the labeling functions is enhanced.

Then, WeSAL applies a generative model G that uses the refined label matrix $L_{refined}$ to generate a set of probabilistic labels to train a downstream classifier of choice. G can be formally defined [15] as,

$$G: \pi_{\emptyset}(L_{refined}, Y) = \frac{1}{Z_{\emptyset}} e^{\emptyset^T L_{refined}\, Y} \tag{2.7}$$

where $Z_{\emptyset}$ is a partition function to guarantee π is a distribution, and Ø represents the average accuracy of the labeling functions [15]. As seen in (2.7), the generative model learns the accuracy of the labeling functions from their disagreements. Therefore, refining L improves the quality of the final labels. The complete algorithm of the proposed method is shown in Algorithm 1. Although there are other approaches [6], [15] that aim at denoising the generated labels of the DP pipeline, none of these methods have employed domain experience in this process. Therefore, we believe that our approach is the first attempt that tries to include humans in the loop in the form of AL within the weak supervision process.

Algorithm 2.1: WeSAL, The Proposed Method

Input: Input data set X with unknown labels Y, selected query strategy q for Active learning, labeling budget $B_{Labeling}$.

Output: Probabilistic labels $y^* = P[y = 1] \in [0,1]$.

1: Write a set of labeling functions $F = \{F_1, F_2, \dots F_t\}$

2: Apply F to X to create a noisy label matrix L

3: Construct disagreements factor $Ø^{dis}(F, Y)$

4: Construct abstaining labels factor $Ø^{abstain}(F, Y)$

5: Initialize $P_U = \{\}$

6: **Loop** until i>N

7: **If** $\emptyset^{dis}_{i,j,k}(F,Y) = 1$ **then** $P_U \cup \{x_i\}$

8: **If** $\emptyset^{abstain}_{i,j,}(F,Y) = 1$ **then** $P_U \cup \{x_i\}$

9: $i \leftarrow i+1$.

10: **End**

11: Initialize $(X,Y)_{AL} = \{\}$

12: **Loop** until stopping criterion is met

13: Select a point x_i from P_U using q

14: query the user to provide a label y_i for x_i

15: $P_U = P_U - x_i$

16: $(X,Y)_{AL} = (X,Y)_{AL} \cup (x_i, y_i)$

17: Train classifier using $(X,Y)_{AL}$

18: **End**

19: denoise L using $(X,Y)_{AL}$ to create $L_{refined}$

20: Train generative model G with $L_{refined}$ to output y^*

2.4. Evaluation

The experiments seek to validate two points. First, how accurately can WeSAL generate labels for real tasks. Second, what is the impact of using WeSAL on the labeling cost. To validate the first point, we compare WeSAL to DP [4] and evaluate the performance of the generative and the discriminative models. Also, we report the accuracy of the generated labels. For the second point, we compare WeSAL against AL and report the labeling cost and the performance of the final classifiers. Although there are other labeling approaches [5], [15], [16], the experiments consider active learning and data programming since WeSAL extends these two approaches. However, future work should include evaluations against different labeling methods, such as transfer learning [16]. Also, the primary goal of WeSAL is to build better predictive models for various classification tasks. Since training models with accurate labels improves their capability to

Table 2.1: Overview of the datasets. Dim is the dimensionality of the dataset. +/Size is the positive class to the dataset size ratio.

Dataset	Size	Dim.	+/ Size
Renewal Sales	1,354,704	11	73.06
Bank	45,211	17	11.70
News	39,797	61	49.34
Credit Card	30,000	24	22.12
Occupancy Detection	20,560	7	23.10
MNIST	70,000	784	-

generalize to unseen observations [1], [4], we report the classification accuracy of the learning models trained with the generated labels.

2.4.1 Datasets

We consider generating training labels for real-world tasks over five open-source datasets along with a real business dataset. Summary statistics are provided in Table 2.1. As for the first dataset, **Renewal Sales** is a business dataset provided by our industrial partner, IBM. The dataset contains more than 1.3 million records of anonymized renewal records describing historical transactions of software subscriptions. The dataset is used in a classification task to predict license cancellations. Another business task is the Bank Marketing dataset **(Bank)** with a classification goal of predicting campaign subscriptions via marketing calls. The default of credit card dataset **(Credit Card)** is used to predict the default payments. The Online News Popularity Dataset **(News)** is a social

Table 2.2: Experimental settings

Dataset	Data Programming Settings						Active Learning Settings		
	# Candidates	# Labeling Functions	Labeling Functions Performance				Initial seed	Train set size	Test set size
			Accuracy	Precision	Recall	F1			
Renewal Sales	1,083,763	4	0.75	0.78	0.75	0.76	67,735	839,917	447,052
Bank	36,169	5	0.77	0.78	0.80	0.79	2,260	28,031	14,920
News	31,716	6	0.74	0.82	0.78	0.80	1,989	24,675	13,133
Credit Card	24,001	5	0.67	0.71	0.72	0.72	1,500	18,600	9,900
Occupancy Detection	16,448	7	0.78	0.81	0.78	0.80	1,028	12,747	6,785
MNIST	56,000	5	0.77	0.79	0.69	0.74	3500	43,400	23,100

dataset to predict the level of popularity of online articles. The fifth data is the Occupancy Detection dataset **(Occupancy Detection)**, which represents a binary classification task for room occupancy. These datasets are all publicly available and were downloaded from the UC Irvine Machine Learning Repository[1]. Moreover, to add an example of a multi-classification situation, the **MNIST** dataset is added to the experiments, which consists of 70K images of hand-writing digits with ten classes.

2.4.2 Experiments settings

Writing the labeling functions. To compare WeSAL with DP, we use Snorkel [4], which is an end-to-end DP framework. To implement the labeling functions, we focus on threshold-based labeling functions [4], [5] in which the labeling functions assign labels to each data instance or abstain based on values of specific features in the data (e.g., values of client's bill statements may influence their default payment). As for the renewal sales dataset, we consulted a set of sales representatives from IBM to help us write the labeling functions. The research team (the first two authors) elicited a set of business rules from end-users and used these rules to write the labeling function. Then, the sales representatives reviewed these functions thought a set of code walkthroughs. As for the other datasets, we relied on pattern matching, which is a consistent approach with best practice found in the literature [4], [5], [14].

Validating the labeling functions. To only accommodate high accuracy sources, we used a set of labeled data (gold labels) to develop labeling functions. We calculated the empirical accuracy of the labeling functions concerning the gold labels. Also, we set an accuracy threshold of 60% and only included the functions that exceed this threshold. Table 2.2 shows the experimental settings. As for the DP settings, the table shows the number of candidates (records) for which labels are generated, the number of labeling functions, and the evaluation buckets (Accuracy, Precision, Recall, and F1 measure) for the labeling functions.

Active Learning settings. We compare WeSAL against different sampling techniques of AL, namely uncertainty sampling (UNC), query-by committee (QBC), and random sampling (RAND).

[1] https://archive.ics.uci.edu/ml/index.php

The results of AL experiments are averaged over ten runs. The general settings used in AL experiments are illustrated in Table 2.2. For each dataset, the table shows the seed, the initial size of X_{train}, and the size of the test set D_{test} used to evaluate the classifier. Following best practice in the literature [2], [8], 5% of each dataset is randomly sampled as the initial seed, 33% is used as the testing set, and the rest is treated as the unlabeled pool.

Also, to decide on the stopping criteria for AL, we examined the learning curves and stopped the process when the classifier performance shows no improvement with additional iterations [17]. We use $\lambda = 0.0001$ as a threshold of performance differences and stop the experiments when the mean of performance differences does not exceed λ for a successive number of iterations. Furthermore, since the active learning process highly depends on the value of λ, we experiment with different values of λ (Section 2.4.4) to observe its effect on the overall performance of both AL and the proposed method. Moreover, to use the same conditions throughout the experiments, we use the number of labels required to satisfy the performance stability condition as the labeling budget $B_{Labeling}$ for the proposed method.

2.4.3. Experiments results

In this section, we present the results of comparing WeSAL to DP and AL.

3.4.3.1. WeSAL vs. DP

First, we compare WeSAL to DP using the same labeling functions. Table 2.3 shows the results in terms of the performance of the generative and the discriminative models. Reporting the performance of the discriminative models assesses the effect of the improved labeling accuracy on the performance of the learning models. To avoid measurement bias, we report a wide range of performance measures. As for the generative model, we report Precision (P), Recall (R), and F1 measure (F1). We calculate the same measures for the discriminative model, along with Matthews correlation coefficient (MCC). MCC considers the four factors of the confusion matrix and calculated as $\frac{TP*TN-FP*FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}$. The table also shows the labeling accuracy, which is calculated as the ratio of the number of correct labels to the size of the training set.

The results show that, with regard to the generative model, WeSAL achieved higher performance in all tasks. Since the generative model performance depends on the labeling functions, this

Table 2.3: Data programming results

Dataset	WeSAL								Data Programming							
	Generative Model			Labeling Accuracy	Discriminative Model				Generative Model			Labeling Accuracy	Discriminative Model			
	P	R	F1		P	R	MCC	F1	P	R	F1		P	R	MCC	F1
Renewal Sales	**0.94**	**0.88**	**0.91**	**0.84**	**0.89**	**0.90**	**0.90**	**0.89**	0.87	0.75	0.81	0.68	0.86	0.75	0.78	0.80
Bank	**0.89**	**0.82**	**0.85**	**0.77**	**0.87**	**0.86**	**0.87**	**0.86**	0.64	0.71	0.67	0.61	0.84	0.74	0.77	0.79
News	**0.87**	**0.80**	**0.83**	**0.59**	**0.88**	**0.97**	**0.96**	**0.92**	0.75	0.73	0.74	0.49	0.85	0.92	0.89	0.88
Credit Card	**0.85**	**0.77**	**0.81**	**0.37**	**0.88**	**0.73**	**0.75**	**0.80**	0.83	0.71	0.77	0.34	0.87	0.65	0.71	0.74
Occupancy Detection	**0.94**	**0.81**	**0.87**	**0.75**	**0.90**	**0.94**	**0.95**	**0.92**	0.82	0.78	0.80	0.67	0.87	0.83	0.84	0.85
MNIST	**0.88**	**0.93**	**0.90**	**0.59**	**0.88**	**0.95**	**0.95**	**0.91**	0.73	0.74	0.73	0.51	0.84	0.83	0.84	0.83

empirically proves the effectiveness of WeSAL in enhancing the accuracy of the labeling functions. WeSAL managed to improve the F1 score of the generative model by 27% and 23% in the Bank and MNIST datasets, respectively. The reason for this improvement is that since the quality of the labeling functions were good (0.79 and 0.74 as F1 (Table 2.2)), the labeling budget was effectively spent to resolve the disagreements between the functions, and hence improve the overall performance. Moreover, WeSAL surpassed DP in discriminative model performance within all datasets. Since providing accurate data to the discriminative model improves its capability to generalize to unseen observations, this proves that WeSAL enhances the quality of the learning models.

As for the labeling accuracy, WeSAL achieved better values than DP in all datasets. In some problems such as the Bank dataset, WeSAL improved the labeling accuracy by 26% when compared to DP. Alternatively, in the credit card dataset, WeSAL achieved a relatively small enhancement of 9%. The reason behind that is the low accuracy of the labeling functions used in the credit card dataset. Therefore, WeSAL could only resolve a small portion of the conflicts, and hence, could not achieve a significant accuracy boost. Overall, WeSAL managed to enhance labeling accuracy by an average of 18% when compared to DP.

Table 2.4: Active learning results

Dataset	WeSAL					Active Learning				
	P	R	MCC	Acc.	# queried instances	P	R	MCC	Acc.	# queried instances
Renewal Sales	**0.98**	**0.98**	**0.91**	**0.98**	**73,320**	**0.98**	0.96	0.84	0.95	125,988
Bank	**0.79**	**0.91**	**0.82**	**0.97**	**2,151**	0.71	0.70	0.66	0.93	3,364
News	**0.93**	**0.95**	**0.85**	**0.92**	**4,374**	0.89	0.90	0.80	0.90	13,818
Credit Card	**0.75**	**0.84**	**0.73**	0.90	**12,958**	0.73	0.80	0.67	**0.91**	**12,958**
Occupancy Detection	**0.75**	**0.98**	**0.81**	**0.94**	**7,283**	0.72	0.82	0.70	0.90	11,855
MNIST	**0.92**	**0.97**	**0.92**	**0.95**	**2,452**	0.88	0.95	0.84	0.92	3,472

3.4.3.2. WeSAL vs. AL

In this part, we compare WeSAL to AL. First, to determine the labeling budget for WeSAL, we applied three query strategies to the datasets. Figure 2.2 shows the learning curves using UNC, QBC, and RAND query strategies. The learning curves illustrate the relationship between the number of queried points and classifier accuracy. Since the curves show that UNC achieved the highest accuracy in all the datasets, we report the evaluation metrics obtained by WeSAL and UNC in Table 2.4. Similar to the experiments with DP, we report the performance of the learning models to assess the influence of the generated labels to the underlying classification tasks. The table also shows the number of queried instances required to obtain the equivalent accuracy values.

The table depicts that WeSAL achieved better MCC values in all the problems with the most significant improvements in the Bank dataset of 24% comparing to AL. Also, the results show that WeSAL did not need to use the labeling budget assigned by AL in most of the problems. Since the size of P_U is much smaller than the size of X_{train}, WeSAL managed to resolve all the disagreements between the labeling functions without exceeding $B_{Labeling}$. For example, while AL needed to label 12% of the training dataset in the Bank dataset, the size of P_U only represents 8% of X_{train}, hence a decrease ratio of 36% in labeling cost. Similarly, WeSAL managed to decrease the labeling cost in Renewal Sales and Occupancy Detection datasets by 42% and 39%, respectively. The only

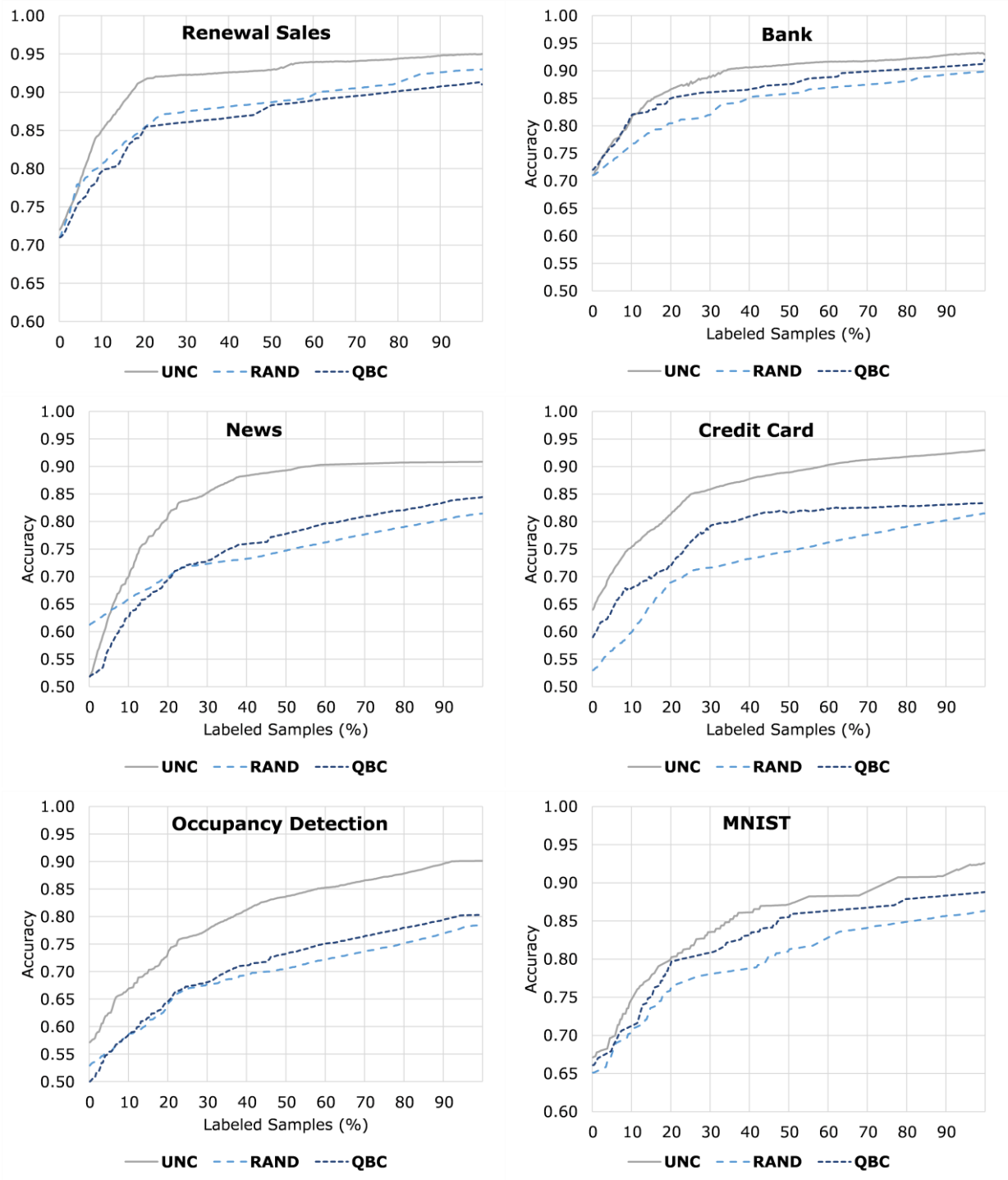


Figure 2.2: Learning curves of active learning

dataset in which WeSAL exceeded the assigned budget is the credit card dataset. The reason for the increased labeling cost is due to the low accuracy labeling functions in this task, which result in a large number of disagreements that surpassed the assigned labeling budget. We, however, find

this point agrees with our conclusion of the importance of utilizing domain experience in the labeling process by designing labeling functions with high accuracy.

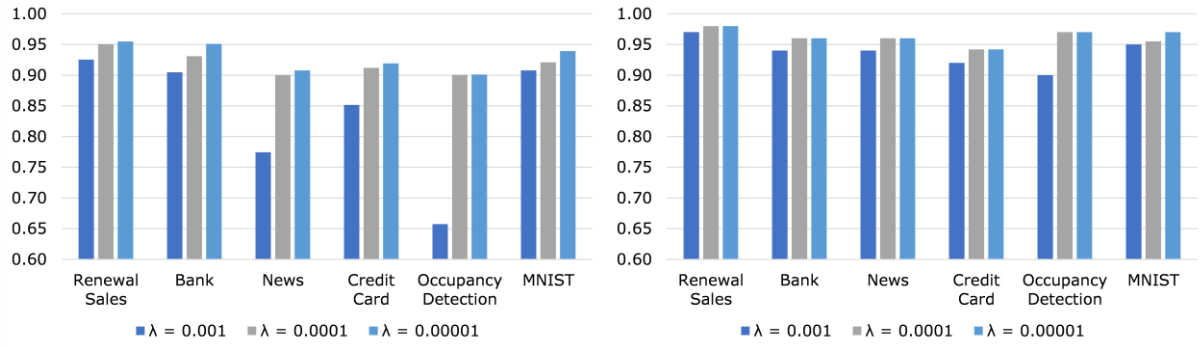
The results also attest that WeSAL outperformed AL in both precision and recall in all the problems. WeSAL managed to enhance the precision values achieved by AL by 10% and 4% in the Bank and the MNIST datasets. As for the recall values, WeSAL improved the performance of the machine learning models in all the problems with the highest enhancements in the Bank and the Occupancy Detection datasets by 30% and 20%, respectively. Overall, the results empirically prove that training models using labels generated by WeSAL results in remarkably improved performance, while reducing the labeling cost on real classification tasks.

2.4.4. Sensitivity analysis of the experimental parameters

In this section, we report the outcomes of the experiments under alternative assumptions of the parameters of the experiments.

2.4.4.1. Sensitivity analysis of the parameter λ

We stop the AL process once the arithmetic mean of performance differences for several iterations is less than a predefined threshold λ=0.0001. We also utilized the number of annotations required by AL as the labeling budget $B_{Labeling}$ in WeSAL. Therefore, to observe the effect of the parameter λ on the performance of both AL and the proposed method, the experiments were repeated with



(a) (b)

Figure 2.3: Accuracy values for (a) the classifiers in AL (b) the discriminative models in WeSAL with changing values of λ = 0.001, 0.0001, 0.00001

various values for λ. Figure 2.3.a shows the accuracy values reported by AL with values of λ = 0.001, 0.0001, 0.00001. Likewise, depending on the number of annotations consumed for each λ, the parameter $B_{Labeling}$ in WeSAL is adjusted accordingly. Table 2.5 shows, for each value of λ, in each dataset, the size of the initial unlabeled pool X_{train}, the number of queried labels at the end of the AL process as a percent of the size of X_{train} (AL Cost %). As for WeSAL, the size of P_U is assumed to be much smaller than the size of X_{train}. To highlight this point, the table shows the size of P_U as a percent of the size of X_{train} (P_U%) and the value of $B_{Labeling}$. Additionally, Figure 2.3.b shows the accuracy levels achieved by WeSAL for each value of $B_{Labeling}$.

As Figure 2.3.b depicts, choosing a larger value for λ may result in missing useful generalizations and force AL process to stop early [18]. For example, in the news, credit card, and occupancy detection datasets, setting λ =0.001 reduced the classifier accuracy in AL by 14%, 7%, and 27%, respectively, when compared to the performance achieved with λ =0.0001 (Figure 2.3.a). Also,

Table 2.5: Values of the experiments' parameters with different values of λ

Dataset	λ	Active Learning		WeSAL	
		Size of X_{train}	AL Cost %	P_U %	$B_{Labeling}$
Renewal Sales	0.001	839,917	7%	19%	61594
	0.0001		15%		125988
	0.00001		23%		195981
Bank	0.001	28,031	6%	8%	1682
	0.0001		12%		3364
	0.00001		40%		11306
News	0.001	24,675	16%	18%	3948
	0.0001		56%		13818
	0.00001		88%		21796
Credit Card	0.001	18,600	26%	72%	4836
	0.0001		70%		12958
	0.00001		83%		15438
Occupancy Detection	0.001	12,747	8%	57%	1020
	0.0001		93%		11855
	0.00001		97%		12365
MNIST	0.001	43,400	6%	6%	2459
	0.0001		8%		3472
	0.00001		68%		29657

Table 2.6: Performance of DP and WeSAL with different sets of labeling functions

Datasets	LFs Sets	Labeling functions		WeSAL (Discriminative Model)			DP (Discriminative Model)				
		Acc	F1	P	R	MCC	F1	P	R	MCC	F1
Renewal Sales	LF_{Best}	0.80	0.78	**0.88**	**0.90**	**0.90**	**0.89**	0.85	0.73	0.75	0.79
	$LF_{Mediocre}$	0.76	0.79	**0.85**	**0.89**	**0.87**	**0.87**	0.82	0.70	0.71	0.76
	LF_{Worst}	0.71	0.77	**0.81**	**0.89**	**0.81**	**0.85**	0.79	0.61	0.68	0.69
Bank	LF_{Best}	0.84	0.76	**0.84**	**0.86**	**0.85**	**0.85**	0.83	0.70	0.75	0.76
	$LF_{Mediocre}$	0.78	0.79	**0.76**	**0.81**	**0.80**	**0.78**	0.80	0.69	0.73	0.74
	LF_{Worst}	0.70	0.81	0.73	**0.80**	**0.79**	**0.76**	**0.77**	0.65	0.72	0.70
News	LF_{Best}	0.79	0.79	**0.86**	**0.90**	**0.92**	**0.88**	0.82	**0.90**	0.88	0.86
	$LF_{Mediocre}$	0.73	0.82	**0.82**	**0.88**	**0.90**	**0.85**	0.80	0.86	0.85	0.83
	LF_{Worst}	0.69	0.81	**0.79**	**0.84**	**0.85**	0.81	**0.79**	0.85	0.81	**0.82**
Credit Card	LF_{Best}	0.72	0.73	**0.90**	**0.89**	**0.86**	**0.89**	0.85	0.60	0.69	0.70
	$LF_{Mediocre}$	0.67	0.71	**0.88**	**0.85**	**0.81**	**0.86**	0.83	0.59	0.62	0.69
	LF_{Worst}	0.63	0.70	**0.86**	**0.80**	**0.78**	**0.83**	0.80	0.57	0.52	0.67
Occupancy Detection	LF_{Best}	0.85	0.79	**0.88**	**0.85**	**0.90**	**0.86**	0.86	0.82	0.80	0.84
	$LF_{Mediocre}$	0.77	0.79	**0.87**	**0.83**	**0.86**	**0.85**	0.84	0.81	0.76	0.82
	LF_{Worst}	0.70	0.85	**0.81**	**0.79**	**0.82**	**0.80**	0.83	0.78	0.71	**0.80**
MNIST	LF_{Best}	0.81	0.72	**0.85**	**0.87**	**0.91**	**0.86**	0.83	0.80	0.84	0.81
	$LF_{Mediocre}$	0.79	0.75	**0.82**	**0.87**	**0.91**	**0.84**	0.80	0.79	0.80	0.79
	LF_{Worst}	0.75	0.74	**0.80**	**0.80**	**0.88**	**0.80**	0.78	0.75	0.77	0.76

setting λ to a small value may enhance the performance but at the risk of wasting annotation effort. However, the figure shows no significant performance enhancement with λ=0.00001. Overall, the results show that the initial choice of λ =0.0001 was valid since, in most of the datasets, it succeeded in catching the elbow values in the learning curves, after which the performance changes become notably smaller.

Moreover, Figure 2.3.b shows that for most of the datasets, changing λ does not impose a big difference in the performance of WeSAL. The reason behind that, as mentioned before, is since the size of P_U is less than the size of X_{train}, the cost of annotating all the points in P_U may have an upper bound of a value less than the predefined $B_{Labeling}$. For example, in the bank, and the news datasets, WeSAL managed to fully annotate P_U with $B_{Labeling}$ corresponding to λ =0.0001 and 0.00001. On the other hand, in datasets such as the credit card and the occupancy detection

datasets, having a value of λ=0.001 suppressed the performance of WeSAL since the AL component could only resolve a portion of the disagreements. As a result, the performance is reduced by 2% and 7% in the credit card and occupancy detection datasets, respectively, when compared to the performance achieved with λ =0.0001 (Figure 2.3.b). Nevertheless, WeSAL still managed to achieve better results than AL in these two datasets. Overall, the results illustrated in Figure 2.3 show that the proposed method manages to achieve better performance than active learning with all variation of λ in all the datasets.

2.4.4.2. Sensitivity analysis of labeling functions

To estimate the effect of changing the accuracy of the labeling functions, we repeat the experiments in Section 2.4.3.1 using sets of labeling functions with varying levels of accuracy. For each dataset, we create three sets of labeling functions, namely LF_{Best}, $LF_{Mediocre}$, and LF_{Worst} by sampling the best, mediocre, worst three labeling functions from the original set (Table 2.2). The overall accuracy and F1 measures for each set are reported in Table 2.6, along with the performance of the discriminative model of both WeSAL and DP.

The results show that the discriminative model in WeSAL achieves better performance in all the problems. The table also illustrates that using a smaller number of labeling functions affects the coverage of the training set, and hence, negatively influences the discriminative models. However,

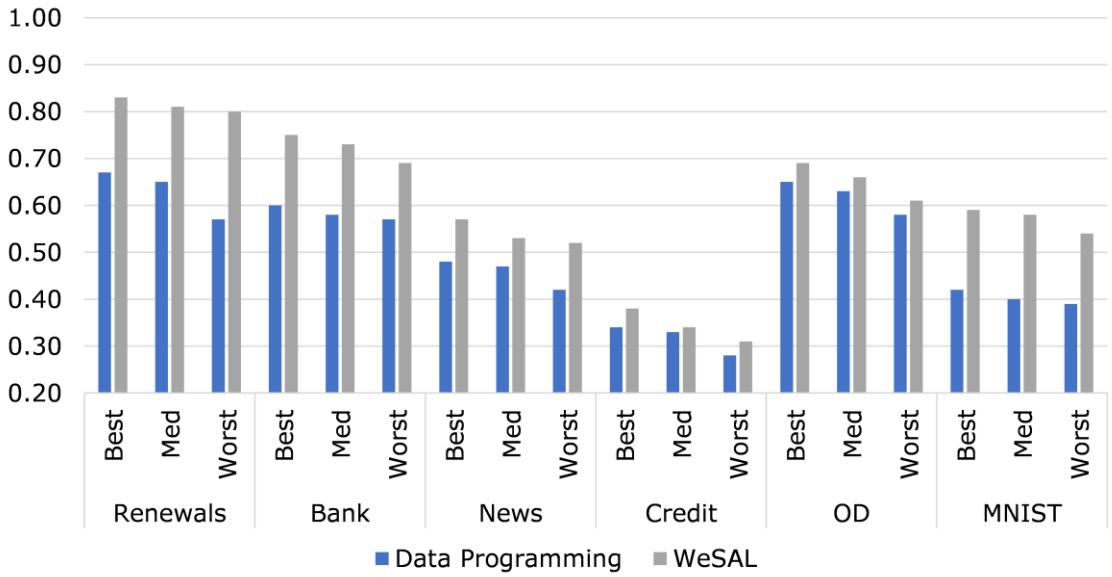


Figure 2.4: Labeling accuracy of DP and WeSAL with different labeling functions

WeSAL tries to address abstaining situations by providing correct labels to improve the coverage. Also, the results show that some LF_{Worst} sets have low accuracy levels close to the accuracy threshold, such as the credit card dataset. As a result, the MCC values of DP and WeSAL decreased by 27% and 14%, respectively, compared to the MCC levels obtained using the original set (Table 2.3). However, WeSAL managed to achieve better performance than DP since it enhances the accuracy of these labeling functions by resolving some of their disagreements.

We also report the labeling accuracy achieved using each set of labeling functions. The results are illustrated in Figure 2.4 and show that WeSAL maintained its superiority of generating more accurate labels than DP in all the problems. Overall, the results depict that reducing the accuracy and the coverage of the labeling functions deteriorate the discriminative model performance. However, the experiments show that WeSAL manages to outperform DP since it injects the domain expertise to resolve the abstaining situations (increase the coverage) and refine the disagreements between the labeling functions (enhance the accuracy).

2.4.5. Threats to Validity

One of the main internal validity threats that may compromise our confidence in the study results is the way the labeling functions were developed. In most of the datasets, one member of the research team (the first author) has applied pattern matching to develop a set of labeling functions. Then, another member of the research team (the second author) has reviewed the labeling functions and evaluated them using a held-out development set. However, to mitigate this threat, among all the developed labeling functions, we have only accommodated high-quality labeling functions (more than 60% accuracy). Also, we conducted a sensitivity analysis in which we experimented with different sets of labeling functions (Section 2.4.4.2). Overall, the experimental results show that the proposed method manages to outperform state-of-the-art techniques with different setups of labeling functions.

2.5. Related work

WeSAL utilizes weak supervision with AL to create large training datasets. Therefore, we surveyed research [3], [19], [20] that employs weak supervision to label datasets. For example, Hickson *et al.* [19] propose an unsupervised clustering method to classify objects using unlabeled data. Another research [3] investigates information retrieval by modeling weak sources as noisy

channels and tries to learn accurate signals. Xu *et al.* [20] design a solution that employs weak labels to learn to segment images semantically. Although all these approaches use weak supervision sources, unlike WeSAL, none of them tried to enhance the accuracy of the resulting labels using domain experience.

Focusing on enhancing the quality of the labels, other research [1], [4], [5], [21], [22] attempt to denoise weak supervision sources. For example, Ratner *et al.* [21] present an end-to-end system for multi-task learning that learns the accuracy of weak sources. Also, Wu *et al.* [22] provide a programming model to convert domain experience to a form of supervision to train knowledge base construction systems. Moreover, Varma *et al.* [5] present a system that creates heuristics automatically and uses generative models to denoise them. Although all these efforts have employed the idea of generative models to denoise the imperfect sources of labels, none of them have investigated the process of refining the input to the generative model using active learning.

On the other hand, there is ample research [23]–[26] that looks into enhancing the scalability of AL. For instance, Tsou *et al.* [23] investigate the annotation cost for AL in real situations and propose a cost-sensitive tree sampling algorithm to reduce the annotation effort. Another recent study [24] applies AL to the social media domain to identify malicious content. Although the results show that the proposed technique achieves respectable classification accuracy, the method is only applicable to shortlisted textual/link-based posts and validated using a set of datasets with a maximum size of 32k records. Addressing the problem of classifying new classes, Coletta *et al.* [25] provide an approach that combines Support Vector Machines with clustering to learn new classes. The approach aims at reducing the annotation cost by optimizing the number of iterations that AL requires. Other research [26] studies the problem of applying AL to large datasets for multi-class classifications tasks and proposes a new query selection criterion to enable hierarchical expansion of candidates. However, in contrast to our approach, the approaches [23], [24], [26] are validated using a group of synthetic and real-world datasets varying in size with a maximum of 100k records. For example, Tsou *et al.* [23] used a set of 12 datasets from the UCI Repository with a maximum size of 32k records. Hence, the applicability of these methods is not guaranteed for large real-world datasets.

Furthermore, several approaches [27], [28] are proposed, which integrate AL with weak supervision. For example, Kang *et al.* [27] explore both AL and weak supervision as ways to use

model assertion to specify constraints on model outputs. Alternatively, Carbonneau *et al.* [28] apply AL to multiple instance classification where data are weakly labeled. Nevertheless, unlike the proposed method, neither of these approaches tries to reduce the labeling cost while improving the scalability of the output labels.

2.6. Conclusions

In this chapter, we present a new method for generating massive labeled data. The proposed method applies weak supervision with active learning to incorporate users while profiting from the scalability of weak supervision. The method starts with collecting noisy labels from high-level inputs. Then, it refines these labels by resolving the conflicts between the inputs using active learning. To evaluate the proposed method, we applied it to a real case within our industrial partner, IBM, to generate labels for a large-scale dataset of more than 1.3 million records along with five real-world classification tasks. The empirical results show that the proposed method outperforms weak supervision by up to 18% in labeling accuracy. The method also achieves better results than active learning while reducing the labeling accuracy by up to 36%.

References

[1] A. J. Ratner, C. M. De Sa, S. Wu, D. Selsam, and C. Ré, "Data Programming: Creating Large Training Sets, Quickly," in *Advances in Neural Information Processing Systems*, 2016, pp. 3567–3575.

[2] Y. Fu, X. Zhu, and B. Li, "A survey on instance selection for active learning," *Knowledge and Information Systems*, vol. 35, no. 2, pp. 249–283, 2013.

[3] H. Zamani and W. B. Croft, "On the theory of weak supervision for information retrieval," in *ACM International Conference on Theory of Information Retrieval*, 2018.

[4] A. Ratner, S. H. Bach, H. Ehrenberg, J. Fries, S. Wu, and C. Ré, "Snorkel: rapid training data creation with weak supervision," *VLDB Endowment*, vol. 11, pp. 269–282, 2017.

[5] P. Varma and C. Ré, "Snuba: automating weak supervision to label training data," *VLDB Endowment*, vol. 12, 2018.

[6] S. H. Bach, B. He, A. Ratner, and C. Ré, "Learning the Structure of Generative Models without Labeled Data," *ArXiv170300854 Cs Stat*, 2017.

[7] G. V. Cormack and M. R. Grossman, "Scalability of Continuous Active Learning for Reliable High-Recall Text Classification," in *ACM International Conference on Information and Knowledge Management*, 2016.

[8] H. Yu, X. Yang, S. Zheng, and C. Sun, "Active Learning from Imbalanced Data: A Solution of Online Weighted Extreme Learning Machine," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 4, 2019.

[9] E.-C. Huang, H.-K. Pao, and Y.-J. Lee, "Big active learning," in *IEEE International Conference on Big Data*, Boston, MA, USA, 2017, pp. 94–101.

[10] K. Wang, D. Zhang, Y. Li, R. Zhang, and L. Lin, "Cost-Effective Active Learning for Deep Image Classification," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 27, no. 12, pp. 2591–2600, 2017.

[11] P. Jain and A. Kapoor, "Active learning for large multi-class problems," in *IEEE Conference on Computer Vision and Pattern Recognition*, Miami, FL, USA, 2009, pp. 762–769.

[12] S. Ertekin, J. Huang, L. Bottou, and L. Giles, "Learning on the border: active learning in imbalanced data classification," in *ACM conference on information and knowledge management*, Lisbon, Portugal, 2007, pp. 127–136.

[13] M. E. Ramirez-Loaiza, M. Sharma, G. Kumar, and M. Bilgic, "Active learning: an empirical study of common baselines," *Data Mining and Knowledge Discovery*, vol. 31, no. 2, pp. 287–313, 2017.

[14] P. Varma, D. Iter, C. De Sa, and C. Ré, "Flipper: A Systematic Approach to Debugging Training Sets," in *Workshop on Human-In-the-Loop Data Analytics*, 2017.

[15] P. Varma, B. He, D. Iter, P. Xu, R. Yu, C. D. Sa, and C. Ré, "Socratic Learning: Augmenting Generative Models to Incorporate Latent Subsets in Training Data," *ArXiv161008123 Cs Stat*, 2017.

[16] M. Liu, W. Buntine, and G. Haffari, "Learning How to Actively Learn: A Deep Imitation Learning Approach," in *Annual Meeting of the Association for Computational Linguistics*, Melbourne, Australia, 2018, pp. 1874–1883.

[17] G. Beatty, E. Kochis, and M. Bloodgood, "The Use of Unlabeled Data Versus Labeled Data for Stopping Active Learning for Text Classification," in *IEEE International Conference on Semantic Computing*, 2019, pp. 287–294.

[18] M. Bloodgood and K. Vijay-Shanker, "A method for stopping active learning based on stabilizing predictions and the need for user-adjustable stopping," in *Conference on Computational Natural Language Learning*, Boulder, Colorado, 2009, pp. 39–47.

[19] S. Hickson, A. Angelova, I. Essa, and R. Sukthankar, "Object category learning and retrieval with weak supervision," *ArXiv Prepr. ArXiv180108985*, 2018.

[20] J. Xu, A. G. Schwing, and R. Urtasun, "Learning to Segment Under Various Forms of Weak Supervision," in *IEEE Conference on Computer Vision and Pattern Recognition*, Boston, MA, USA, 2015.

[21] A. Ratner, B. Hancock, J. Dunnmon, R. Goldman, and C. Ré, "Snorkel MeTaL: Weak Supervision for Multi-Task Learning," in *the Second Workshop on Data Management for End-To-End Machine Learning*, New York, NY, USA, 2018.

[22] S. Wu, L. Hsiao, X. Cheng, B. Hancock, T. Rekatsinas, P. Levis, and C. Ré, "Fonduer: Knowledge Base Construction from Richly Formatted Data," in *International Conference on Management of Data*, Houston, USA, 2018, pp. 1301–1316.

[23] Y.-L. Tsou and H.-T. Lin, "Annotation cost-sensitive active learning by tree sampling," *Machine Learning*, 2019.

[24] S. D. Bhattacharjee, W. J. Tolone, and V. S. Paranjape, "Identifying malicious social media contents using multi-view Context-Aware active learning," *Future Generation Computer Systems*, vol. 100, pp. 365–379, 2019.

[25] L. F. S. Coletta, M. Ponti, E. R. Hruschka, A. Acharya, and J. Ghosh, "Combining clustering and active learning for the detection and learning of new image classes," *Neurocomputing*, vol. 358, pp. 150–165, 2019.

[26] W. Fu, M. Wang, S. Hao, and X. Wu, "Scalable Active Learning by Approximated Error Reduction," in *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, New York, NY, USA, 2018, pp. 1396–1405.

[27] D. Kang, D. Raghavan, P. Bailis, and M. Zaharia, "Model assertions for debugging machine learning," in *NeurIPS MLSys Workshop*, 2018.

[28] M. Carbonneau, E. Granger, and G. Gagnon, "Bag-Level Aggregation for Multiple-Instance Active Learning in Instance Classification Problems," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 5, pp. 1441–1451, May 2019.

Chapter 3 : M-Lean: An End-to-end Development Framework for Predictive Models in B2B Scenarios

3.1. Introduction

Big data is increasingly becoming a major organizational asset for all sizes of industries. The volume of the data generated in industry, from cloud-based systems, management solutions, and so forth, is growing enormously [1]. This exponential growth of data presents new challenges of how to make sense of the data, discover hidden trends in it, and employ this information to improve business operations. The primary objective of using big data in the industry is to maintain cost-effective business processes. By the appropriate interpretation of such big data, businesses can create more efficient risk management systems and derive value in areas such as decision making, product development, and improvement of customer experience. As a result, research focusing on big data solutions is becoming an essential requirement for future industrial applications.

Consequently, for the last few years, there has been a considerable amount of research focusing on big data analytics in the industry. Some research focuses on defining the current challenges of big data [1]–[5]. Other research presents novel solutions that utilize big data in solving business challenges [6]–[8]. Most of these solutions exploit machine learning (ML) techniques to resolve business problems. The massive rise of ML techniques opened a wide range of possibilities in developing predictive models that help in many categories of business problems such as marketing [9], [10], sales [11], customer churn [6], etc. However, most of the solutions presented in the literature are tailored to solve domain-specific problems. The topic of providing a systematic framework for utilizing big industrial data receives minimal attention in the literature. Based on our survey, none of the existing studies has focused on guiding businesses to define possible opportunities for exploiting their data to build predictive models.

Therefore, in this chapter, we propose *M-Lean,* which is a framework to steer businesses to transform their data into actions through building Business-to-Business (B2B) [12] predictive systems. The framework employs the Lean Startup methodology [13] to maximize the business value of the developed systems while eliminating wasteful development practices. To eliminate

uncertainty coupled with the application of ML in industry, M-Lean applies different types of research designs through a sequence of phases. Precisely, since the question of the usefulness of the final system constitutes a significant source of uncertainty, the framework applies an exploratory research phase in the beginning to qualify the business value of the final model based on insights collected from real users and business leaders. Afterward, the framework conducts improving research in subsequent phases to test and maintain the business vision about the final model. Moreover, to sustain an adequate level of model performance, the framework applies various methods for data collections to obtain feedback from different stakeholder groups throughout the development and deployment phases. The primary contributions of this research can be summarized as follows:

- An end-to-end development framework is proposed to develop, evaluate, and deploy predictive products in business domains. It is argued that this is the first such end-to-end life-cycle process for data-intensive application development for B2B scenarios where a rich cross-section of stakeholders is actively engaged in the process. It is also argued that such engagement is essential if we hope to realize successful product lines.
- With the help of our industrial partner, IBM, we have applied our framework to a case study to build a B2B predictive product that predicts software license cancellations. That is, we undertake and report on an initial evaluation of the feasibility of the approach.

This chapter is structured as follows: Section 3.2 presents the related work. The study scope and the research questions are represented in Section 3.3. While Section 3.4 discusses the research methodology, Section 3.5 introduces the overall architecture of the M-Lean framework. The application of the M-Lean framework to a case study in the IT industry is presented in Section 3.6. Section 3.7 discusses the lessons learned from the case study, reflects the cost of the implementation, and presents the threats to validity of the case study, while Section 3.8 concludes the chapter.

3.2. Related Work

Giving the essential role of ML systems in the business domain, there is a need to address the challenges that ML components bring into software systems. The primary focus of this research is to present a systematic structure for developing, evaluating, and deploying predictive systems in

B2B scenarios. Therefore, we survey previous work that looks at the intersection between ML and software engineering. We classify the related work into three categories. The first category considers the application of ML in software engineering [14]–[17]. The applications include utilizing ML techniques in predicting software fault and defects [14], [16] recommending process model [17], and estimating development effort [15]. Although this category aims at using ML techniques to optimize the process of creating software systems, none of these efforts looks at the challenges of using ML components as a part of the software systems.

As for the second category, since we present predictive models as a new class of requirements engineering problems, we survey existing work that combines the domains of ML and requirements engineering. We found that, over the last decade, many researchers have used ML models in analyzing the requirements for different software systems [18]–[20]. Research [18] employs supervised learning approaches to classify requirements as functional and non-functional requirements. Perini *et al.* [19] use ML to prioritize software requirements by combining the stakeholders' preferences with the requirements ordering. Also, Avesani *et al.* [20] present an automated ranking system for managing potential risks. However, in contrast to our work, none of the existing studies have focused on eliciting the requirements of ML applications themselves.

The third category aims at addressing the challenges in developing ML systems [21]–[24]. Previous work [21], [22] states that distributed systems are required for an end-to-end ML pipeline. Meng *et al.* [22] propose an open-source distributed ML library for scalable implementation of standard ML techniques. Other research [23] presented a system to optimize end-to-end ML systems, while Vartak *et al.* [24] produced a system to manage ML models. Although these studies focus on offering solutions to ease the process of creating ML models, none of these solutions tried to consider the perspective of business management and end-users. These efforts did not address the challenges of identifying opportunities to improve business processes using ML. Moreover, most of these solutions did not consider the requirements enforced by the business domain.

In contrast to the previous work, we offer an integrated ML framework that incorporates a broader range of insights, as we think that developing, evaluating, and maintaining ML products should not only consider the perspective of the data science team. Instead, the real world's input must be accommodated as well. Perhaps, the closest works we can find are life-cycle descriptions for ML

development, which describe the process from the data scientists' viewpoint[2]. However, these life-cycles are focused on technical aspects and ignore the essential roles of business leaders, the marketplace, end-users, and other stakeholders required to produce a holistic product line rather than just a ML algorithm. Hence, it is believed that this work is unique in its scope in providing an end-to-end life-cycle process for data-intensive, commonly ML-based, B2B applications.

3.3. Study Scope

In the last few years, a considerable amount of research has taken place to apply ML techniques to industrial problems [6]–[12]. However, most of this work focuses on solving domain-specific problems. Therefore, it is hard to generalize these solutions to a broader range of applications. In this research, we start by asking some questions that formed the basis of our study:

RQ1: What extent of research has been done to create an end-to-end framework for building predictive models in B2B scenarios?

RQ2: What are the design decisions required to create an end-to-end framework for building predictive models in B2B scenarios?

RQ3: What actions must be taken to apply and evaluate the effectiveness of such frameworks in the industry?

To be able to answer the research questions, we limit our literature survey to the research that has been undertaken during the last five years. Regarding the firmographics variables [25], we focus on international organizations that have more than 10,000 employees. Each organization has its data science team, which is responsible for managing and analyzing the business datasets. Consequently, technical experience in predictive analytics and ML is expected within this team, since they work to apply ML to provide business solutions.

Additionally, this research primarily focuses on predictive systems in B2B [12] situations where both the seller and the buyer are organizations. Data in B2B scenarios is usually more complex than the data in Business-to-Consumer situations. In B2B environments, companies build long-term relationships with their customers, which results in data collected from diverse sources such

[2] https://docs.microsoft.com/en-us/azure/machine-learning/team-data-science-process/lifecycle

as historical records and social media analytics. Therefore, models built to process such data must be able to handle this level of data complexity. Also, predictions generated for B2B scenarios are often followed by business decisions. Therefore, a level of interpretability [12] is required to gain more confidence about the following business decision.

3.4. Research Methodology

In this section, we discuss the undertaken steps of our research methodology. Based on our literature survey and interaction with industry, we perceive that there are many challenges for applying ML in the business domain [5], [9], [26]. As a result, we employed an action research approach [27] intending to resolve some of these challenges. This research involves a longitudinal case study applied, with the help of our industrial partner, over more than nine months. A group of two researchers (the first two authors) worked to coordinate the application of the proposed framework. In our approach settings, the research team was treated as a part of one stakeholder group (i.e., the data science team) and not seen as researchers by the other stakeholder groups. As a result, the research team was able to iteratively integrate their theories within the practice and continuously validate their hypotheses based on the experience gained throughout the case study.

The rationale behind using the action research approach originated from the correlation between the general elements of the action research approach and the primary principles of the proposed framework in two main points. First, action research shapes a collaborative process between researchers and different stakeholders in a given context. Second, it enforces a process of critical inquiry and reflective learning as a part of the research.

Therefore, as a start, the two researchers surveyed previous efforts in the literature to establish a framework of reference for the research and to answer RQ1 (Section 3.2). Then, we developed the research hypotheses for our study that aims at answering the research questions stated in Section 3.3.

As mentioned in Section 3.2, communication with different stakeholder groups is an essential factor that influences successful development of any ML model. Hence, we formulate our first hypothesis that seeks to answer RQ2 as follows:

H1: Effective interaction with business leaders and end-users is positively connected with successful product lines. That is, the more interaction, the more successful the predictive model.

Also, since a significant gap is noticed in research that focuses on analyzing the requirements of predictive models in the industry, we formulate our second hypothesis that seeks to answer RQ3 as follows:

H2: Addressing the business requirements for the predictive model is positively connected with a satisfactory level of performance in production.

Subsequently, we proceed with the research by designing the framework (Section 3.5). The framework creates a continuous interaction with the stakeholders in which the business domain is recurrently investigated for possible opportunities for applying ML solutions. To facilitate the application of the proposed framework, the framework was designed as a sequence of phases. Along with the design of each phase, the framework outlines the application of each phase by defining the phase objective, the research questions, and the recommended methods for data collection. To apply the framework, the framework users can apply the suggested practices in each phase to collect data, answer the research questions, and progress to the subsequent phase. Thus, the framework can be easily integrated into the business workflow. The final output of the framework is a B2B predictive model that has been trained using business data and has proven its effectiveness in building user trust. To validate our hypotheses, we apply the framework to a real case study (Section 3.6), in which the two researchers followed the framework design to build a system to predict software license cancellations.

Regarding data collection, several data collection methods were employed through the application of the M-Lean framework, including interviews (Section 3.6.2), participant observation (Section 3.6.3), group meetings (Section 3.6.4), and analysis of historical data records (Section 3.6.5). The diversity of the data method collection helped with data triangulation [28] and eliminated the risk of systematic biases. The rationale for choosing each data collection method is further elaborated in the following section, along with the description of each phase of the framework.

3.5. Proposed Framework Design

This section describes the overall structure of the M-Lean framework. A high-level component overview of the framework is illustrated in Figure 3.1. The figure highlights the components discussed in the following subsections. Section 3.5.1 discusses the first component, *suggesting ideas and data discovery*. While Section 3.5.2 encapsulates *data preparation*, *model development*,

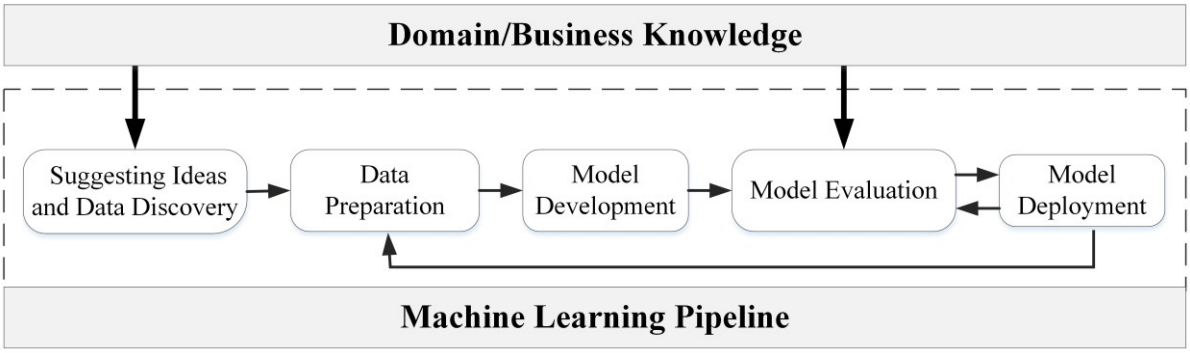


Figure 3.1: High-level component overview of the M-Lean framework

and *model evaluation* components in one phase named the development phase. The *model deployment* component is discussed in Section 3.5.3. The figure shows the standard functionalities in the ML pipeline with their interactions with the business domain. Although in the framework, the business domain supervises the complete pipeline, it interacts with the pipeline in two points. The first point is the *suggesting ideas and data discovery* component. Since the framework is designed to utilize business data in developing B2B predictive systems, data coming from the business domain is considered the primary input to the framework. The second point is the *model evaluation* component, as the framework creates a feedback loop from the business domain to the ML pipeline.

The framework was designed according to the Lean Startup methodology [13]. The Lean Startup is an approach that aims to shorten the development process of products and startups. It follows an iterative process of multiple product releases, hypothesis-driven experimentation, and validated learning. Similarly, M-Lean adopts the same methodology while considering the challenges in the ML pipeline. Table 3.1 concludes the main points in the framework and maps them to the principles of the Lean Startup methodology.

Table 3.1: Proposed framework vs. Lean startup approach

Main Points	Lean Startup	M-Lean Framework
Main Motive	Startups begin with an idea for a product; entrepreneurs think it is fit for the market. However, after development, they fail to reach their customers because they never spoke to a sample of the customers before.	In ML, data scientists build a model, get good results against training data. However, after deployment, the model performance shows severe degradation.

Work Around Uncertainty	Lean Startup methodology eliminates uncertainty by conducting iterative experiments with real customers, so the management can continuously check if the market window is still valid. Therefore, Lean Startup methodology can help organizations to test their vision iteratively, and eliminate the uncertainty through the development phase.	Uncertainty in ML originates from different resources: • Model usefulness: M-Lean eliminates this uncertainty by initiating discussion circles from the very beginning (Section 3.5.1). • Model performance: M-Lean eliminates a part of this uncertainty by providing business data as an input to the development phase (Section 3.5.2).
Eliminate Inefficient Practices	At each development cycle, Lean Startup experiments to validate business hypotheses. So, the management can decide if the product is ready for the market. In the meantime, the experiments help to test the product with real customers and increase consumer awareness.	The framework accommodates user culture through the development phase. Hence, the development team can ensure that the model has prospective users. Moreover, the framework creates a feedback loop from the business domain to the development cycle at each development iteration (Section 3.5.2).
MVP/ MVM Development	A central module of the Lean Startup methodology is the build-measure-learn loop. In each development cycle, the business develops (build) a minimum viable product (MVP) to begin the experiments (measure). Then, the startup starts working on adding more improvements that depend on the learning process (learn) obtained from the experiments.	The build-measure-learn loop is the core of the development phase. The development team starts the development (build) once the system hypotheses are defined. Each development iteration aims at building a minimum viable model (MVM). Then, the model is evaluated to validate the hypotheses (measure). Based on the evaluation results (learn), the development team can decide on the next steps.
Validated Learning	Validated learning demonstrates the startup progress. Once entrepreneurs adopt the concept of validated learning, they can shorten the development process significantly.	In the framework, validated learning follows the experimental results. If the decision is to Pivot, the hypotheses need to be readjusted to reflect the learning. If the decision is to Persevere, the system is deployed while planning for future improvements.

Continuous Maintenance vs. Continuous Deployment	At some point, the startup stabilizes with successful launches and steady growth. However, the management maintains its success by continuously re-evaluating their vision and running experiments to validate new hypotheses regarding future improvements.	In ML, the model performance degrades once it is put in production. Therefore, the development team must keep monitoring the model performance forever. Once the development team notices a problem in production, the team can step in and fix it.

3.5.1. Getting More from Business Data: Ideas Suggestions and Data Discovery

The framework starts with a preliminary phase in which possible opportunities for domain improvements are recognized. In the framework, this is accomplished by initiating discussion circles with individuals in different roles in the organization. Table 3.2 shows the overall outline of the framework phases. For each phase, the table shows the main objective, the research questions, and the recommended method for data collection. The research questions that this phase aims at answering are listed in the table. Answering the first two research questions (RQ1.1 and RQ1.2) is about matching the right dataset with the right opportunity. Therefore, insights from business executives and data scientists are required to formulate a unified answer for both questions. Also, after defining this match, the impact of the proposed solution on the business must

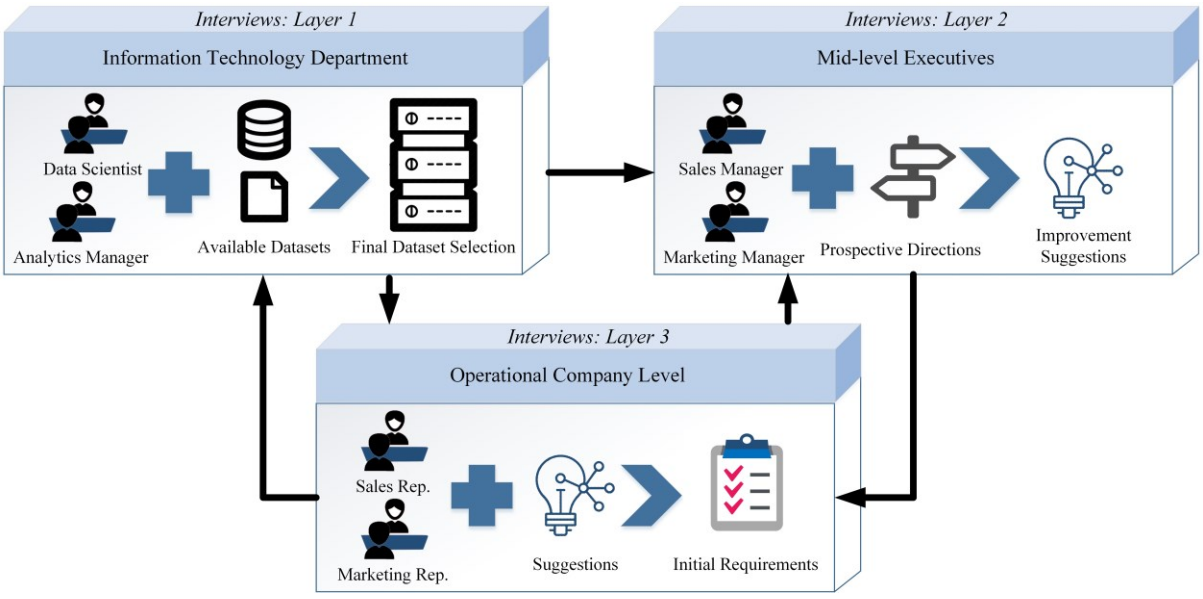


Figure 3.2: Interviews structure in Phase 1

be evaluated (RQ1.3). Therefore, business leaders should assess the derived business impact to answer the third research question.

Thus, we recommend adopting in-depth semi-structured interviews [29] to collect data in this phase. Semi-structured interviews are guided by topics containing primary questions that must be used in the same way through all interviews. However, this structure allows new ideas to be discussed during the interviews. An important reason to recommend semi-structured interviews is that, in this phase, the framework users already have some understanding of what is happening

Table 3.2: Outlines of the framework phases

Phase	Objective	Research Questions	Methods for Data Collection
Phase 1	Exploratory	**RQ1.1.** What business problem can be solved using ML? **RQ1.2.** Does the business have enough good quality data to apply ML techniques to solve the defined problem? **RQ1.3.** Should this predictive model be built? If yes, what are the initial business requirements for this model?	Iterative process of in-depth semi-structured interviews
Phase 2	Improving	**RQ2.1.** What hypotheses can be derived from the system requirements? **RQ2.2.** What data preparation activities the development team needs to perform to prepare the dataset? **RQ2.3.** What are the primary design decisions to build the MVM? **RQ2.4.** How can user feedback be iteratively incorporated in the model development-evaluation loop?	Indirect methods including job shadowing, observation, and think-aloud protocol
Phase 3	Improving	**RQ3.1.** What are the thresholds of quality metrics that define the need for retraining the model? **RQ3.2.** Can automation be adapted to maintain the feedback loop from the real world to the model development team?	Independent analysis using recorded user feedback for model's predictions

within the organization. However, they could use open-ended questions in semi-supervised interviews to obtain a deeper understanding and encourage respondents to share their opinions.

The process of data collection is structured in an iterative layered fashion and summarized in Figure 3.2. As the figure shows, interviews in Layer 1 aim at obtaining qualitative data about the datasets stored by the organization. The respondent sample in Layer 1 is a group of data scientists and data analysts. We recommend using purposive sampling to choose the respondents sample [30]. More specifically, expert sampling [30] can help in acquiring the knowledge established in the form of expertise. Based on the findings of the interviews in Layer 1, interviews in Layer 2 can be structured to capture the main challenges the business faces (RQ1.1). The respondent sample in this layer is formed from the middle management of the organization. This layer of interviews could help in refining a list of suggestions for proposed solutions (RQ1.2). These solutions can utilize the datasets (from the interviews in Layer 1) and assist in resolving the challenges in the work process.

After forming a clear understanding of the possible solutions, interviews in Layer 3 can be used to give a qualitative answer to RQ1.3. The respondents in this layer are a sample of the potential end-users of the final predictive model. They need to qualify the business value derived from the prospective model and define the essential requirements to maximize its business value [31].

Although requirements elicitation is a crucial task of requirements analysis, we think that using interviews can be beneficial for many reasons. First, there is a high probability that the respondents in this layer will be the end-users of the final model. Hence, the interviews can help to estimate the business value derived from the model. Second, many studies [32], [33] highlight that, among the existing methods for requirements elicitation, interviews are the most frequently used for determining requirements. Requirements elicited, at this point, must identify the business preferences regarding the following points:

- Which quality metrics affect the model's business value the most? Usually, there is a tradeoff between quality metrics (i.e., a model that achieves a perfect precision value usually has low recall). Therefore, the initial requirements must define user preferences for performance measures used to evaluate the model.
- Which data sources can be used to train the model? On the one hand, input data needs to imitate the same data the domain experts use to come to a decision. On the other hand, the diversity

of the data sources can affect the complexity of the data and the choice of the underlying algorithm. Therefore, the data science team must confirm the availability of the data and its conformity with the rest of the requirements.

As the answers for the three research questions (RQ1.1, RQ1.2, and RQ1.3) must converge to describe one unified system, the proposed process of data collection presented here is iterative. Thus, after eliciting the requirements for a predictive model, the framework users can go back and talk to the data scientists to validate the compatibility of the dataset with these requirements. The framework users can only exit this phase when they acquire a consistent set of answers for the research questions.

3.5.2. Developing the Solution: Data Preparation, Model Development, and Evaluation

This section describes the components of data preparation, model development, and model evaluation. A detailed illustration of the phase is presented in Figure 3.3.

3.5.2.1. Data Preparation

As shown in Figure 3.3, the data preparation phase has two main goals. As for the first goal, it aims at collecting data from the sources identified in the previous phase (Section 3.5.1) and transforming it into a form that can be used to train a ML model. As mentioned before, data in B2B scenarios are more complex. Hence, data scientists may need to collect data from different unknown sources (e.g., news feeds and social media contents). Therefore, the figure depicts that data sources are not only limited to historical records but also additional sources can be identified throughout the framework phases. For example, even though some of these sources may be recognized during the first phase, other sources may not be revealed until the evaluation phase as a part of user feedback.

Moreover, during the process of data transformation, data scientists perform a set of complex activities. Examples of such activities can be summarized as follows:

- *Feature Engineering*: although the term "*feature engineering*" may sound related to product line engineering, in ML, the term refers to the process of selecting (and engineering) specific attributes (features) from the input data. It is one of the key activities of data preparation in

ML. Data scientists, using the domain knowledge obtained during the last phase, can extract features from the data that are useful for the model to learn.

- *Ground Truth Generation*: in supervised ML techniques, the model learns from labeled examples. Thus, data scientists need to gather labeled data to train and test the model. Both the size and accuracy of the training data affect the final performance of the model.
- *Deciding on missing values and outliers*: while missing values can compromise the model performance, outliers can affect the model output [34]. Therefore, data scientists need to decide on which method should be adopted when dealing with missing data and outliers.
- *Data anonymization*: even though our framework aims at creating predictive systems that will be deployed internally within the organization, the process of data anonymization is an essential step before conducting any analytics.

The second goal of the data preparation phase is to formulate a set of hypotheses to define the response of the end-users towards the model's anticipated behaviors [13]. At this preliminary point, the hypotheses should describe the business value expected from this system, and the performance level that must be obtained before deployment. The hypotheses help the data science team to start building the model as soon as possible. For example, to implement a recommender system, instead of spending time refining the requirements list, the development team can formulate a set of hypotheses from an initial requirements list and start experimenting. The requirements list can be further updated when the users evaluate the model. Adopting the approach of build-measure-learn [13] in this early phase enables the end-users to trust the model from the very beginning.

There are at least two main hypotheses that can be formulated at this point. These hypotheses are the business value hypothesis and the performance criteria hypothesis.

- The business value hypothesis tests whether a model can deliver business value to the enterprise or not.

- The performance criteria hypothesis tests if the developed model can meet the performance criteria specified in the requirements list.

As for the performance hypothesis, it is important to note that evaluating ML models differs from evaluating the quality of software products. While evaluating software products considers different

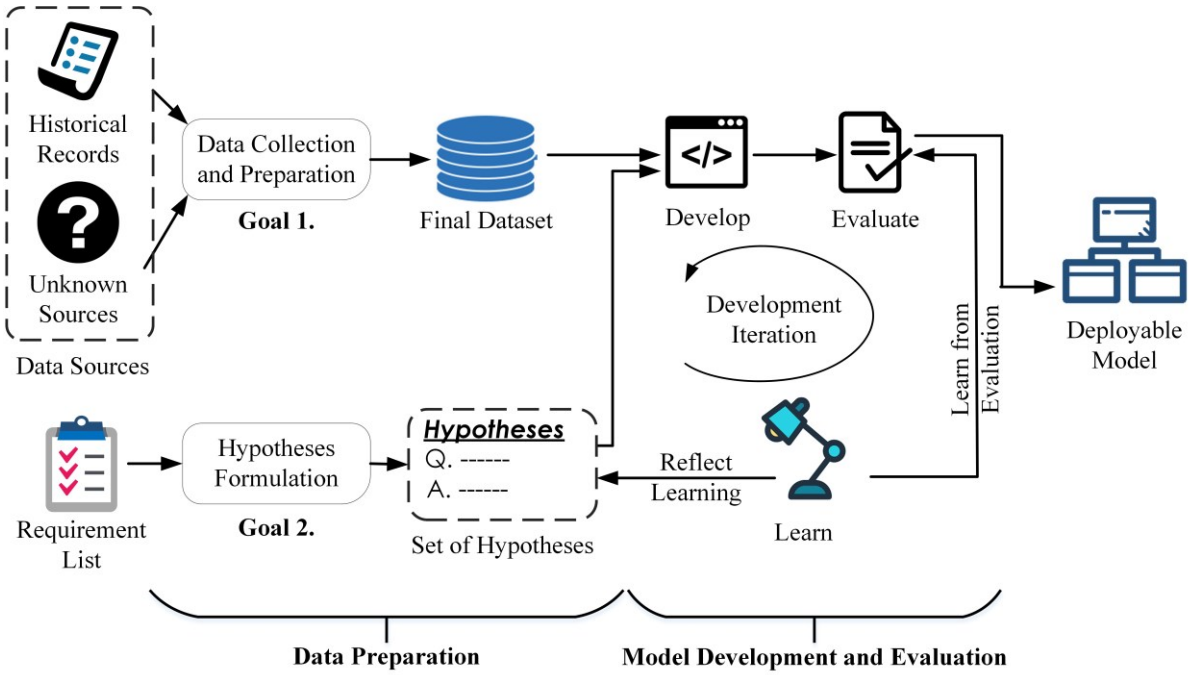


Figure 3.3: Data preparation, model development, and evaluation

characteristics (e.g., ISO 25010) such as efficiency and usefulness [35], evaluating ML model primarily depends on statistical evaluation metrics such as precision and the classification accuracy [36].

3.5.2.2. Model Development

There are two inputs to the development component: the final dataset and the set of hypotheses. The development phase aims at producing an initial model that validates the set of hypotheses. The development phase tries to shorten the development time by applying a set of development iterations. Every iteration involves an experiment to assert the validity of the hypotheses. The first iteration starts by building a minimum viable model (MVM). Then the model is evaluated to validate the hypotheses. The evaluation procedure needs to quantitatively measure the model performance and collect qualitative user feedback as well. A sample of end-users, ideally actual end-users, but end-user proxies are a viable alternative, must test the prototype as a part of the evaluation process. As in the first phase, purposive sampling [30] can be used to select this sample.

The evaluation results will determine if more development iterations are needed. Terminating the development phase and deploying the model is considered as a business decision that is made after

analyzing both the data collected from user feedback and the quantitative data obtained from the model's statistical evaluation. Since, in the business domain, end-users will derive business actions from the model's output, users may have specific requirements that must be satisfied in the model's final predictions. Therefore, the development team can use the insights collected during each iteration to shape the next development cycle in a way that increases user trust and acceptance.

3.5.2.3. Model Evaluation

After receiving the evaluation results, the development team and business leaders can examine these quantitative and qualitative results to decide if they should pivot (initiate more development iterations) or persevere (terminate the development phase). A pivot is a structured set of corrections to test another fundamental hypothesis. In this case, the set of hypotheses is changed, and a new development iteration is initiated. As a result, the business value hypothesis can be iteratively validated. Alternatively, persevering means that the current set of hypotheses is initially validated, which means that the model is ready for the phase of *continuous deployment*, in which the model is deployed and continuously improved and evaluated to maintain the achieved business value.

The development phase adopts an improvement approach [37]. A list of research questions of the phase, along with the overall outline, is summarized in Table 3.2. Since the phase accommodates user feedback to evaluate the model at each (development) iteration, the process of collecting user feedback can be challenging for many reasons. First, collecting qualitative data requires qualitative analysis. Second, analyzing user feedback must consider user culture. For instance, if users think that the automation provided by the predictive system can threaten their jobs, they might not be willing to provide constructive feedback. Thus, the framework users may need to collect data about the internal work process that forms the end-user culture.

The process of refining the hypotheses set while considering the user culture is modeled in Figure 3.4. The figure shows that the framework can employ two techniques for cultivating user culture. As for the first technique, the framework users can conduct informal interaction, adopting ethnographic research to derive the causes of user behaviors and consider these causes when analyzing user feedback. Moreover, the development team can influence user culture and assure the users that using ML can never replace the need for human creativity. On the way to accomplish such a goal is by demonstrating interactive ML approaches [38] as an optional design path. The

framework users may need to convince the users that they can be involved in evaluating and modifying the model.

3.5.3. Starting it all over again: Model Deployment

The deployment phase is designed to preserve model performance achieved in the development phase. One crucial factor that affects model performance is data freshness. Since data in the business domain is affected by many factors such as competitor's promotions (external) or changes in a business policy (internal), the model performance can drastically vary once put in production [39]. Therefore, the framework requires the model developer to keep evaluating the model with the latest data on a regular basis. The framework treats this re-evaluation step as another set of experiments to validate the system hypotheses. Figure 3.5 shows an overview of the deployment phase, with its interaction with the development phase. The figure shows that the development team needs to continue collecting business data to re-evaluate the model. The results obtained from the re-evaluation are then used to validate the system hypotheses. If the hypotheses were validated, this means that the users still trust the model. Alternatively, if the evaluation results did not validate the performance hypotheses, this may mean that the performance degradation is caused by a drastic

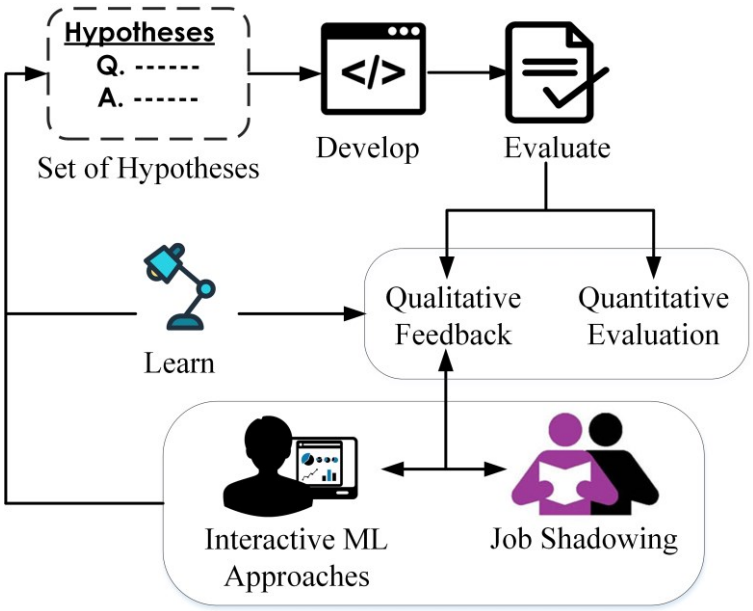


Figure 3.4: Cooperating user culture in model evaluation

change in the input data. In this case, the development team can investigate this change and reflect this learning on the hypotheses set.

The deployment phase aims at achieving an improving objective with the outlines presented in Table 3.2. Moreover, by examining Figure 3.5, one can spot a repeatable loop of collecting data, retraining, evaluation, and deploying. As the loop does not require much designing effort, we recommend that businesses consider automation by creating a validation platform. The framework recommends designing the validation platform to achieve three goals:

- automatically collection of user feedback;
- collecting quantitative data about the model's performance; and
- performing statistical estimates of the status of the real world.

By collecting user feedback, the development team can analyze collected data without the need for conducting direct or indirect collection methods. As a result, the cost of data collection is decreased. Also, since the model must be evaluated using new data at frequent intervals, the validation platform can employ "visualization assistants" to enable business leaders to understand the trends in the evaluation results.

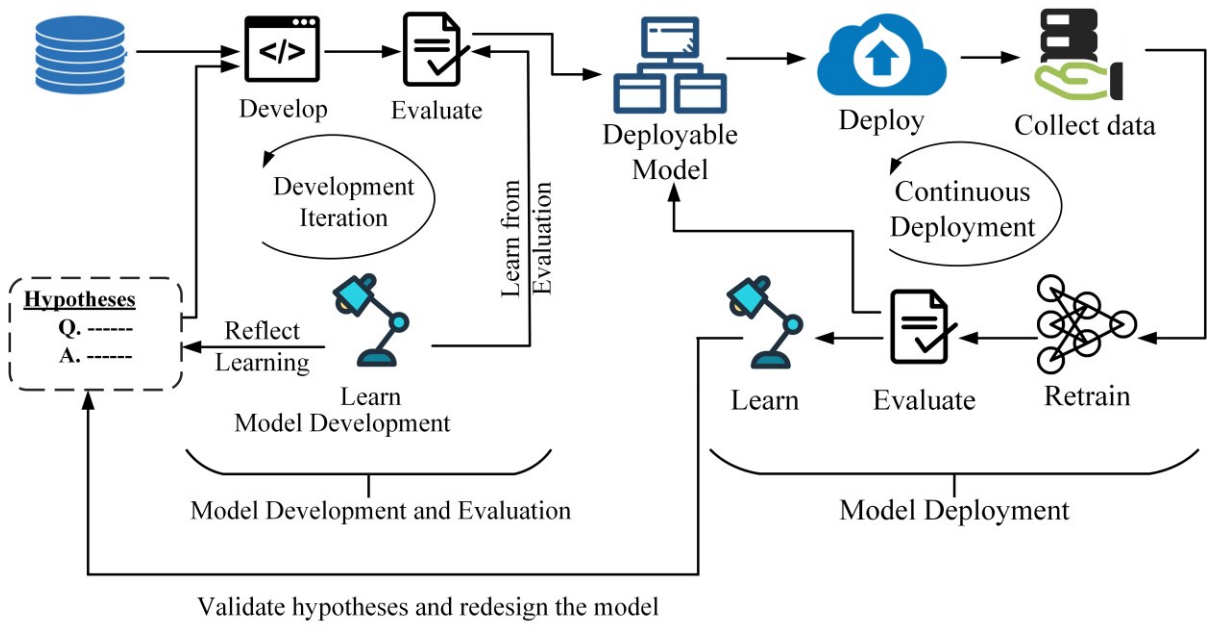


Figure 3.5: Model development and model deployment phases

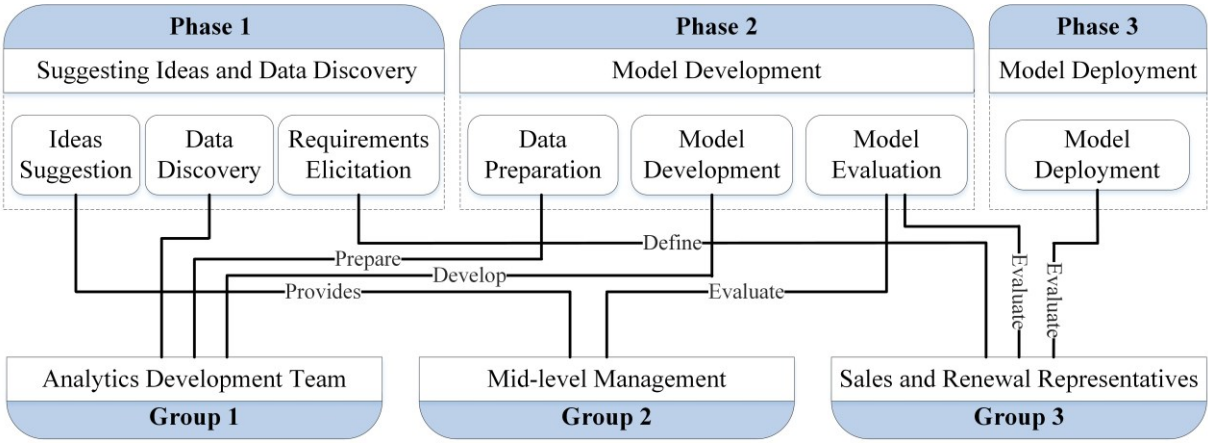


Figure 3.6: Stakeholder groups and their interactions

Regarding the third goal, it is essential, when accommodating changes in the real world, to consider the changes for which the model is responsible. For example, after deploying a system that recommends clients for marketing campaigns, the collected data will reflect the policies enforced by the system. Therefore, analyzing such data will not consider some blocked changes in the world, such as assessing the success of approaching unrecommended clients. Thus, for evaluation, the development team needs an approximation of the distribution of events that would exist in the absence of their intervention (the model). Hence, we recommend designing the validation platform to perform this statistical estimate [40]. One attempt to do such estimates is to deliberately let through some of the blocked events. For example, if the model is recommending clients with a confidence score P_i(success) larger than a threshold (e.g., P_i(success)>0.6), the validation platform can recommend some clients by applying propensity function [40] to choose a set of blocked clients whose confidence score is close to the threshold (e.g., 0.4<P_i(success)<0.6). Since the model is more uncertain around the threshold value, monitoring the outcome of these recommendations can help to refine the model policy.

To give an example of how to calculate precision and recall in such situations, let us assume that the model examined 1M clients and initially recommended 400,000 clients. Then, the platform recommends a random set of 30,000 blocked clients to the users. After examining the results, the end-users reported that only 6,000 of this set were true positives (the clients positively responded to the campaign), while the rest (24,000) were false positives (the clients rejected the campaign). At this point, the platform should give each of these allowed events a weight value (W_i), as each

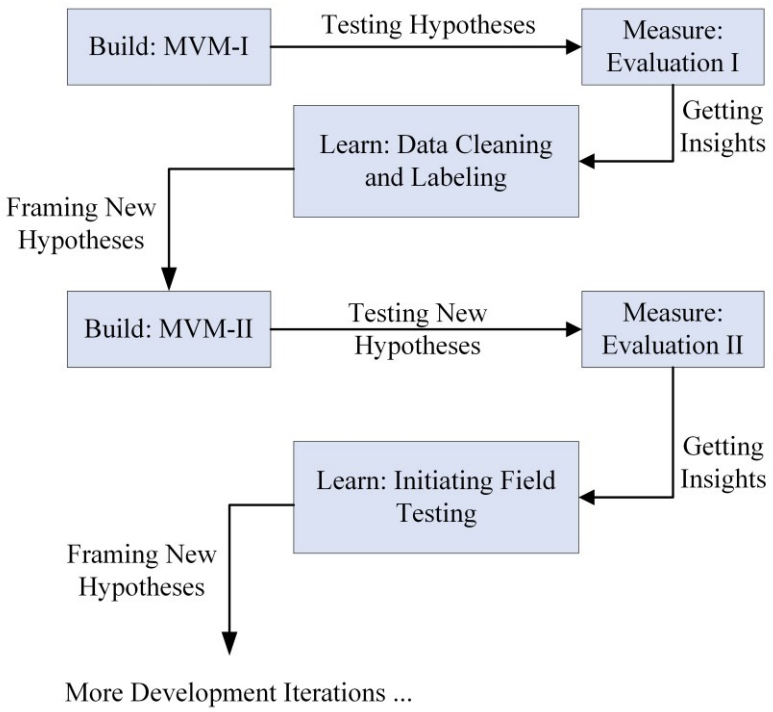


Figure 3.7: Develop-Evaluate-Learn cycles

event is a representative of the pool of the blocked clients. Therefore, the platform can weigh each allowed sample by $W_i = \frac{1}{P_i}$ which represents a geometric series. Alternatively, for the original 400,000 recommended clients, the results showed that only 4,400 were false positives.

Hence, by analyzing the outcome of the allowed events, the model performance can be estimated as follows:

- Negative replies caught by the platform (TN) = $24{,}000 * \sum_1^{24{,}000}(\frac{1}{P_i})$
- Precision of the blocking policy = $(24{,}000 * \sum_1^{24{,}000}(\frac{1}{P_i}))/600{,}000$
- Recall of the blocking policy = $(24{,}000 * \sum_1^{24{,}000}(\frac{1}{P_i}))/(24{,}000 * \sum_1^{24{,}000}(\frac{1}{P_i}) + 4{,}400)$

Moreover, the development team can repeat the experiments with different values for thresholds to test many alternative policies.

3.6. Case Study: License Cancellation Prediction

This section describes the application of the M-Lean framework to a real-world case study. At each phase, we followed the proposed outlines to answer the research questions.

3.6.1 Case Study Settings

In this subsection, we firstly discuss the stakeholder profiles and their interactions during each phase. We then present the settings of the final model targeted from the framework.

3.6.1.1. Stakeholder Profiles and Interactions

IBM is a multinational IT company that provides a range of products and services. We work closely with the Analytics Development team. The team consists of a machine learning architect who leads a team of four data scientists and three data engineers. The team's skills include machine learning, artificial intelligence, and data visualization. The team works to provide data analytics services and build revenue-impacting ML models to different business units in IBM.

During the case study, the research team interacted with three stakeholder groups, namely, the analytics development team, a group of mid-level managers, and a group of sales and subscription (S&S) representatives. Figure 3.6 shows the three groups and their interactions during each phase. As the figure depicts, the analytics development team's main interactions are in the *data discovery* component, *data preparation* component, and *model development component*. In the data discovery component, the analytics development team provided information about the organization's stored datasets. While in the development phase, the analytics development team, with their technical experience, directed the data preparation and model development components. As for the *idea suggestion* component, the mid-level management team provided information about the business challenges which ML models can resolve. The management team also played an essential role in the *model evaluation* component as they supervised the model development phase and decided when to terminate the development phase and deploy the model. The third group of S&S representatives interacted in the *requirements elicitation* component as they helped the research team to define an initial list of goals (requirements). They also evaluated the model during the *model development* and the *model deployment* phases.

Table 3.3: The iterative process of interviews in Phase 1

Layers	Number of interviewees	Number of interviews	Interview Duration (min.)
Layer 1	4	6	50
Layer 2	3	1	30
Layer 3	8	1	30

3.6.1.2. The Final Model Settings

With the analytics team's assistance, we applied the framework to an end-to-end situation where we started by identifying possible opportunities for ML, identifying data sources, developing a predictive model for license cancellations, evaluating its outcome, and initiating field tests as a preliminary phase of deployment. Figure 3.7 shows an overview of the develop-evaluate-learn cycles followed in the case study. The figure only shows the executed cycles up to the point this article was written. The final MVM is currently undergoing its first iteration of field testing; hence, unfortunately, no further data could be collected. This MVM was created after running two development iterations and was trained and evaluated using five years of renewal transactions of over 1.3 million purchase orders and 11 attributes. The dataset is commercially sensitive and describes the customer license agreements. Therefore, we only have limited access to a completely anonymized version of the data. These agreements include conditions on using the features of the purchased software. When the license is about to expire, the customer needs to either renew it by placing a purchase order or cancel it.

3.6.2. Phase 1: Suggesting Ideas and Data Discovery

To start our case study, we began with the phase of ideas suggestions and data discovery (Section 3.5.1). To collect the data, two researchers followed the iterative process of in-depth semi-structured interviews. Table 3.3 gives an overview of the application of this process. The table shows the number of interviewees in each layer, the number of interviews conducted with each interviewee, and the duration of each interview in minutes.

All the interviews in this phase were structured according to the funnel model [41]. A sample of the interviews scripts is presented in Appendix A. The interviews were structured to start with

open questions and progress to specific ones. During the interviews, all sessions were recorded in an audio format and then transcribed by the two researchers to capture all the details. Anonymized transcripts were then reviewed by a third researcher (the third author) to validate the analysis results. Although transcribing the interviews was time-consuming and can be avoided, especially in the first two layers, the research team found it useful, in this setting, for many reasons. Firstly, it facilitated information sharing among the research team. Second, it helped the research team to agree about data interpretation. Thirdly, it formed a source of reference for the research team in the follow-up interviews in later phases. Lastly, transcripts were useful in interviews in Layer 3, as the research team found coding necessary to define and prioritize user requirements. The process of interviews was conducted as follows:

- In the interviews of layer 1, we conducted weekly group interviews with IBM Analytics Development team for six months. The respondent sample included the team leader, two data scientists, and one data engineer. During these interviews, the research team collected

Table 3.4: Available datasets

Dataset	Description
Renewal Purchase Transactions	The dataset contains anonymous information about the customers' entitlements data. Entitlement information includes the purchase date, license type, the expiration date, and information about the purchased product.
Products Download History	The dataset contains information about customers downloads log for each product such as download date and exact time, the number of downloads, and the software license.
Problem Management Reports	The dataset includes support tickets submitted by the customers. It has information describing the ticket's lifetime and the conversations between customers and the support team.
Products Allocations and Deallocations History	The dataset contains information related to the products deallocation to different locations (sites). Usually, in B2B scenarios, the customer has multiple business locations. Thus, since the license is given to a specific site, the customer can choose to move some of his licensed products to another location.
Products Evolution	The dataset contains all the information related to product evolution. This is when the product lifecycle comes to an end, and a new version is available.

information about the datasets managed by the analytics team. All the datasets that the team explored were anonymized. The list of these datasets and a short description for each dataset are summarized in Table 3.4.

- In layer 2, the research team had a series of interviews with three individuals from the management. The management sample contained one program director of IT and analytics, one program leader of worldwide S&S business, and one software S&S specialist. During these interviews, information about the renewal process was collected and analyzed at IBM Canada Head Office in Toronto. At this point, one opportunity arose, which is to build a predictive model to predict renewal risks. Thus, the answer to *RQ1.1* was framed in terms of a top-level function [42] as: *"a system is needed to predict and report renewal risks to the sales team beforehand, so the sales team can proactively try to mitigate these risks"*.
- In layer 3, the research team utilized the data collected from layer 2 in their domain analysis and started the requirements elicitation process (*RQ1.3*). The team conducted a set of interviews with eight S&S representatives. During the interviews, the team elicited the initial requirements in the form of goals using the KAOS model [43].
- The initial identification of the functional and non-functional requirements is presented as a generic goal pattern in Figure 3.8. The figure shows that, as for the functional goals, the stakeholders stated the following:
    - FR.1. The system must report the renewal risks at least three months before the renewal due date.
    - FR.2. The system must analyze data from different sources, such as purchase records and submitted support tickets, to achieve accurate predictions.
    - FR.3. The model should not only list renewal risks, but it should suggest an action plan of how to mitigate these risks. It is preferred for the model to adopt the prescriptive analytics [44] paradigm and quantify the effect of future decisions to advise on possible results.
    - FR.4. The system should allow the user to give feedback about the predictions.
- Regarding the non-functional requirements, the users stated two soft goals which are shown in Figure 3.8 as parallelograms with dashed borders:

- NFR.1. As for the quality metric that determines the model's performance, the stakeholders expressed a business goal of achieving a high level of accuracy. They set a value of 85% as the minimum accuracy value the model must achieve to be considered for deployment.
- NFR.2. Additionally, they emphasized that a certain level of interpretably is required to understand the reasoning behind the model decisions. Since interpretability [12] of ML models denotes producing predictions that are understandable to the end-users, it is considered as an unavoidable requirement in B2B predictive systems.

Furthermore, to ensure that the business has enough data (RQ 3.1.2), the research team initiated a second iteration of interviews in which they had a group meeting with two data scientists and one business analytic engineer from the analytics team. During this interview, both teams decided on using supervised learning algorithms to build the model with the *renewal transactions dataset*. With the initial requirements and the identification of the input dataset, the research team decided to proceed to the second phase.

3.6.3. Phase 2 – First Development Iteration

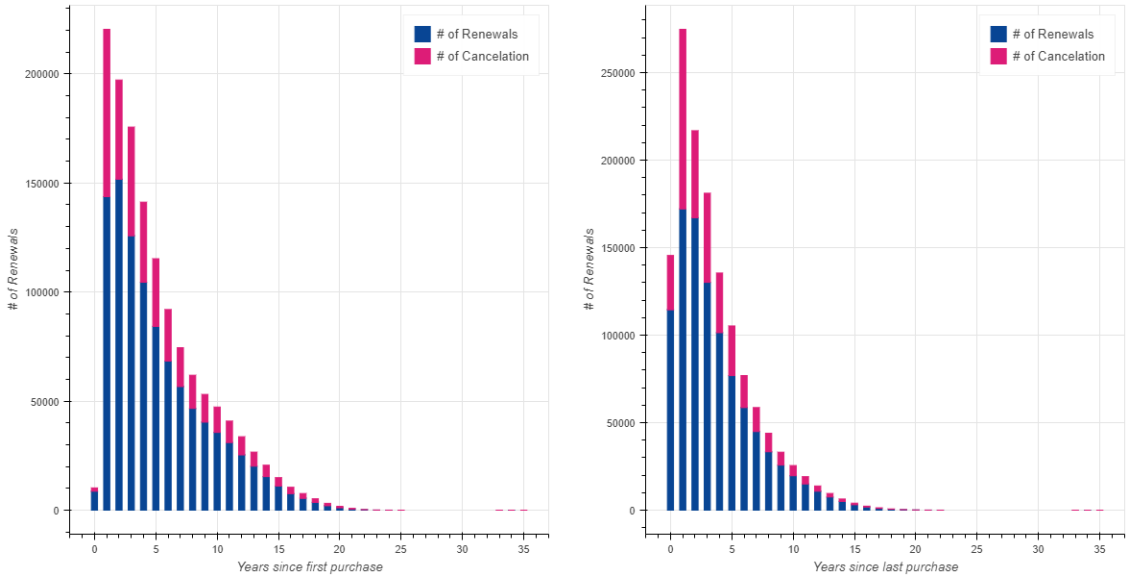
During the data preparation phase, the research team conducted an ethnographic study in which they observe the analytics team while performing the following preparation actions:

- *Feature engineering:* the analytics team employed the domain knowledge gained from the last phase to derive the features. The final set of features has eleven features, which can be classified into three categories. The first category includes attributes associated with the client's agreements, such as the agreement's revenue. The second category contains information about the customer's history, such as the number of years since the customer purchased his first and last product of the license. The last category comprises a set of aggregated features, such as the number of previous renewals and cancellations.
- *Generating ground truth:* since supervised learning algorithms need labeled data [45], the analytics team created a labeling function to automatically label the dataset. The labeling function considered specific attributes in the data, such as the end date and the purchase date for two consecutive purchase records. The team used this function to generate ground truth for the data.

- *Deciding on missing values and outliers:* on the one hand, the data analytics team found a small percentage (1.65%) of points with missing values for some attributes, and considering the massive volume of the data, these points were ignored by deletion. On the other hand, most of the outliers in the dataset were results of human errors, and hence, were excluded.
- *Data anonymization:* due to the commercial sensitivity of the dataset, the analytics team completely anonymized the data before sharing it with the research team.

Afterward, using the list of the initial requirements, the research team formulated the following hypotheses list. To speed up the development process, the hypotheses list only focused on a subset of the requirements, more specifically FR.1 and NFR.1:

- ***Hypothesis 1:*** a model that predicts and reports renewal risks at the beginning of each quarter will add business value (FR.1).
- ***Hypothesis 2:*** the generated ground truth, which is used to build the model is accurate.
- ***Hypothesis 3:*** a model can be built using supervised learning algorithms and achieve a minimum accuracy value of 85% (NFR.1).



(a) (b)

Figure 3.8: Demonstration of the model's input

(a) Renewals stats vs. Year since first purchase (b) Renewals stats vs. Years since last purchase

Table 3.5: MVM preliminary results in the first development iteration

Model	MCC	Accuracy	Precision	Recall	Training Time (seconds)
Logistic Regression	0.01	0.73	0.81	0.73	24.40
Random Forest	0.32	0.76	0.79	0.91	41.20
XGBoost	0.33	0.77	0.78	0.96	39.70

The development of the model was straightforward; the model has a binary output representing the target class as 0 (The customer will cancel) or 1 (The customer will renew). To develop the first MVM, we fed the dataset into multiple supervised ML algorithms: Logistic Regression [46], Random Forest [47], and XGBoost [48]. Since we had access to a large number of renewal records, the held-out method was preferred over cross-validation. Therefore, each model was trained with a subset of 907,651 records and tested using a held-out test subset of 447,053 records. Table 3.5 presents the preliminary results of the performance metrics achieved in each case, along with the computational time of the model training (in seconds). The performance metrics include Matthews correlation coefficient (MCC) [49], accuracy, precision, and recall. The experiments were conducted on a machine with a Core i7 processor and 32 GB RAM.

Based on the results shown in the table, we selected XGBoost classifier to build the initial MVM, as it achieved the highest accuracy values (NFR.1). To evaluate the model, both qualitative and quantitative data were collected using statistical validation and user feedback (Section 3.5.2). The evaluation data was collected as follows:

- *Quantitative data:* the results of the evaluation can be seen in the confusion matrix [50] in Table 3.6. The table presents True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN). The table shows that, for predicting non-renewals, the model achieved an accuracy value of 26.21% and a precision of 70.88%. The results attest that the model could not achieve the quality threshold defined in the performance criteria hypothesis (Hypothesis 3).

Table 3.6: MVM confusion matrix

True Labels	Total	Non-Renewals	Renewals
Non-Renewals	120,686	31,631 (TN) 26.21%	89,055 (FP) 73.79%
Renewals	326,367	12,994 (FN) 3.98%	313,373 (TP) 96.02%

- *Qualitative feedback:* To gather user feedback, we conducted a follow-up meeting with three S&S representatives and a program leader to review the model output. During the meeting, the research team demonstrated an overview of the dataset and a set of ten predictions produced by the model, along with the quality metrics achieved by the model. Figure 3.9 shows a sample of our demonstration regarding the dataset. The figure shows the relationship between the renewals statuses and the number of years since the customer's first purchase (Figure 3.9 (a)) and the last purchase (Figure 3.9 (b)).

The meeting aimed at involving the users in evaluating both the input and the output of the model. After complete anonymization and aggregation, summarized results were presented to two program leaders and one program director. The meeting's findings can be summarized in the following points:

- The end users pointed out that there is a level of inconsistency between the statistical insights derived from the data and their domain experience. For example, Figure 3.9 (a) shows an increasing trend in renewals after the first year of purchase, while most representatives stated that most cancellations occur after the first year. Hence, some concerns were raised about the accuracy of the labels, which were generated using a programmatic procedure in the data.
- The model performance was not satisfactory for end-users, as they were not willing to look at a set of reported renewal risks, while only 70.88% of them are real problems.

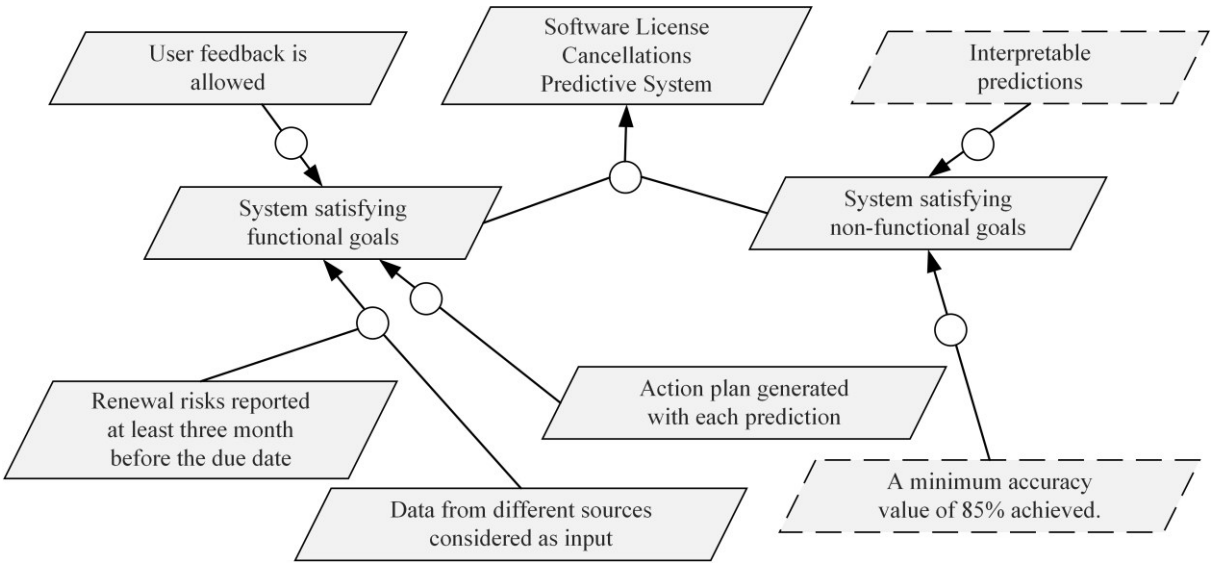


Figure 3.9: Generic goal model for the license cancellations predictive system

As a result, the research team, the analytics team, and the management team decided to reflect the insights into the hypotheses list and initiate a second iteration of development.

3.6.4. Phase 2 – Second Development Iteration

The insights collected from the evaluation guided the research team to refute the hypothesis of the data quality (hypothesis 2) and pivot to a new direction for generating the ground truth. As a result, a new hypothesis is added as:

- ***Hypothesis 2:*** Generating the ground truth must exploit domain expertise to ensure the validity of the labels and enhance the model accuracy.

To test the new hypothesis, the research team reviewed the data preparation component, as a new set of labels is required. The team concluded that for any labeling technique that is applied to the business domain, it is required to find a midpoint between labeling accuracy and labeling cost. To find this mid-point, the research team applied the hybrid method from their previous work [51] to generate the ground truth. The final dataset, along with the updated ground truth, is then used to train the XGBoost classifier. The generated model was evaluated as follows:

- *Quantitative data:* The results after using the updated ground truth are presented in Table 3.7. The table attests that the model accuracy increased by 116% and 3.05% in predicting non-renewals and renewals, respectively, compared to the last development iteration. Moreover,

the model performance validated the third hypothesis by achieving an accuracy value of 0.87 and an MCC value of 0.67.

- *Qualitative feedback:* The research team had a group meeting with four S&S representatives and two program leaders to review the model output. Since the end-users were involved in creating the ground truth [51], they trusted the input data. The meeting focused on reviewing a set of ten predictions presented by the model. Overall, the model performed well in predicting the license cancellations. However, the end-users suggested some enhancements regarding the model's interpretability (NFR. 2).

With these findings, the management team and the analytics team decided to persevere. The persevering decision had two main aspects. Firstly, the teams decided to start a phase of field testing to deploy the model and revalidate the hypotheses with live data. Secondly, the teams decided to run a third iteration of development in which the hypotheses set are updated to accommodate future improvements.

3.6.5. Phase 2 – Plans for The Third Development Iteration

Field testing aims at evaluating the model with live data taken from the business domain. Although, at this point, we have a model that has validated a partial set of its requirements (Figure 3.8), the framework aims at pushing the MVMs into production as soon as possible. Running the model with live data has its advantages. First, it gives a real perception of model performance. Second, it allows the development team to monitor performance degradation at the same time they are working on future improvements. Thus, the team can formulate new hypotheses that need to be validated. Finally, starting the field testing in parallel with conducting further development iterations can speed up model development.

To initiate the field testing, the teams took the following steps:

- The model was deployed internally in IBM with a re-evaluation period of three months.
- At the beginning of each quarter, the model will be used to predict renewal risks. During the quarter, S&S representatives will review the predictions and give feedback to evaluate each prediction.

Table 3.7: MVM confusion matrix (Iteration II)

True Labels	Total	Non-Renewals	Renewals
Non-Renewals	120,686	68,197 (TN) 56.51%	52,489 (FP) 43.49%
Renewals	326,367	3,450 (FN) 1.06%	322,917 (TP) 98.94%

- At this point, a sample of the end-users is chosen to engage in field testing. The sample includes two program leaders of worldwide S&S business and four S&S representatives.
- At the end of the quarter, the quarter data of renewal purchase transactions, along with the user feedback, will be used to create a status report about the model performance for re-evaluation.
- Unfortunately, automation could not be fully adopted at this point. However, the development team decided to schedule a job on the production server to collect archival data that includes:
  - the data of renewal purchase transactions during the last quarter; and
  - the user feedback recorded during the quarter
- This data will then be used to evaluate the model performance and decide if the model needs retraining. As mentioned in Section 3.5.3, this process of continuous monitoring should run forever to prevent performance degradation.

Alternatively, the development team has formulated the following hypotheses for the next development iterations:

- ***Hypothesis 4:*** considering the client submitted support tickets along with the purchase records in the model's input will enhance the model accuracy (FR.2).
- ***Hypothesis 5:*** providing the model's rationale for the generated predictions will enhance end-user trust (NFR.2).
- ***Hypothesis 6:*** providing an action plan along with each prediction will enhance the model's business value and guide the users to optimized solutions to save the renewals (FR.3).

As we were writing this article, the development team was about to start their first round of field testing. Hence, no further data could be collected at this point. As for our next steps, the development team plans to proceed with the third iteration of development to validate hypothesis 4. The team is currently preparing the problem management reports dataset (Table 3.4) to add it

to the input data. All in all, the parallel process of conducting field testing, along with initiating more development iterations, will be repeated as needed to optimize the model's compliance with the system requirements.

3.7. Discussion and Threats to Validity

This section, firstly, discusses the results and lessons learned from the case study. Secondly, it lists the threats to validity.

3.7.1. Discussion

Our research seeks to build an end-to-end framework for developing B2B predictive systems. During the application of the M-Lean framework, the results validated our hypotheses (Section 3.4). As for the first hypothesis (H1), interaction with business leaders and end-users helped to shorten the development time. Without them pointing out the inaccuracies in the input labels during the first development iteration (Section 3.6.3), it would take the development team a longer time before considering reviewing the ground truth. Also, regarding the second hypothesis (H2), the results acquired during requirements elicitation show that business requirements are essential for defining the business value of the model and the statistical metrics used to validate its performance. Defining these business constraints, in the beginning, helped the team to shape the development iterations to increase user trust and acceptance. In the following subsection, we discuss the lessons learned from the case study through the main principles of action research methodology [52].

3.7.1.1 Examining Problem Features

The M-Lean framework is designed to be applicable in other organizations that match firmographics variables specified in Section 3.3, given that these organizations have enough data to apply ML. Also, the framework assumes that the organizations are serious about their analytics transformation, which implies that the data science teams in these organizations are actively engaged in collecting masses of data from various sources and developing models to serve different business units. Another hypothesis the framework assumes about the organization is about the technical background of its data science team, as the framework requires the analytics team to have experience in requirements engineering, data analytics, and ML techniques.

Moreover, before applying ML techniques, organizations should do a cost-benefit analysis to estimate if the potential business value is worth the implementation cost. In our case study, this was accomplished during the first phase (Section 3.6.2), when mid-level management stated that, for large multinational IT companies like IBM, renewal of software licenses contributes to the selling organization's revenue. Therefore, utilizing ML to anticipate the renewal risks is expected to have positive business value to the sales unit.

3.7.1.2 Practitioners Commitment

The application of the framework requires a long-term commitment from the organization and the participants. As for the organization, we found applying the framework in many ways, similar to committing to building a product, except in this case, the product is a B2B predictive system. According to the report from McKinsey institute [53], organization's commitment to ML is empowered by the benefits anticipated from the output systems. Moreover, since building user

Table 3.8: Overhead cost for applying the M-Lean framework

Stakeholder Group	Number of individuals	Phase	Number of hours per individual	Total
Sales and Subscription Representatives	8	Phase 1 (Requirement Elicitation)	1.0	8
	3	Phase 2 (Development Iteration I)	2.0	6
	4	Phase 2 (Development Iteration II)	1.5	6
Mid-level Managers	3	Phase 1 (Idea Suggestion)	1.0	3
	1	Phase 2 (Development Iteration I)	2.0	2
	2	Phase 2 (Development Iteration II)	1.5	3
Analytics Development Team	4	Phase 1 (Data discovery)	6.0	24
	2	Phase 2 (Development Iteration I)	2.0	4
Research Team	2	Phase 1 (Transcribing and Analyzing Interviews)	46.0	92

trust and accommodating user feedback is essential to create a successful B2B predictive model [12], in our case study, the organization's commitment to the M-Lean processes was trivial to sustain a successful product line.

Also, as for participant's commitments, applying M-Lean indeed adds an overhead cost to the costs associated with the standard application of ML (e.g., cost of data preparation). The overhead cost is related to data collection and analysis processes during the framework phases (e.g., conducting and transcribing interviews). To quantify these costs in the form of personal effort, Table 3.8 shows a detailed cost breakdown of our case study. The table shows that the groups that bore the highest costs were the research team and the analytics team. As for the research team, the costs came from transcribing the interviews. Although we found transcribing necessary in this setting, this cost could be significantly reduced if transcribing is skipped in the first two layers (as mentioned before in Section 3.6.2). Alternatively, as for the analytics team, a total of 24 hours of interviews were logged during the data discovery phase (Phase 1). However, most of this time was to inform the research team about the datasets stored in the organization. Therefore, it is expected for this time to decrease when an organization applies the framework without consultancy. Also, based on the feedback collected throughout the case study, we think that these costs are justified by the outcome, which is a MVM that was iteratively developed and evaluated to reflect domain knowledge about the renewal process.

3.7.1.3 Cyclical Process Model

It can also be seen that, in our case study, the research team played the mediator role during the first phase (Section 3.6.1). Since the phase involves iterating through different layers of interviews, there must be a team whose responsibility is to coordinate between the stakeholder groups and ensure that all the stakeholders are actively participating. As mentioned in Section 3.4, the research team was integrated with the data science team; and hence, their roles can be interchanged by the analytics team. Moreover, we think that the mediator role can also be traded by applying some performance management strategies such as a balanced scorecard [54]. Balanced scorecards can be designed according to the business needs to accommodate the business goals targeted from each phase, then the management team can determine the best way to achieve these needs. In this manner, the mid-level managers can keep track of the execution of framework activities throughout its phases and monitor the outcomes arising from each layer.

3.7.2. Threats to Validity

Regarding the threats of validity [55], we list the first external validity threat as the possible lack of generalizability of the framework to other organizations since the framework was designed and evaluated using a partnership with only one organization. To mitigate this threat, we specify the firmographic variables of the proposed framework (Section 3.3). Moreover, we isolated the design specifications in each phase from the phase outline to allow the users to explore other methods for data collections. Another threat to construct validity [55], is related to the mapping of the data collected during each phase of the framework to the thematic codes. To mitigate that threat, we applied different types of data triangulation [28] to increase the precision of the results. First, we combined different methods of data collections. Second, we collected data from different sources (i.e., interviewees with different roles in the organization) on different occasions (i.e., collecting data at fixed points). Moreover, we shared our thematic codes with the IBM Analytics team and the management sample through general discussions to validate the data mappings. Finally, the application of the proposed framework with IBM lasted over nine months, which helped create a continued interaction with the participants and validating the collected information at different temporal points.

3.8. Conclusions

In this chapter, we present an end-to-end framework for developing B2B predictive models. We designed the framework following the lean startup methodology to accommodate the business domain and the end-user perspectives. The proposed framework consists of a set of phases during which the framework users can define possible opportunities for machine learning, develop, evaluate, and deploy machine learning systems. The framework presents the predictive models as a new class of problems in requirements engineering. Thus, it proposes a generic approach to organizations to define and elicit requirements for such models. To evaluate the framework, we have undertaken a case study to which we applied the framework to develop a predictive model within our industrial partner IBM. The framework was used to identify the initial requirements of the final model. To develop the model, we performed two development iterations. At the end of each iteration, the model was evaluated using both statistical validation and user feedback. The first iteration introduced having appropriate training labels as one of the most critical systems

requirements for machine learning models in the business domain. Additionally, the case study originated a minimum viable model to detect risks of license cancellation. The model is currently undergoing field testing. The results of the case study attest that the application of the proposed framework can help organizations to utilize their stored datasets to effectively build predictive models.

References

[1] M. Bilal, L. O. Oyedele, J. Qadir, K. Munir, S. O. Ajayi, O. O. Akinade, H. A. Owolabi, H. A. Alaka, and M. Pasha, "Big Data in the construction industry: A review of present status, opportunities, and future trends," *Advanced engineering informatics*, 2016.

[2] S. Yin and O. Kaynak, "Big data for modern industry: challenges and trends [point of view]," *Proceedings of the IEEE*, 2015.

[3] X. Jin, B. W. Wah, X. Cheng, and Y. Wang, "Significance and challenges of big data research," *Big Data Research*, 2015.

[4] I. A. T. Hashem, I. Yaqoob, N. B. Anuar, S. Mokhtar, A. Gani, and S. U. Khan, "The rise of 'big data' on cloud computing: Review and open research issues," *Information Systems*, 2015.

[5] C. P. Chen and C.-Y. Zhang, "Data-intensive applications, challenges, techniques and technologies: A survey on Big Data," *Information Sciences*, 2014.

[6] N. Gordini and V. Veglio, "Customers churn prediction and marketing retention strategies. An application of support vector machines based on the AUC parameter-selection technique in B2B e-commerce industry," *Industrial Marketing Management*, 2017.

[7] V. Tsoukalas and N. Fragiadakis, "Prediction of occupational risk in the shipbuilding industry using multivariable linear regression and genetic algorithm analysis," *Safety Science*, 2016.

[8] M. Chen, Y. Hao, K. Hwang, L. Wang, and L. Wang, "Disease prediction by machine learning over big data from healthcare communities," *IEEE Access*, 2017.

[9] S. Erevelles, N. Fukawa, and L. Swayne, "Big Data consumer analytics and the transformation of marketing," *Journal of Business Research*, 2016.

[10] A. Y. L. Chong, E. Ch'ng, M. J. Liu, and B. Li, "Predicting consumer product demands via Big Data: the roles of online promotional marketing and online reviews," *International Journal of Production Research*, 2017.

[11] M. Salehan and D. J. Kim, "Predicting the performance of online consumer reviews: A sentiment mining approach to big data analytics," *Decision Support Systems*, 2016.

[12] M. Vlachos, V. G. Vassiliadis, R. Heckel, and A. Labbi, "Toward interpretable predictive models in B2B recommender systems," *IBM Journal of Research and Development*, 2016.

[13] E. Ries, *The lean startup: How today's entrepreneurs use continuous innovation to create radically successful businesses*. Crown Books, 2011.

[14] R. Malhotra, "A systematic review of machine learning techniques for software fault prediction," *Applied Soft Computing*, 2015.

[15] J. Wen, S. Li, Z. Lin, Y. Hu, and C. Huang, "Systematic literature review of machine learning based software development effort estimation models," *Information and Software Technology*, 2012.

[16] M. Shepperd, D. Bowes, and T. Hall, "Researcher bias: The use of machine learning in software defect prediction," *IEEE Transactions on Software Engineering*, 2014.

[17] Q. Song, X. Zhu, G. Wang, H. Sun, H. Jiang, C. Xue, B. Xu, and W. Song, "A machine learning based software process model recommendation method," *Journal of Systems and Software*, 2016.

[18] Z. Kurtanović and W. Maalej, "Automatically classifying functional and non-functional requirements using supervised machine learning," in *Requirements Engineering Conference (RE), 2017 IEEE 25th International*, 2017.

[19] A. Perini, A. Susi, and P. Avesani, "A machine learning approach to software requirements prioritization," *IEEE Transactions on Software Engineering*, 2012.

[20] P. Avesani, A. Perini, A. Siena, and A. Susi, "Goals at risk? Machine learning at support of early assessment," in *Requirements Engineering Conference (RE), 2015 IEEE 23rd International*, 2015.

[21] S. Landset, T. M. Khoshgoftaar, A. N. Richter, and T. Hasanin, "A survey of open source tools for machine learning with big data in the Hadoop ecosystem," *Journal of Big Data*, 2015.

[22] X. Meng, J. Bradley, B. Yavuz, E. Sparks, S. Venkataraman, D. Liu, J. Freeman, D. Tsai, M. Amde, S. Owen, D. Xin, R. Xin, M. J. Franklin, R. Zadeh, M. Zaharia, and A. Talw, "Mllib: Machine learning in apache spark," *The Journal of Machine Learning Research*, 2016.

[23] E. R. Sparks, S. Venkataraman, T. Kaftan, M. J. Franklin, and B. Recht, "Keystoneml: Optimizing pipelines for large-scale advanced analytics," in *IEEE 33rd International Conference on Data Engineering*, 2017.

[24] M. Vartak, H. Subramanyam, W-E. Lee, S. Viswanathan, S. Husnoo, S. Madden, and M. Zaharia, "Model DB: a system for machine learning model management," in *Proceedings of the Workshop on Human-In-the-Loop Data Analytics*, 2016.

[25] B. R. Bodenmann and K. W. Axhausen, "Synthesis report on the state of the art on firmographics," *Institute for Transport Planning and Systems, ETH, Zurich*, 2010.

[26] P. Sugimura and F. Hartl, "Building a Reproducible Machine Learning Pipeline," *arXiv preprint arXiv:1810.04570*, 2018.

[27] R. L. Baskerville and A. T. Wood-Harper, "A critical perspective on action research as a method for information systems research," *Journal of information Technology*, 1996.

[28] N. Carter, D. Bryant-Lukosius, A. DiCenso, J. Blythe, and A. J. Neville, "The use of triangulation in qualitative research.," in *Oncology nursing forum*, 2014.

[29] R. B. Svensson, T. Gorschek, B. Regnell, R. Torkar, A. Shahrokni, and R. Feldt, "Quality requirements in industrial practice—an extended interview study at eleven companies," *IEEE Transactions on Software Engineering*, 2012.

[30] I. Etikan, S. A. Musa, and R. S. Alkassim, "Comparison of convenience sampling and purposive sampling," *American Journal of Theoretical and Applied Statistics*, 2016.

[31] E. Souza, A. Moreira, J. Araújo, S. Abrahão, E. Insfran, and D. S. da Silveira, "Comparing business value modeling methods: A family of experiments," *Information and Software Technology*, 2018.

[32] I. Hadar, P. Soffer, and K. Kenzi, "The role of domain knowledge in requirements elicitation via interviews: an exploratory study," *Requirements Engineering*, 2014.

[33] A. Sutcliffe and P. Sawyer, "Requirements elicitation: Towards the unknown unknowns," in *Requirements Engineering Conference (RE), 2013 21st IEEE International*, 2013.

[34] S. K. Kwak and J. H. Kim, "Statistical data preparation: management of missing values and outliers," *Korean Journal of anesthesiology*, 2017.

[35] M. Ortega, M. Pérez, and T. Rojas, "Construction of a systemic quality model for evaluating a software product," *Software Quality Journal*, 2003.

[36] H. Brink, J. W. Richards, M. Fetherolf, and B. Cronin, *Real-world machine learning*. Manning, 2017.

[37] S. Easterbrook, J. Singer, M.-A. Storey, and D. Damian, "Selecting empirical methods for software engineering research," in *Guide to advanced empirical software engineering*, Springer, 2008.

[38] S. Liu, X. Wang, M. Liu, and J. Zhu, "Towards better analysis of machine learning models: A visual analytics perspective," *Visual Informatics*, 2017.

[39] D. Sculley, T. Phillips, D. Ebner, V. Chaudhary, and M. Young, "Machine learning: The high-interest credit card of technical debt," 2014.

[40] L. Li, S. Chen, J. Kleban, and A. Gupta, "Counterfactual estimation and optimization of click metrics in search engines: A case study," in *Proceedings of the 24th International Conference on World Wide Web*, 2015.

[41] P. Runeson and M. Höst, "Guidelines for conducting and reporting case study research in software engineering," *Empirical software engineering*, 2009.

[42] A. T. Bahill and A. M. Madni, "Discovering system requirements," in *Tradeoff Decisions in System Design*, Springer, 2017.

[43] S. Tueno, R. Laleau, A. Mammar, and M. Frappier, "The SysML/KAOS Domain Modeling Approach," *arXiv preprint arXiv:1710.00903*, 2017.

[44] S. Srinivas and A. R. Ravindran, "Optimizing outpatient appointment system using machine learning algorithms and scheduling rules: A prescriptive analytics framework," *Expert Systems with Applications*, 2018.

[45] R. Caruana, N. Karampatziakis, and A. Yessenalina, "An Empirical Evaluation of Supervised Learning in High Dimensions," in *Proceedings of the 25th International Conference on Machine Learning*, 2008.

[46] D. W. Hosmer Jr, S. Lemeshow, and R. X. Sturdivant, *Applied logistic regression*, vol. 398. John Wiley & Sons, 2013.

[47] P.-N. Tan, M. Steinbach, and V. Kumar, "Classification: alternative techniques," *Introduction to data mining*, 2005.

[48] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016.

[49] D. M. Powers, “Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation,” 2011.

[50] T. Fawcett, “ROC graphs: Notes and practical considerations for researchers,” *Machine learning*, 2004.

[51] M. Nashaat, A. Ghosh, J. Miller, S. Quader, C. Marston, and J. Puget. “Hybridization of Active Learning and Data Programming for Labeling Large Industrial Datasets.” In *2018 IEEE International Conference on Big Data (Big Data)*, 2018.

[52] C. Wohlin and A. Aurum, “Towards a decision-making structure for selecting a research design in empirical software engineering,” *Empirical Software Engineering*, 2015.

[53] “Special Edition on Advanced Analytics in Banking,” McKinsey&Company.

[54] H. A. Akkermans and K. E. Van Oorschot, “Relevance assumed: a case study of balanced scorecard development using system dynamics,” in *System Dynamics*, Springer, 2018.

[55] F. Shull, J. Singer, and D. I. Sjøberg, *Guide to advanced empirical software engineering*. Springer, 2007.

Chapter 4 : Asterisk: Generating Large Training Datasets with Automatic Active Supervision

4.1. Introduction

Organizations in different domains are increasingly investing in machine learning to empower their data-driven decisions. However, one of the most tedious tasks in creating machine learning models is obtaining hand-labeled training data, especially with the new revolutionary advances that deep learning methods bring to the field of machine learning. Since such techniques require large training datasets [1], the cost of labeling these datasets has become a significant expense for businesses and large organizations. In real-world settings, domain experience is usually required to accomplish, or at least supervise such labeling processes; this makes the process of obtaining large-scale hand-labeled training data prohibitively expensive.

For these reasons, several researchers [2]–[7] have proposed techniques to generate training data with minimal annotation effort. One approach that aims at generating labeled datasets at scale is weak supervision [2]. In weak supervision, practitioners turn to noisy labels [3], which are programmatically generated using cheaper annotation sources such as crowdsourcing [4], external knowledge bases [5], and user-defined heuristics [6]. Previous research [6], [8], [9] has shown that weak supervision can produce less-than-ideal training datasets at a large scale for a wide range of applications. These labels can then be used to train many complex machine learning models, such as deep learning. Alternatively, other well-studied techniques rely on semi-supervised learning [10], [11]. Semi-supervised techniques exploit a small labeled set to derive assumptions about the data structure and leverage a larger unlabeled dataset. For this purpose, some techniques [11] employ the concept of generative models to utilize the unlabeled data and learn the data representation. Generative models produce samples after learning the underlying data distribution; these samples can then be used as training labels for discriminative models.

On the other hand, active learning (AL) [7] is a special kind of semi-supervised learning which has been used for decades to achieve a high level of classification accuracy while optimizing the annotation cost. In AL settings, instead of manually labeling an entire dataset, an algorithm

iteratively selects the most valuable points to classify and asks the user to only label these points. Although AL does not aim at producing labeled datasets, it helps in reducing the annotation cost while building machine learning models that generalize beyond the training data.

A closer look at these labeling techniques, however, reveals several gaps and shortcomings [12]–[16]. On the one hand, since cheaper annotation methods are used in weak supervision, these sources are expected to overlap and conflict, which affects the quality of the resulting labels [12]. To estimate the level of noise in the generated labels, previous studies introduce the data programming (DP) paradigm [2], [12], which uses generative models to integrate the outcome of multiple weak supervision. Nevertheless, the uncertainty levels originating from these weak sources can complicate the process of learning the structure of these generative models [12]. Moreover, these approaches require users to design a set of user-defined heuristics [6] to encode their domain experience, which can be an expensive and time-consuming process [13].

On the other hand, active learning can be expensive when applied to high-dimensional datasets [14]. Since, in pool-based settings [17], the active learner performs an iterative process to choose one or more points from an unlabeled pool to query the user in each iteration. This iterative process involves ranking all the points in the unlabeled pool, selecting the points for which correct labels should be provided, training a model, and evaluating its performance using a held-out test set. Therefore, any imbalance between the sizes of the unlabeled pool and the labeled dataset can affect the time complexity of the process and increase the annotation cost [14]. Moreover, other studies [15], [16] show that in situations where the unlabeled data points cannot be entirely separated, active learning does not provide much superiority over passive learning.

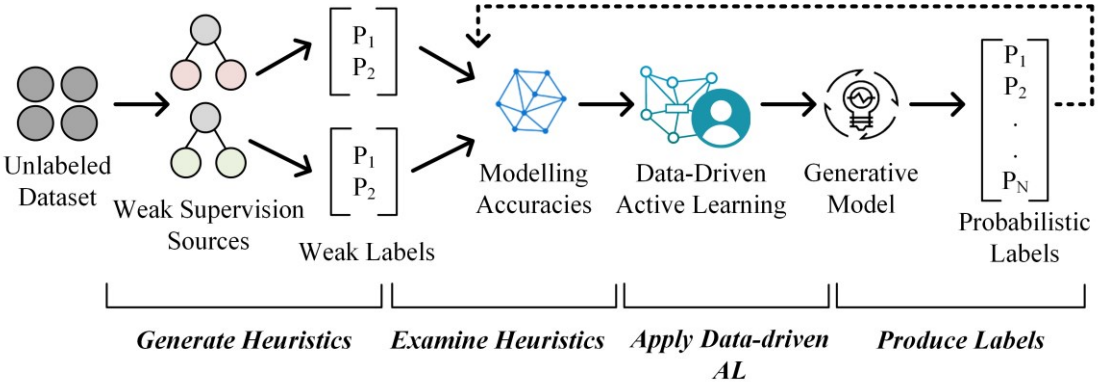


Figure 4.1: An overview of the proposed system

To overcome some of these challenges, we propose Asterisk, a framework to generate high-quality training datasets at scale. An overview of the system is presented in Figure 4.1. As shown in the figure, instead of relying on the end-users to write user-defined heuristics, the proposed approach exploits a small set of labeled data and automatically produces a set of heuristics (weak supervision sources) to assign initial labels. In this phase, the system applies an iterative process of creating, testing, and ranking heuristics in each, and every, iteration to only accommodate high-quality heuristics. Then, Asterisk examines the disagreements between these heuristics to model their accuracies. To enhance the quality of the generated labels, the framework improves the accuracy of the heuristics by applying a novel data-driven AL process. During the process, the system examines the generated weak labels along with the modeled accuracies of the heuristics to help the learner decide on the points for which the user should provide true labels. The process aims at enhancing the accuracy and the coverage of the training data while engaging the user in the loop to execute the enhancement process. Therefore, by incorporating the underlying data representation, the user is only queried about the points that are expected to enhance the overall labeling quality. Then, the true labels provided by the users are used to refine the initial labels generated by the heuristics. As the figure shows, the refinement process can be repeated to further enhance the quality of the generated labels. Finally, the framework examines the refined labels and outputs a set of probabilistic labels that can be used to train any downstream classifier.

To evaluate the proposed method, we compare its performance with the performances of four state-of-the-art techniques, including data programming [2], automated weak supervision [13], and traditional active learning strategies [17]. During the experiments, we report the labeling accuracy, annotation cost, and the performance of the end model trained with the generated labels. The primary contributions of this research can be summarized as follows:

- An end-to-end labeling framework is proposed to create high-quality, large-scale training datasets. We describe the architecture of the proposed system, which includes a novel process of automatic generation of labeling heuristics instead of relying on the end-user to manually define the weak sources.
- We propose a data-driven active learning process to enhance the accuracy of the generated weak labels. The process learns the selection policy while considering the distribution of the underlying data and the labeling confidence to optimize user engagement.

- We applied a comprehensive set of experiments to evaluate the proposed method against state-of-the-art techniques. The experimental evaluation explores a wide range of domains with ten datasets that vary in size and dimensionality with a maximum size of 11M records. We also use a real-world business dataset of 1.5M records provided by our industrial partner, IBM. The experiments also include a micro-benchmarking to evaluate the individual components of the proposed approach.

The remaining of the chapter is structured as follows: Section 4.2 presents the background related to this research. Section 4.3 states, in detail, the design of the proposed solution. Section 4.4 presents the performed experiments and reports the obtained results. While Section 4.5 discusses related work, and Section 4.6 concludes the chapter.

4.2. Background

In this section, we first review weak supervision and the methods of automating weak supervision sources. In the second subsection, we discuss active learning and, more specifically, different approaches for meta-active learning.

4.2.1. Automated Weak Supervision

In weak supervision, domain experience is encoded in the form of high-level, low-quality sources such as user-defined heuristics. Each of these sources is then used to automatically generate noisy (weak) labels for an unlabeled dataset. Since these weak supervision labels are collected from sources with different coverage and accuracies, the main challenge of weak supervision is to combine these conflicting sources into a single label for each data point. To overcome such a challenge, researchers [2], [6], [12], [18], [19] try to estimate the accuracy of different weak supervision sources and use these estimates to produce combined labels. Most of these efforts [6], [18], [19] utilize generative models to assess the accuracies of multiple weak supervision sources and model the true label as a latent variable based upon a set of noisy observations. After modeling the accuracies, the generative model can output a set of probabilistic labels to work as the training dataset for any discriminative model.

Aiming at aggregating labels from different supervision sources, the data programming paradigm [2] learns the accuracy of different weak supervision sources by examining the disagreements

between them without the need for any ground truth. Data programming allows users to encode their domain experience using an ensemble of labeling functions [6]. The abstract concept of the labeling functions in data programming supports a wide range of weak supervision sources, including crowdsourcing and external knowledge bases. To denoise these sources, data programming builds a generative model to examine the dependency structure among these labeling functions and model their accuracy. However, other studies [20], [21] state that it can be challenging to estimate the noise level in the generated labels from data programming. Also, since the quality of the labels produced from the generative model affects the subsequent models being trained, it is essential to allow the user to debug and trace the output of the generative models [20]. However, this also can be a challenging task giving the complex structure of such models, especially when the weak sources show a high-level of dependencies [12].

Moreover, as the success of data programming depends on the quality of the weak supervision sources encoded as labeling functions, some research [13], [22] argues that the task of writing these labeling functions can be monotonous for end-users. Therefore, recent studies [13], [22] try to automate the process of creating weak supervision sources. For example, Varma *et al.* [13] present a system that can automatically generate weak supervision sources for an unlabeled dataset using a small labeled set. The system uses the small labeled dataset to iteratively create heuristics and tries to terminate this iterative process before the quality of the generated labels degrades. Also, Das *et al.* [22] propose an affinity coding paradigm that infers true labels of an unlabeled dataset by examining the similarity between the unlabeled points. The proposed system [22] derives the affinity scores from convolutional neural networks and uses these scores as signals to decide upon class membership.

However, although these efforts have illuminated the importance of automating weak supervision sources, there are some potentially open questions about the applicability of these techniques to real-world cases with large-scale datasets. For example, one of these approaches [13] was evaluated using a set of datasets with varying sizes but with a maximum size of 100K unlabeled points. Also, another technique [22] was only assessed with image classification tasks with a maximum size of 37,322 images. Therefore, we believe that the scalability of automated weak supervision is yet to be explored in real-world tasks with millions of unlabeled records, which represents one of the main motives driving our research.

4.2.2. Meta-Active Learning

Active learning optimizes the process of data collection needed to train a classifier by deciding upon which instances an oracle should label. In our research, we focus on the pool-based settings [17] in which a classifier is initially trained using a small set of labeled points (the seed). Then, the active learning algorithm iteratively selects one or more points from an unlabeled pool and asks the user to provide the correct labels for these points; then, it adds them to the labeled set to retrain the model. The model is then evaluated using a held-out test set, and the process is repeated to label more points until a target performance is reached or a predefined annotation budget is exhausted. The algorithm that decides on which points should be labeled is called the query strategy, and it is an essential part of the active learning process. Over the last decades, many query strategies have been proposed for different classification tasks [7], [17]. One of the most effective query strategies is uncertainty sampling [17]. The algorithm ranks the data instances in the unlabeled pool and chooses the point about which the current classifier is most uncertain. Another efficient algorithm is query-by-committee [17], which employs a committee of classifiers and selects the points about which the committee members disagree. Since these two methods tend to choose the points that lie on the classification boundary [17], they are known to be prone to select outliers. Therefore, to ensure that the selected points can be seen as representatives of other instances in the distribution, previous studies [23], [24] propose density-weighted uncertainty sampling in which the uncertainty sampling algorithm is augmented to consider both uncertainty and density measures simultaneously.

Although these algorithms have performed remarkably well in various tasks [17], [25], [26], previous studies [27]–[30] have pointed out that these strategies can be limited when dealing with different data distributions. Since these strategies apply fixed heuristics to measure the informativeness of the unlabeled points, they do not employ characteristics specific to the underlying learning problem [27]. Therefore, various factors, such as imbalanced classes and label noise, can make uncertainty sampling result in suboptimal decisions [30]. As a result, to overcome these limitations, recent studies [27], [29]–[32] propose the use of meta-active learning. In meta-active learning, the choice and the design of the query strategy changes depending on the underlying data distribution. For example, some studies [26], [31] propose combining existing query strategies to reduce over-fitting [26] or transfer the active learning experience [31].

Alternatively, other research [27], [29], [30], [32] goes further and tries to learn the query strategy; for instance, one technique [29] structures the process of designing the query strategy as a reinforcement learning problem to learn the data selection policy. Similarly, Konyushkova and Raphael [30] propose an active learning algorithm in which the query strategy is replaced with a regression function to decide which points to label. Also, Pang *et al.* [32] model the query strategy algorithm as a deep neural network to predict the next best point to query the user.

However, most of these studies have only been applied to certain domains and specific classification settings; for example, some techniques [30], [32] only consider a binary base classifier. Also, other research [30] tries to learn an active learning strategy for the aim of transferring query strategies to new domains; hence, their experiments focus more on the computational cost rather than reducing the annotation budget. Moreover, none of these works has been evaluated using massive datasets with millions of records. For instance, one approach [29] could cut the annotation cost by up to 30% with a labeling budget of 200 instances. Similarly, another technique [32] achieved more than 95% accuracy on a dataset with 70,000 records. Also, the approach proposed by Konyushkova *et al.* [30] could reach 95% accuracy with a dataset of

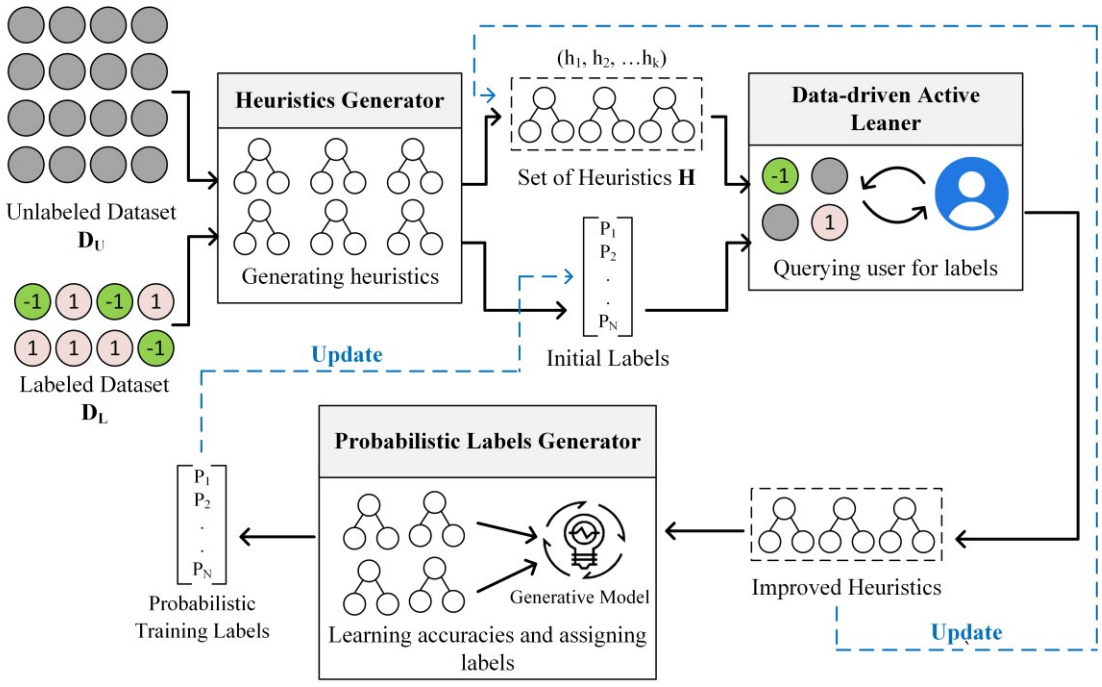


Figure 4.2: A component overview of the Asterisk framework

3,190 records. In short, among all these algorithms [29], [30], [32] that aim at learning the strategy of AL, a maximum of 70,000 records [32] was used in the evaluation. Also, with the increasing popularity of weak supervision, a critical research question rises, which is whether any of these meta-active learning techniques can work with the high level of noise in the labels collected from the weak sources.

4.3. Asterisk Architecture

In the following two subsections, we describe the architecture of the proposed system. While Section 4.3.1 formulates the input and output for Asterisk, Section 4.3.2 describes in detail the individual components of the proposed system.

4.3.1. Input and Output

The input to Asterisk is an unlabeled dataset D_U of size N and a small labeled dataset D_L with size M where M << N. The unlabeled dataset D_U consists of data points which are described as $\{\mathbf{x}_i, y_i\}_{i=1}^{N}$ where x_i represents a set of features describing the i^{th} observation (data point) in the dataset, and y_i is the unknown label associated with this point. Similarly, data points in the labeled dataset D_L are defined with a set of points $\{\mathbf{x}_i^*, y_i^*\}_{i=1}^{M}$ denoting the set of features $\mathbf{x}_i^*$and the corresponding known label y_i^* that describe the i^{th} data point. Both $\mathbf{x}_i, \mathbf{x}_i^* \in R^F$, are viewed as F features representing the data. Features are a set of measurable properties of the observed data points. Hence, a set of numerical values describing the i^{th} point can be described by a feature vector $\mathbf{x}_i$. For example, if the classification task is to predict the default payment of a client, $\mathbf{x}_i$ can be a set of numeric features describing the previous credit payments. For the sake of simplicity, we consider the binary classification situation, hence $y_i, y_i^* \in \{-1, 1\}$. As for the output data, the proposed system generates probabilistic training labels $\bar{y} = P[y = 1] \in [0, 1]$ for the points in the unlabeled dataset D_U which can be further used to train any noise-aware classifier.

4.3.2. Asterisk Design

The proposed system exploits the small labeled dataset D_L to produce a set of probabilistic labels $\bar{y}$ for the data points in the unlabeled dataset D_U. An overview of the system is shown in Figure 4.2. As the figure depicts, the system consists of three main components, namely *the heuristics*

generator, *the data-driven learner*, and *the probabilistic labels generator*. The heuristics generator component (Section 4.3.2.1) aims to automatically produce a set of heuristics to assign initial labels to the points in D_U. The second component, the data-driven learner (Section 4.3.2.2), works with the outcomes of the first component to further examine the data and refine the initial labels. This component aims to enhance the accuracy of the generated heuristics and increase the coverage of the generated training labels. Therefore, the component tries to economically engage the user to express their domain experience and uses their input in the refinement process. Finally, the probabilistic labels generator (Section 4.3.2.3) is used to learn the accuracy of these labels and assign a single label for each data point in D_U.

As the figure shows, the process of denoising and generating the final labels (the second and the third components) can be repeated to enhance the accuracy of the final labels. Since each iteration outputs an improved set of heuristics (from the second component) and a refined set of probabilistic labels (from the third component), the user can decide to initiate another cycle where these outputs are fed to the data-driven learner component to enhance the quality of the heuristics. Then, the label generator component can be used to produce more accurate labels. However, this requires increasing the budget of manual labeling since the user will be queried to label more points to help with the refinement process. Nonetheless, we found that running only one iteration of the process can help obtain a satisfactory level of classification accuracy for real-world tasks (Section 4.4) and achieve labeling accuracy of 90.09% on average (Section 4.4.2.1).

4.3.2.1. Heuristics Generator: Automating the Heuristics Production

The system starts with the heuristics generator component which takes the labeled set D_L and the unlabeled set D_U as inputs and outputs a set of heuristics H of size K denoted as (h_1, h_2, …h_k) and a vector of initial probabilistic labels $\bar{y}_{initial} \in [-1, 1]^N$ for the N points in D_U. Each heuristic in H follows the form $h_j(\dot{\mathbf{x}}_i) \rightarrow y_i \in \{-1, 0, 1\}$ where $\dot{\mathbf{x}}_i$ is a subset of the F features and is used as an input to h_j; and y_i is the weakly supervised label for the i^{th} point.

Since the heuristics exploit the labeled data in D_L to output labels for D_U, the examples provided in D_L are assumed to be handled by subject-matter experts and hence have a strong ground-truth value. In this component, we treat the process of generating heuristics as a process of creating a set of probabilistic classification models that take one or more features as input and calculate

probability distribution over a set of classes [33]. The process utilizes D_L to train and evaluate the generated heuristics. Hence, if there is a high level of noise in the examples provided in D_L, this can affect the process of creating the initial heuristics.

Algorithm 4.1: The Procedure of The Heuristics Generator Component

Input: Unlabeled Input dataset D_U, Small Labeled Input Dataset with Ground truth labels D_L

Output: Set of Heuristics H and a vector of initial probabilistic labels $\bar{y}_{initial}$

1: Compute C as the maximum number of features for the heuristics (Equation (4.1))

2: **for** $\grave{F}$ = 1…C **do**

3: F_combinations = all distinct subsets of features of size $\grave{F}$ from the original set of features F

4: **for** j = 1… length(F_combinations) **do**

5: $\grave{x}$= F_combinations [j,:]

6: Use $\grave{x}$ as input to build a heuristic h_j using an ensemble of decision stumps

7: Create a heuristic model $h_j = \sum_{m=1}^{M} \omega_m f_m(x)$

8: Apply h_j to D_L and produce predictions as y_j^*

9: P_j, R_j, MCC_j = calculate_performance(y_j^*, y^*) (Equation (4.3))

10: Apply h_j to D_U and compute the conditional probability $P(y_i = +1 \mid \grave{\mathbf{x}}_i)$

11: Estimate the confidence interval for h_j

12: Use the confidence interval to force h_j to abstain from labeling low confidence labels.

13: Calculate Hamming distance for h_j to estimate coverage

14: Compute $Rank_j$ for h_j (Equation (4.2))

15: Add the heuristic with the highest Rank to H

16: Use a generative model to learn the accuracies of H and produce $\bar{y}_{initial}$

17: return H, $\bar{y}_{initial}$

Nevertheless, applying some noise filtering techniques [3], [12] can help to eliminate the noise effect. Moreover, the proposed method utilizes meta-active learning in the second component to enhance the quality of the generated labels, which will help reverse the noise effect.

Then, the component uses the distribution provided by each heuristic to either assign labels to the unlabeled dataset (i.e., assigns either -1 or 1) or abstain (i.e., outputs (0)). More specifically, in our implementation, we use an ensemble of decision stumps [34] as the inner classification model to mimic the threshold-based heuristics that users usually write [6], [13]. However, changing the classification models in Asterisk to any probabilistic classifier should not require much engineering work.

To create the final set of heuristics H, we follow the iterative process shown in Algorithm 1. As the algorithm indicates, the process starts with *defining the input* (features) for the potential models (steps 1-3). Then, the process continues with *creating the models (heuristics)* (steps 5-8) and *evaluating their performance and coverage* (steps 9-14). Finally, the process *ranks the heuristics* generated by each, and every, iteration to decide upon which heuristic to add to the set H (steps 14-17); further details about these steps are presented as follows:

Defining the input. First, to choose the input features for the heuristics (models), we iterate over a range $\grave{F}$ from 1 to C, where C is the maximum number of features that can be used as input to the heuristics. In each iteration, the system generates distinct subsets of features of size $\grave{F}$. These subsets can be donated as $\binom{F}{\grave{F}}$ where $\grave{F}$ is the value assigned to the size of the input of the heuristics generated in this iteration. Although the users can define the maximum size C, based on the insights obtained from the conducted real-world experiments (Section 4.4), we adjust a default value for C as:

$$C = \lceil \ln(F) \rceil + 1 \tag{4.1}$$

to bound the number of iterations when the number of features F grows continuously. In other words, for high dimensional data, the proposed method tries to control the growth rate at which C expands as the number of features increases. At this point, the component aims to limit the number of inputs to the heuristics, so it does not affect the computational complexity of the proposed method. On the other hand, the component specifies a lower bound for C as one input feature for all the possible values of F where $F \geq 1$.

The inner classifiers are implemented as an ensemble of decision stumps that try to split the training examples in D_L into two subsets based on the values of one or more features in the data.

So, each classifier defines the input features and a set of thresholds to classify the training data into two groups.

Creating heuristics. To define the input for each heuristic, the component proceeds with designing a heuristic (model) for each possible combination resulting in $\sum_{F=1}^{C}\binom{F}{F}$ models. As for the threshold, each classifier finds the best threshold that fits the training examples and gives the best accuracy over D_L. The generated classifier in each iteration can be formulated as $h_j = \sum_{m=1}^{M} w_m f_m(x)$ where M is the number of decision stumps, w_m are the learned coefficients, and $f_m(x) \in \{-1,1\}$ denotes a single decision stump as $f(x) = s(x_k > T)$ where $s \in \{1,-1\}$, x_k is the k^{th} element in $\dot{x}$, and T is the threshold specified by the decision stump.

Moreover, like other probabilistic classification models, the generated heuristics estimates the conditional probability of a class label $P(y_i = +1 \mid \dot{\mathbf{x}}_i)$. The component forces the models to abstain when they are not confident about a generated label. To decide on the abstaining interval, the component examines the confidence interval for each heuristic h_j using D_L as a validation set [35]. The confidence level is adjusted according to the coverage of the heuristic (i.e., the percentage of D_U that receives a label from h_j). Since the confidence level denotes the degree at which the generated labels represent the distribution of D_U [35], we rely on the coverage achieved by each heuristic to determine the confidence level. Then, the component adjusts the abstaining interval accordingly as $\{a \mid 0 \leq a \leq 1 \text{ and } a \notin CI\}$ where CI is the confidence interval for h_j. This way, the component obliges the heuristics to only output labels for datapoints where they have high confidence, which helps to increase the accuracy of the generated heuristics.

Evaluating heuristics. When evaluating the performance of the heuristics produced during each iteration, the component also considers the overall coverage of the heuristics when applied to D_U. The component aims to widen the range of the data points that receive labels from H in D_U. In other words, the goal of the component is to output a set of heuristics that are individually accurate while achieving high labeling coverage when combined. Therefore, to estimate the performance of the heuristics, the system computes Precision (P), Recall (R), and MCC metrics for heuristics generated during each iteration. The performance metrics are computed by applying each heuristic to D_L. Since the generative model [6] assumes that weak sources encoded by the users always perform better than random, the component holds this assumption by only including heuristics with MCC values greater than 0.60. Alternatively, to evaluate the coverage of a heuristic h_j, we

examine the dissimilarity between the data points in D_U that are labeled by h_j and the points that already received labels from H. To compute the dissimilarity, we construct a vector $v_j \in \{0,1\}^N$ which represents whether each point in D_U receives a label from heuristic h_j (1) or not (0). Then, we obtain another vector $v \in \{0,1\}^N$ to represent whether any heuristic in H has assigned a label to the data points in D_U. Next, we compute the Hamming distance [36] between v_j and v and use it as a measure for the coverage of h_j. The motive behind using Hamming distance is that it is preferred when dealing with categorical attributes [26]. Also, since the important bits are ones that are different, Hamming distance can be used to return the number of bits at which the two vectors differ.

Ranking heuristics. After that, heuristics generated during each iteration are ranked based on performance (i.e., Recall, Precision, and MCC) and coverage (i.e., the Hamming distance) to decide on which heuristics to add to the final set. The ranking uses a weighted average of the performance metrics and the coverage distance as:

$$\text{Rank}_j = \omega \times f_1(R, P, \text{MCC}) + (1 - \omega) \times f_2(v_j, v) \tag{4.2}$$

where f_1 is the harmonic mean of R, P, and MCC. The value of MCC [52] is adjusted since the harmonic mean is only calculated for positive real numbers. Therefore, f_1 is computed as:

$$f_1(R, P, \text{MCC}) = ((R^{-1} + P^{-1} + (\text{MCC} + 1)^{-1})/3)^{-1} \tag{4.3}$$

and f_2 is the function to calculate the Hamming distance for h_j [36], and $\omega = 0.5$ to indicate equal weight between the coverage and the performance. Then, the component only chooses the highest-ranking heuristic to add it to the set H.

Finally, to combine the output of the heuristic and generate an initial vector of probabilistic labels $\bar{y}_{\text{initial}}$, we employ a generative model [12] to learn the accuracies of the heuristics in H and estimate any statistical dependency between their outputs. Then, the generative model employes the learned accuracies to produce a single probabilistic label for each data point in the unlabeled dataset.

It is essential to mention that, since the component encourages the heuristics to abstain from labeling points with low confidence labels, there might be a subset of points in the unlabeled dataset that do not receive a label from any heuristic in H, especially when $M << N$, which is the case in many real-world problems. On the one hand, this means that the final set of heuristics

generates more accurate labels than other methods [6] since it does not produce low accuracy, high coverage heuristics. On the other hand, suppressed coverage may pose a problem, especially when a large amount of accurate training data is needed. Therefore, the Data-driven learner component, explained in the next subsection, tries to reverse the effect of abstaining while further refining the generated labels.

4.3.2.2. Data-driven Active Learner: Utilizing the Generative Model Output into AL Ranking

As mentioned earlier, the heuristics generator component outputs two outcomes: a final set of heuristics H, and a vector of initial probabilistic labels $\bar{y}_{initial}$. The heuristics set H can be denoted by a sparse matrix of weakly supervised labels as:

$$h_{i,j} = h_j(x_i) \text{ where } 1 \leq i \leq N, 1 \leq j \leq K \tag{4.4}$$

Conversely, the vector of probabilistic labels $\bar{y}_{initial}$ represents how confident the generative model is about the assigned labels. For example, if a data point did not receive a label from the heuristics set H, the generative model will have a probabilistic label for this point that is equal to $P[y_i = 1] = 0.5$, which represents an equal probability for the data point being of either class. Moreover, when the generative model assigns a probabilistic label $P[y_i = 1]$ close to 0.5 to a data point x_i, this indicates a point with low confidence labels, which may happen when many heuristics with similar accuracies disagree on the label for that data point. Formally, we define low confidence labels as:

$$|P[y_i = 1] - 0.5| \leq \alpha \tag{4.5}$$

where $P[y_i = 1]$ is the probabilistic label assigned by the generative model and α is a threshold to ensure that the definition of low confidence changes according to the number of the generated heuristics in H. α is denoted as:

$$\alpha = 0.3 - (1/e^{\sqrt{K+1}}) \tag{4.6}$$

where K is the number of heuristics generated in the first component. It is important to know that as the number of heuristics becomes larger, the value of α is expected to approach an initial value denoting fewer data points with low confidence labels (with $P[y_i=1]$ close to 0.5). Also, the formula specifies a value of 0.3 as the initial value before measuring the exponential decay as K

increases. The value is determined based on the insights obtained from the experiments as the value succeeds to capture the right range of low confidence labels with less number of heuristics. In other words, we expect to have fewer data points with low confidence labels when more heuristics are generated. Thus, The learner component uses the formula above to classify the data points in D_U into two groups namely, points with high confidence labels D_{HC} where $D_{HC} \subseteq D_U, \forall x_i \in D_{HC} \{x_i \mid |P[y_i = 1] - 0.5| > \alpha\}$, and points with low confidence labels D_{LC} with size L_c where $D_{LC} \subseteq D_U, \forall x_i \in D_{LC} \{x_i \mid |P[y_i = 1] - 0.5| \leq \alpha\}$. The learner component aims at eliminating the second group by replacing the low confidence labels with more accurate ones.

To accomplish such a goal, the component tries to integrate the user in the loop at this point by employing active learning. However, our problem settings do not impose traditional active learning scenarios where we usually have a small set of labeled points and a larger set of unlabeled data. Instead, we deal with a set of probabilistic labels that are classified based on the confidence of the generative model. Therefore, we adopt meta-active learning in this component and propose a data-driven approach to learn the query strategy. The approach formulates the process of designing the query strategy as a regression problem. We train a regression model to predict the reduction of the generalization error associated with adding a labeled point $\{x_i, y_i\}$ to the training data of a classifier. The idea of implementing machine learning to decide or develop the selection policies in active learning has been applied to many situations in the literature [29], [30], [31], and proved to achieve a competitive performance against traditional querying strategies. Therefore, our main hypothesis is that this regressor can serve as the query strategy in our problem settings to outperform the baseline strategies since it is customized to the underlying distribution and considers the output of the generative model.

Accordingly, the component consists of two main processes. First, *designing the AL query strategy* that fits the data distribution for a given problem. Second, applying it to the D_{LC} as a *Data-driven AL process*. The overall structure of this component is illustrated in Figure 4.3. Detailed descriptions about each of the two processes are given in the rest of this section.

Designing the data-driven strategy. Since the task of designing the AL query strategy is framed as a regression problem, the goal is to train a regressor that, when applied to a set of points, it chooses the point that results in the maximum reduction to the generalization error. To set up the regression process, we need a set of labeled observations to train the regressor. Formally, at this

point, we aim at creating a training dataset D_{reg} of size Q which can be described as $\{\gamma_i, \nabla_i\}_{i=1}^{Q}$ where γ_i represents a set of features describing the i^{th} observation in D_{reg}. In our implementation, we consider features that are specific to the data distribution and represent the state of the points in D_U. Therefore, we utilize the probabilistic label $P[y_x = 1]$ which is assigned to the point x by the generative model, the distance to the closest point in the dataset, and the distance to the nearest labeled point. Alternatively, ∇_i represents the label associated with the i^{th} point in D_{reg}, which is the potential reduction to the generalization error after annotating this point and adding it to the labeled dataset.

Creating the training data for the regressor. Therefore, to collect these observations and create D_{reg}, we design an experiment in which we iteratively train a classifier and evaluate it to record the corresponding generalization errors. In this scenario, we use both the labeled dataset D_L along with the points with the high confidence labels produced by the generative model D_{HC}. After combining

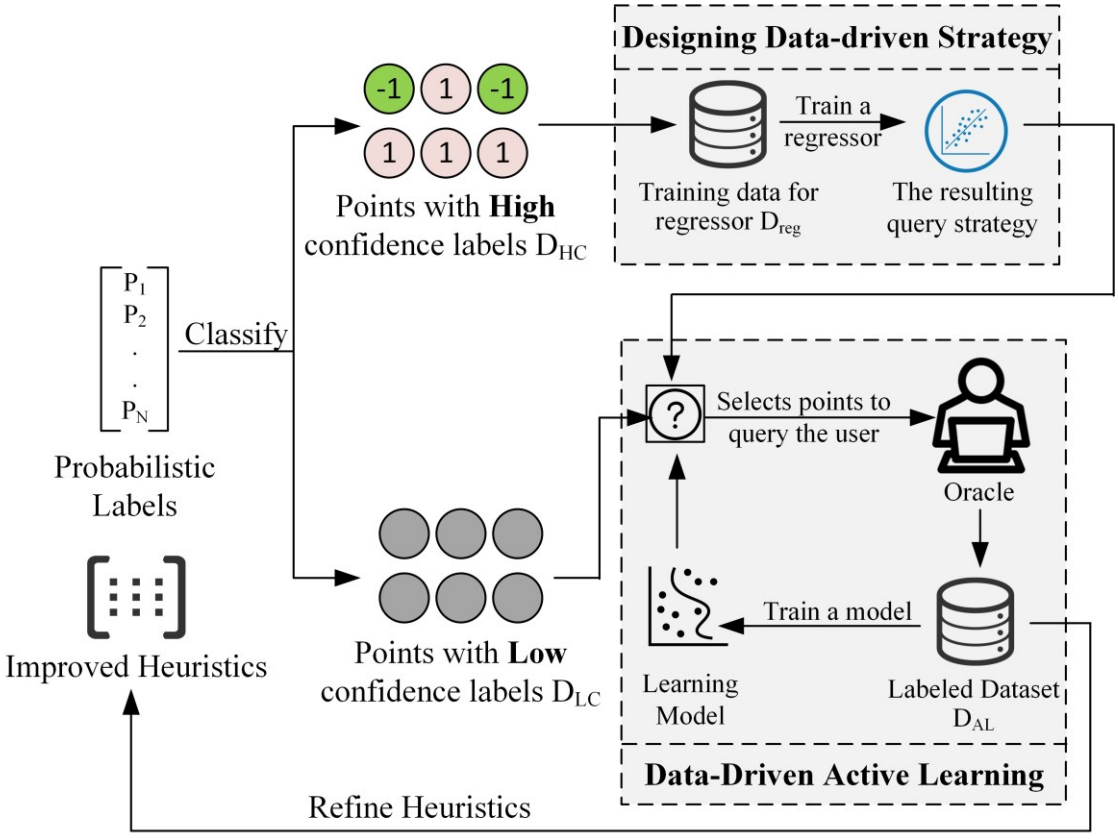


Figure 4.3: An overview of the data-driven active learner

these two datasets, we split them into a training set $D_{Train,}$ and testing set D_{Test}. Then, we split D_{Train} into a labeled dataset D_{Train_L} of size S and an unlabeled dataset D_{Train_U} consisting of the remaining points. Next, we train a classifier with D_{Train_L}, resulting in a model m_s that can be used to output class labels for the data point in D_{Test} and calculate the corresponding classification loss L_s using the test data D_{Test}. After that, we iteratively select a new point x from D_{Train_U} and add it to the labeled set as:

$$D_x = D_{Train\_L} \cup \{x\} \tag{4.7}$$

Then, we use D_x to train the classifier again and create a new model m_x, test the new model using D_{Test}, and calculate the new classification loss L_x. We then record the reduction in the classification loss associated with adding x to the training set as:

$$\nabla_x = L_s - L_x \tag{4.8}$$

Moreover, as we are recording the reduction in the generalization error ∇_x associated with labeling each point x, we compute the set of features γ_i that represent the point status in D_U. Finally, To construct the final dataset, we repeat this experiment using different initializations of D_{Train_L} with varying sizes of $S \in \{S_{min}, \dots, S_{max}\}$. Based on the insights obtained from the experiments (Section 4.4), we repeat the process with different sizes equal to 70%, 80%, and 90% of the total size of D_{Train}. Although S can be initialized with any range of sizes, we find these values result in enough data points to learn the query strategy while sustaining an acceptable computational cost.

Based on the insights obtained from the experiments, the process of training the regression function does not impose a high cost on the active learning process. The total time required for learning the AL strategy for a dataset of 11M records was less than 15 minutes on an Intel i7 machine with 32 GB RAM. Also, for training the regression function for more massive datasets, the user can always adjust the sampling range to reduce the number of iterations and speed the process. As for applying the query strategy, the component runs in $O(I. L_c)$ where I is the number of queries consumed by the AL component, and L_c is the size of the unlabeled pool D_{LC}. Consequently, during each iteration, we randomly sample S points from D_{Train} and record the characteristics γ_x of different data points along with their reductions γ_x to the generalization error. As a result, a new training dataset D_{reg} is created, which can be used to train the regressor.

Training the regressor. As the data-driven learner aims at enhancing the quality of the labels in D_{LC}, we use D_{reg} to train a regression function g to predict the potential error reduction of annotating the instances in D_{LC}. Although the distribution in D_{LC} is different from the distribution of D_U, at this point, we aim at creating an active learning strategy that considers the distribution of the unlabeled pool, which in this case, D_{LC}. Therefore, in our implementation, a random forest regressor is used and trained using D_{reg}. Although the regressor can be implemented using any regression function, random forest regression is applied since it maintains high accuracy for large high-dimensional data while preventing overfitting. Since the random forest regressor requires a set of meta parameters, we use cross-validation to define a grid of hyperparameter ranges and use it to choose the best model that reduces the chance of overfitting. Then, we treat the output model as our query strategy that is built while considering the outcome of the generative model. The resulting policy can be used to greedily select data points with the highest potential error reduction by taking the maximum of the value predicted by the regressor g as:

$$x^* = \arg\max_x \ g(\gamma_x) \tag{4.9}$$

Furthermore, the whole process is explained in Algorithm 2.

Data-driven AL process. The regression function g is then applied to rank the points in D_{LC}. Since the number of data points in D_{LC} is much smaller than the number of points in D_U ($L_c << N$), the ranking time is highly reduced. Moreover, to overcome the cold-start problem in active learning [37], the component uses the dataset with the high confidence labels D_{HC} along with the labeled dataset D_L to form the initial seed and the test set. The initial seed is used to train the classifier at the beginning of the active learning process, while the test set is used to evaluate the classier after each iteration [17]. The component also allows the user to specify a budget B for the maximum number of points that can be manually labeled using g. The output of this component is a labeled set D_{AL} that can be described as $\{\mathbf{x}_i, y_i^*\}_{i=1}^{\min(B,L_c)}$. In other words, D_{AL} represents the data points x in D_U that are selected by g to receive true labels y^* from the user. Since the active learning process only terminates when either all the points in D_{LC} receive a true label from the user or the labeling budget B is exceeded, the size of D_{AL} is denoted as min(B, L_C). Finally, the component uses this set to refine the sparse matrix H as:

$$H_{\text{updated } i,j} = \begin{cases} y_i^* & \text{if } (x_i, y_i) \in D_{AL} \\ H_{i,j} & \text{otherwise} \end{cases} \quad j = 1,2,\dots K \tag{4.10}$$

Algorithm 4.2: The Process of Designing the Data-driven Strategy

Input: The vector of probabilistic labels $\bar{y}_{initial}$, The labeled input dataset D_L

Output: A regressor model g

1: Use $\bar{y}_{initial}$ to classify D_U into D_{HC} and D_{LC}

2: initialize dataset $D = D_L \cup D_{HC}$

3: split D into D_{Train} and D_{Test}

4: **for** S in $\{S_{min}, \ldots S_{max}\}$ **do**

5: Split D_{Train} into D_{Train_L} of size S and D_{Train_U}

6: train a classification model m_S

7: calculate the test loss L_S

8: **for** each point x in D_{Train_U} **do**

9: form a new dataset $D_x = D_{Train\_L} \cup \{x\}$

10: train a classifier model m_x

11: calculate the new test loss L_x

12: calculate the error reduction $\nabla_x = L_s - L_x$

13: collect the data point parameters γ_x

14: return the labeled data point $\{\gamma_x, \nabla_x\}$

15: return D_{reg} of size Q as $\{\gamma_x, \nabla_x\}$

16: train and evaluate a random forest regressor g using D_{reg}

17: return g

The matrix $H_{updated}$ is used as an improved version of H. By utilizing these processes, a portion of the low confidence labels are replaced by true labels. As mentioned earlier, the low confidence points are originated when either the heuristics abstain from labeling or disagree on specific points. Therefore, the learner component enhances the quality of the labels by eliminating the abstaining effect and resolve the disagreements between the heuristics to increase their accuracies.

4.3.2.3. Probabilistic Labels Generator: Aggregating the Output of Different Heuristics

The final component of Asterisk is the label generator, which aims at learning the accuracies of the generated heuristics using the refined heuristics matrix $H_{updated}$, and then combines all the output of these heuristics to produce a single probabilistic label $\bar{y}_i$ for each point in D_U. This process is accomplished by learning the structure of a generative model Gen [12], which utilizes the refined matrix to model the process of labeling the training set. Since the generative model treats the final label Y^* as a latent variable, it learns the distribution over the labels generated by each heuristic in $H_{updated}$. To learn the accuracies of the refined heuristics and the correlations among them, the generative model maximizes the L_1 regularized marginal pseudolikelihood [12] of the output of each heuristic in $H_{updated}$. The process uses the agreements and disagreements between the refined heuristics to encode the generative model as a factor graph [38]. It employs three factors, which are labeling accuracy, labeling propensity, and the heuristics pairwise correlations. These factors formally define our model as [6]:

$$\text{Gen: } \pi_{\varphi}\left(H_{updated}, Y^*\right) = \frac{1}{Z_{\emptyset}} e^{\emptyset^T H_{updated} Y^*} \tag{4.11}$$

where Ø denotes the accuracy of the heuristics and represents the factor graph, and $Z_{Ø}$ is a partition function to ensure π is a distribution. As a result, the generative model Gen employs a distribution to describe the relationship between the heuristics $H_{updated}$ and the latent variable for the true label Y^*. Hence, after learning the relative accuracies of the heuristics, the generative model can estimate $P(H_{updated} \mid Y^*)$ by combining their output into a single label or each data point.

As Figure 4.2 depicts, the processes of updating the heuristics and generating the final probabilistic labels are iterative. Therefore, at this point, the user is informed about the performance of the final heuristics, the coverage obtained in D_U, the status of the generated probabilistic labels such as the number of low confidences labels, and the number of true labeled consumed so far. Then the user decides to either terminate the process or initiate another cycle to further refine the output labels. The output of the generative model can then be used to train any noise-aware discriminative model to generalize beyond the generated observations.

4.4. Evaluation

To evaluate the proposed method, we run a set of experiments to compare Asterisk to other labeling approaches. The experiments are three-fold and seek to validate the following claims:

- **Labels from Asterisk outperform labels produced by weak supervision sources that are automatically created.** We compare Asterisk to another approach that automatically generates weak supervision sources and produces training labels [13]. During the experiments, we consider both the accuracy of the generated labels (compared to the ground truth) and the performance of the end model. As for the accuracy of the generated labels, Asterisk outperforms this method by 14.84% on average (Labeling accuracy). Also, the experiments show that the proposed method improves the classification accuracy by 7.15% on average (End model Accuracy) when compared to this approach.
- **Labels generated from Asterisk outperform labels generated using user-defined weak supervision sources.** We compare Asterisk to other methods that allow users to express their domain experience in the form of labeling functions [6], [38]. The experimental results show that the proposed system outperforms these tools in labeling accuracy by 34.20% on average. The proposed method also enhances the learned accuracy of the generative models by 10.42% on average.
- **Labels generated from Asterisk outperform labels generated using active learning.** We compare Asterisk to baseline active learning techniques. The experiments consider four different active learning query strategies [7]. Asterisk enhances the classification accuracy by 4.15% on average while reducing the labeling cost by up to 52.77%.

Although the experimental evaluation considers a wide range of ten classification tasks and reports different evaluation metrics, the reader should not over infer from those ten samples. The results obtained from the experiments depict that the proposed method achieves competitive labeling results along with adequate classification performance. However, these samples cannot describe the entire sampling frame for this problem space; and hence, it is wrong to assume that the algorithm will provide significantly superior performance in every situation. Therefore, even though the proposed method outperforms both semi-supervised learning and weak supervision techniques, for other tasks and with different problem settings, other paradigms such as unsupervised learning models may result in more superior performance.

Table 4.1: Datasets statistics

Datasets	Domain	Data Size	F	+/ Size	M	N
Higgs	Physical	11,000,000	28	52.96	440,000	10,560,000
Renewal Sales	Business	1,354,704	11	73.06	54,188	1,300,516
Rain Prediction	Weather	142,000	24	22.42	7,100	134,900
Travel Insurance	Business	63,300	11	14.60	3,165	60,135
Bank	Business	45,211	17	11.70	2,261	42,950
News	Social	39,797	61	20.38	1,990	37,807
Credit Card	Business	30,000	24	22.12	1,500	28,500
Occupancy Detection	Physical	20,560	7	23.10	1,028	19,532
Magic	Physical	19,020	12	35.16	951	18,069
MNIST	Image	70,000	784	-	3,500	66,500

Data size is the number of records each dataset has. F is the number of features used to create the labels and train the final classifier. +/Size is the percentage of the positive class to the dataset size. M, N are the sizes of labeled and unlabeled datasets, respectively.

The section is divided into three subsections. In the first subsection, we discuss the experimental setup. Next, we report the results of comparing Asterisk to other labeling methods. Finally, in the third subsection, we evaluate the individual components of the proposed system by experimenting with two variations of Asterisk to assess the effect of each component on the final model performance.

5.4.1. Experimental Setup

The section describes the datasets used in the experiments, the baseline methods, and the implementation details.

4.4.1.1. Datasets

We consider real-world applications and tasks over open-source tabular datasets. Summary statistics are provided in Table 4.1. We examined classification tasks for various domains, including business, physical, social, and multiclass image classification.

Business. We use four business datasets in the experiments. First, we employ a real-world business dataset, **Renewal Sales,** that is collected from our industrial partner, IBM. The data contains more than 1.3 million records and is used to classify renewal risks in which clients decide not to renew their software licenses [39]. Second, we use the **Travel Insurance** dataset, which is collected from

a third-party travel insurance company that is based in Singapore. The dataset has more than 63K records and is used to detect insurance claims. Another business dataset is the **Bank** [40] dataset, which is a business dataset that contains more than 45K instances. It represents direct marketing campaigns using phone calls of a banking institution and is used in a classification task to predict if the client will subscribe to a term deposit. Finally, we include the **Credit Card** dataset [41], which is a business dataset of 30K records for customers' credit card payments. The classification task of this dataset is to predict the default payments.

Physical. The experiments incorporate three physical datasets, namely, **HIGGS**, **Occupancy Detection**, and **Magic**. **Higgs** is a large-scale dataset of 11M records and 28 features. It is used in a classification problem to distinguish between a signal process of Higgs bosons and a background process [42]. **Occupancy detection** [43] dataset is used for binary classification to determine if a room is occupied or not based on seven measurements, such as room temperature, humidity, and light. **Magic** [44] is a dataset of 19K records and 12 features that simulates the registration of high energy gamma particles in an atmospheric telescope. The classification target is to discriminate between photons that are caused by primary gammas and the images of hadronic showers.

Social. We also use the **Rain Prediction** dataset, which contains daily weather observations from numerous Australian weather stations. It has more than 140K records and 24 features to classify whether or not it will rain tomorrow. Another dataset is the **News** [45] dataset, which summarizes a set of 61 heterogeneous features of online articles. The dataset has more than 39K records and is used to classify the popularity of a given article.

Image. Finally, the experiments include an example of multiclass classifications tasks using the **MNIST** dataset [46]. The dataset contains 70K hand-written digits images with ten classes, from '0' to '9'.

4.4.1.2. Baseline Methods

We compare the proposed method to the following methods:

- **Data Programming** [2]: The experiments include the DP system [6], which requires users to write labeling functions to express arbitrary heuristics. Then, the system denoises their outputs without access to ground truth by incorporating the DP paradigm [12].

- **Automated weak supervision:** We incorporate an automated weak supervision approach (WS-Automatic) [13]. The method takes advantage of a small labeled dataset to automatically create weak heuristics and generate labels to an unlabeled dataset.
- **Hybrid approaches:** We compare the proposed method to a hybrid approach (DALP) [38]. Similar to DP, the hybrid method allows users to write labeling functions. Then, it applies active learning to enhance the accuracy of the generated labels.
- **Active Learning** [7]: We compare Asterisk to different active learning strategies [17], namely uncertainty sampling (UNC), query by committee (QBC), density-weighted uncertainty sampling (DWUNC), and random sampling (RAND).
- Also, to evaluate the individual components of the proposed system, we added two variations of the proposed system, namely **Asterisk-manual** and **Asterisk-AL**. In Asterisk-manual, the component of the automatic generation of heuristics is disabled (Section 4.3.2.1), and the system relies on the users to write labeling functions for each classification task. On the other hand, Asterisk -AL does not incorporate the data-driven active learner component (Section 4.3.2.2). Instead, it applies uncertainty sampling to rank the points in D_{LC}.

4.4.1.3. Implementation Details

Generating heuristics. To write the labeling functions used for DP [6], DALP [38], and Asterisk-manual, we implemented a set of threshold-based labeling functions [6], [13] in which the labeling functions rely on numerical thresholds to output a label for each data point or abstain. To create these labeling functions, a set of gold labels is required to evaluate the quality of these functions. To ensure fairness of the experiments, the same labeling budget B specified by the proposed method is used as gold labels. Also, as for WS-Automatic, the same set of gold labels is used to develop and evaluate the generated heuristics.

For the renewal sales dataset, we consulted a set of sales representatives from IBM to help us write seven labeling functions. As for the rest of the datasets, we applied pattern matching to decide on the threshold values. We consider using patterns matching techniques since we do not have access to domain experts. Also, we find the technique consistent with the best practice followed in the literature [6], [13], [20], [47].

Table 4.2: Settings for the user-defined labeling functions

Dataset	# Candidates	# Labeling Functions	Labeling Functions Performance	
			Acc	F1
Higgs	8800000	9	0.63	0.50
Renewal Sales	1239849	7	0.85	0.80
Rain Prediction	113755	11	0.87	0.75
Travel Insurance	50662	6	0.81	0.79
Bank	36,169	8	0.78	0.67
News	31,716	10	0.69	0.74
Credit Card	24000	7	0.73	0.75
Occupancy Detection	16,448	9	0.65	0.80
Magic	15216	8	0.82	0.73
MNIST	56,000	12	0.68	0.74

Moreover, to only append accurate sources, we calculated the empirical accuracy of the labeling functions using D_L and only included the functions that reported more than 60% accuracy. Table 4.2 shows the settings of the labeling functions used in the experiments, including the number of candidates (data points) for which labels are created, the number of labeling functions designed for each task, and the empirical evaluation metrics (Accuracy and F1 measure) calculated using D_L.

Active Learning Settings. When comparing the proposed method against different AL sampling techniques, we averaged the results over ten runs. For each classification task, the labeled dataset D_L was used as the initial seed. The unlabeled dataset D_U was split into an unlabeled pool and a separate held-out test set to evaluate the classifier after each iteration. Moreover, to decide when to terminate the active learning process, we examined the classifier performance and stopped when the classification accuracy does not show significant improvements with additional iterations [48], [49]. More specifically, we used a threshold λ for the differences between accuracy values achieved by the classifier. We stopped the process when the mean of these differences does not exceed λ for a successive number of iterations. While having a small value for λ can increase the annotation cost, choosing a larger value can result in missing useful generalizations [49]. Therefore, in earlier research [50], we experimented with a range of different values for λ to decide on the optimal value to stop the active learning process. We found that having $\lambda = 0.0001$ succeeds

in catching the elbow values in the learning curves, after which the performance does not notably change. Moreover, to keep all the conditions the same throughout the experiments, we report the number of queried points consumed by the active learning process and used it as the labeling budget B for the proposed method.

Performance Metrics. Since training models with accurate labels improve their capability to generalize to unseen points in the test data [2], [6], the experiments report the classification performance of the final models trained with the generated labels along with the labeling accuracy of the generated labels. Also, reporting only one evaluation metric has been proven to be not enough to judge machine learning models [51]. Therefore, we report a handful of evaluation metrics. As for the end models, we report Classification Accuracy (Acc), F1 measure (F1), and Matthews correlation coefficient (MCC). Classification accuracy represents the number of correct predictions made divided by the total number of predictions reported and is calculated as (TP+TN)/(TP+TN+FP+FN) where TP, TN, FP, and FN are the numbers of true positives, true negatives, false positives, and false negatives respectively. However, accuracy can be misleading for classifying imbalanced datasets. Since some of the datasets used in the experiments are imbalanced, we consider MCC [52] to describe the confusion matrix and the classifier performance. MCC is calculated as:

$$\mathrm{MCC} = \frac{TP*TN-FP*FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}} \tag{4.12}$$

for binary classifications and has been generalized to the multiclass situations [52]. Moreover, we report F1 measure as a harmonic mean of both precision and recall.

Also, since all the datasets used in the experiments contain ground truth labels, the experiments record the labeling accuracy achieved by the proposed method, WS-Automatic, data programming, and the hybrid approach DALP. To calculate the labeling accuracy, the ground truth labels are initially removed to create the unlabeled dataset D_U. Then, each of these approaches is used to generate labels for D_U. Later, the generated labels are compared to the ground truth to calculate the labeling accuracy. The labeling accuracy is measured as the ratio between the number of correctly labeled instances to the training set size.

Final Learning Models. Since the proposed method aims at generating labeled training datasets for any downstream model, we experimented with a wide range of classification algorithms to find the best model for each task. In the case of the renewal sales and the Higgs datasets, we used the

Table 4.3: Asterisk vs. automatic weak supervision approach, WS-Automatic

Dataset	The Asterisk System							Asterisk Improvement over WS-Automatic (%)						
	Heuristics Performance			End-Model Performance			Labeling Accuracy	Heuristics Performance			End-Model Performance			Labeling Accuracy
	Acc	F1	Cov.	ACC	MCC	F1	(%)	Acc	F1	Cov.	Acc	MCC	F1	(%)
Higgs	0.81	0.85	0.80	0.85	0.72	0.84	72.14	0.67	1.25	0.99	6.40	13.51	2.28	7.67
Renewal Sales	0.71	0.82	0.78	0.86	0.83	0.90	83.53	3.04	5.40	16.02	-2.55	20.88	50.28	43.08
Rain Prediction	0.73	0.85	0.81	0.85	0.73	0.80	85.25	7.13	5.24	-5.81	3.91	-9.54	0.48	7.91
T. Insurance	0.78	0.72	0.74	0.78	0.81	0.71	78.14	25.99	32.67	6.15	-17.55	18.94	0.25	20.22
Bank	0.97	0.92	0.87	0.87	0.86	0.82	93.67	13.74	27.41	4.99	6.21	6.19	12.38	6.13
News	0.92	0.91	0.88	0.96	0.92	0.90	96.57	13.67	21.55	14.71	20.91	18.31	73.07	15.16
Credit Card	0.96	0.89	0.91	0.92	0.93	0.95	87.75	10.53	13.98	10.96	18.52	21.08	26.05	12.72
O. Detection	0.90	0.92	0.88	0.95	0.93	0.90	98.91	15.88	24.03	15.37	20.73	16.56	20.20	20.51
Magic	0.82	0.74	0.81	0.90	0.91	0.93	89.73	7.82	34.74	17.43	9.13	25.08	15.38	6.69
MNIST	0.96	0.93	0.86	0.95	0.94	0.92	93.16	8.85	12.05	14.37	5.77	5.75	8.08	8.33

The performance of the generated heuristics (Accuracy (ACC), Coverage (Cov.) and F1 measure (F1)), the end model performance (Accuracy (ACC), MCC, and F1), and the accuracy of the generated labels (Labeling Accuracy) compared to the ground truth in each dataset.

gradient boosting algorithm XGBoost [53]. As for the bank, the rain prediction, and the travel insurance datasets, we applied random forest classifier since it has been evaluated as a reliable classifier in various classification tasks [54]. In the news and the MNIST datasets, classifiers were implemented using linear Support Vector Machine (SVM) classifier [55]. As for the credit card, magic, and occupancy detection tasks, we chose logistic regression. The principal motive for using this range of different classification algorithms is to demonstrate the resiliency of the proposed method to the classifier. Also, to show that, regardless of the choice of the classifier, the final model can generalize beyond the generated labels and produce predictions to the unseen points in the test set.

Handling imbalanced data. Moreover, some of the datasets used in the experiments are highly imbalanced, such as the renewal sales, the rain prediction, the travel insurance, and the bank datasets. To deal with such data, the experiments are designed to maintain an equal number of labeled samples for each class in the labeled dataset D_L. As a result, a balanced new dataset is retrieved and used to create the initial set of heuristics H. Then, the final learning models are tuned to learn with imbalanced data. For example, we adjust the *scale_pos_weight* parameter in the gradient boosting algorithm to control the balance of positive and negative weights before applying

XGBoost to the renewal sales and the Higgs datasets. As for the random forest classifier with the bank, the travel insurance, and the rain prediction datasets, we assign higher weights to the minority class and penalize the misclassification for instances of this class. Similarly, a class-weighted SVM algorithm is applied to deal with imbalanced data in the news dataset. Overall, the experimental results depict that the framework can handle imbalanced data and achieve up to 90.36% in classification performance in terms of F1 measure and 83.02% on average in terms of MCC (Table 4.3).

4.4.2. Experimental Results of End to End Systems

In this section, we compare the proposed method against WS-Automatic, DP, DALP, and various active learning strategies. The results obtained from the experiments demonstrate that classifiers trained with labels from the proposed system can achieve better performance (F1 measure) than WS-Automatic by 21% on average (Section 4.4.2.1) and outperform user-defined sources (DP and DALP) by 14.56% on average in F1 measure (Section 4.4.2.2). The results also show that, when compared to active learning, the proposed method can improve the performance of end models by 6.20% on average in F1 measure (Section 4.4.2.3).

4.4.2.1. Asterisk vs. Automatic Generation of Weak Supervision

This section compares Asterisk to the automatic weak supervision approach, WS-Automatic. Since both methods rely on using a small labeled dataset to generate labels for a bigger unlabeled dataset, we provide each method with the same set of labeled points of size M depicted in Table 4.1. The rest of the data points in each dataset was then used as the unlabeled dataset D_U. Then, each method was used to generate training labels for D_U and train the same classifier with these labels to create a final model. Table 4.3 shows the results of the proposed method, along with its improvements over WS-Automatic. As shown in the table, the experiments consider the performance of the final heuristics in terms of accuracy (ACC), F1 measure (F1), and the achieved coverage of the generated labels (Cov.). The table also presents the performance of the end models trained using the generated labels in terms of accuracy (ACC), MCC, and F1 measure, along with the accuracy of the generated labels (Labeling Accuracy).

The proposed model does not only rely on the automatic generation of weak sources; it also uses the data-driven learner to enhance the accuracy of the generated heuristics. Therefore, the

generative model learns more accurate heuristics. For instance, the results show that the proposed system creates more accurate heuristics than WS-Automatic in all the tasks with the maximum improvement in the travel insurance dataset with a 26% increase in accuracy. Also, the results show that the learner component helps in resolving situations of labeling abstaining, which results in enhancing the coverage of the training dataset by up to 17.43% in the magic dataset. In some datasets such as the credit card, the proposed method manages to create labels for more than 90% of the points in D_U. Also, in large-scale datasets such as the Higgs and the renewal sales, the proposed method creates heuristics that labeled 80% and 78% of D_U, respectively. In general, the results demonstrate how, for many tasks, using the component of data-driven learner helps in enhancing the quality and the coverage of the generated heuristics.

As for the classification performance, the end models in Asterisk perform better in most of the tasks. Although for some tasks, such as the renewal sales and the travel insurance datasets, WS-Automatic achieves higher accuracy than Asterisk, the accuracy metric could be misleading here due to the class imbalance in these datasets. Alternatively, the results show that Asterisk enhances the MCC values by 21% and 19% in the renewal sales and travel insurance tasks, respectively. Also, the proposed method maintains a better F1 measure throughout all datasets. Since training the models using accurate data improves their capabilities to generalize to unseen observations, this proves that the proposed method could enhance the quality of the training labels.

Finally, as to the labeling accuracy, the results show that Asterisk generates more accurate labels than WS-Automatic. In some datasets such as the news and the occupancy detection datasets, the proposed method achieves labeling accuracy more than 90%. Asterisk also improves the labeling accuracy when compared to WS-Automatic with the highest accuracy reported in the renewal sales dataset with a boost of 43.08%. Overall, the results demonstrate that, since the proposed method uses the learner component to provide true labels for a portion of the dataset, it manages to output more accurate labels and hence improves the performance of the final models.

4.4.2.2. Asterisk vs. User-defined Heuristics

We compare the proposed method to two labeling methods that rely on user-defined heuristics: DP [6] and DALP [38]. During the experiments, the two methods, along with Asterisk, were used to generate labels and perform the classification tasks using the ten datasets. Table 4.4 shows the

Table 4.4: Improvements of Asterisk over user-defined heuristics (DP and DALP)

Dataset	Asterisk Improvement over DP (%)						Asterisk Improvement over DALP (%)					
	Generative Model Performance		Final Model Performance			Labeling Accuracy (%)	Generative Model Performance		Final Model Performance			Labeling Accuracy (%)
	Acc	F1	Acc	MCC	F1		Acc	F1	Acc	MCC	F1	
Higgs	31.52	70.40	60.69	40.22	47.13	36.02	10.32	7.64	16.60	5.16	18.12	18.26
Renewal Sales	-1.25	2.95	7.62	7.21	11.89	22.84	-14.34	-8.95	-7.42	-8.34	6.96	-0.67
Rain Prediction	-12.04	13.66	9.17	3.20	24.71	9.49	-9.87	0.29	4.32	-7.25	5.49	4.53
T. Insurance	2.78	-11.37	6.82	20.72	22.53	11.79	-9.17	-13.06	-10.91	6.42	2.85	8.53
Bank	43.68	36.38	19.30	16.24	6.55	49.63	19.85	9.43	0.10	1.43	1.28	22.13
News	4.05	24.30	6.81	4.21	7.57	88.74	15.01	8.39	2.27	1.57	0.40	62.35
Credit Card	37.98	18.85	42.20	35.31	29.42	68.37	17.24	10.43	14.11	9.68	9.48	36.82
O. Detection	20.51	15.52	17.32	12.62	7.20	57.86	6.33	6.19	10.50	1.03	2.45	21.57
Magic	18.96	2.07	24.37	30.44	36.67	16.58	6.60	-6.21	1.60	11.35	31.29	9.42
MNIST	17.05	25.51	16.09	13.67	9.76	81.83	3.20	3.33	9.41	4.58	9.37	57.90

The performance measures reported by the generative models (Accuracy (ACC) and F1 measure (F1), the performance measures reported by the end models (Accuracy (ACC), MCC, and F1 measure (F1)), and the accuracy of the generated labels (Labeling Accuracy) compared to the ground truth.

improvement of the proposed method over DP and DALP. First, since all the three methods use generative models to produce probabilistic labels, the table shows the improvement of the proposed approach with regard to the performance metrics learned by the generative model (Accuracy (ACC) and F1 measure (F1)) over the two methods. The table also shows the improvements made by Asterisk for the performance of the end model (Accuracy (ACC), MCC, and F1 measure (F1)) along with the labeling accuracy.

The results show that, in most of the tasks, the proposed approach outperforms the other two methods in the generative model performance. For example, in the Higgs, the credit card, and the occupancy detection datasets, the proposed approach surpasses the other methods by significant margins. As for the accuracy learned by the generative model in the Higgs dataset, Asterisk improves the performance by 31.52% when compared to DP and 10.32% when compared to DALP. It also achieves the highest F1 measure among the three methods in these datasets. Generally, the results show that the proposed method sustains better results for the generative model in tasks where designing weak supervision sources is challenging. For example, for datasets like the news and MNIST dataset, writing the labeling functions is hard due to the large number of features that must be considered to write accurate heuristics. Also, for some other datasets such

as the magic and the Higgs datasets, a high level of domain experience is needed to design the labeling functions. In these situations, automatic creation of the heuristics can be beneficial to obtain a high-quality set of heuristics.

On the other hand, in datasets where domain experience is available (e.g., renewal sales), or the learning task is easy enough to facilitate designing the weak supervision sources (e.g., rain prediction), the proposed method is outperformed by the other approaches. For example, in the renewal sales dataset, DALP achieves the highest accuracy and F1 measure among the three approaches. Also, in the rain prediction dataset, DP surpasses both Asterisk and DALP in terms of accuracy by 13.69% and 2.47%, respectively. Nevertheless, although the proposed method worked with less accurate supervision sources, it enhanced the overall labeling accuracy in both renewal sales and the rain prediction tasks when compared to DP by 22.84% and 9.49%, respectively. It also outperformed DALP in the rain prediction task by 4.53% in labeling accuracy.

Moreover, when considering the end model performance, Asterisk outperforms DP in all the problems, with the most significant improvement in the credit card dataset with 40% in MCC. Also, when compared to DALP, the proposed method improves the performance of the final model in most of the tasks. Although DALP outperforms the proposed method in the renewal sales and the rain prediction datasets with 9.10% and 7.82% in MCC values, respectively, the proposed method maintains its superiority in the rest of the tasks with the highest enhancement in the magic dataset with 11.35% increase in MCC values. Moreover, the results show that except for the renewal sales dataset, the proposed method achieves the highest labeling accuracy in all the problems. It attains an average improvement in the labeling accuracy of 44.31% when compared to DP and 24.08% when compared to DALP. All in all, the results show that the proposed method can be a suitable solution to achieve a high level of labeling accuracy and classification performance, especially when designing supervision sources becomes expensive.

4.4.2.3. Asterisk vs. Active learning query strategies

In this section, we report the results obtained when comparing the proposed method with four sampling techniques. As mentioned before, the labeling budget B in the proposed method is determined in each task based on the labeling cost of the active learning process. Therefore, to set up the experiments, we first applied the four query strategies (UNC, DWUNC, QBC, and RAND)

Table 4.5: Asterisk vs. active learning

Dataset	The Asterisk System				Active Learning (UNC)					
	End-model Performance			L_C	End-model Performance			Lift % (F1)	AL Cost	Lower % (Labeling Cost)
	Acc	MCC	F1		Acc	MCC	F1			
Higgs	0.88	0.71	0.86	941,857	0.72	0.65	0.75	14.01	1,198,850	21.44
Renewal Sales	0.96	0.93	0.89	81,710	0.95	0.84	0.82	8.60	125,988	35.14
Rain Prediction	0.97	0.83	0.93	10,645	0.92	0.79	0.88	5.35	15,004	29.05
Travel Insurance	0.98	0.83	0.91	4,639	0.94	0.80	0.90	0.97	6,704	30.80
Bank	0.89	0.87	0.85	2,867	0.93	0.66	0.84	0.99	3,364	14.77
News	0.98	0.97	0.96	7,193	0.93	0.81	0.91	5.16	13,818	47.94
Credit Card	0.92	0.93	0.95	6,120	0.91	0.67	0.91	4.10	12,958	52.77
Occupancy Detection	0.99	0.98	0.93	7,521	0.93	0.74	0.86	8.00	11,855	36.56
Magic	0.94	0.91	0.93	1,739	0.96	0.82	0.83	12.01	2141	18.78
MNIST	0.95	0.94	0.92	2,173	0.92	0.84	0.89	2.77	3,472	37.41

The performance measures reported by the end models (Accuracy (ACC), MCC, and F1 measure (F1)), the labeling cost of the Asterisk system (L_c), and the number of labeled instances consumed by active learning (AL Cost).

to the ten datasets. Figure 4.4 shows the classification accuracy and the MCC values of the end models achieved by the four sampling strategies. As the figure shows, uncertainty sampling maintains the highest levels of accuracy through all the tasks. Therefore, we report the results of UNC against the proposed method in Table 4.5. The table shows the performance of the end models and the number of labeled points consumed by UNC to achieve the reported accuracy level (AL Cost). Since we use the value of AL cost as the labeling budget B, the table also reports the value of L_C, which represents the size of the unlabeled pool D_{LC} in the data-driven learner component. As mentioned before, D_{LC} represents the points with low confidence labels, and its size affects the size of D_{AL}. Therefore, the value of L_C is reported to demonstrate the cost of manual labeling in the proposed method.

The results show that Asterisk achieves a higher level of accuracy when compared to active learning in almost all of the datasets. Although UNC outperforms the proposed method in the bank and the magic datasets by 4.81% and 2.01% in accuracy respectively, for large-scale datasets such as the Higgs, the renewal sales, and the rain prediction, Asterisk manages to surpass UNC by 21.24%, 0.93%, and 4.68% respectively. Moreover, Asterisk also manages to enhance MCC values in all the datasets when compared to UNC, with an average of 17.28%. As for the F1 measure,

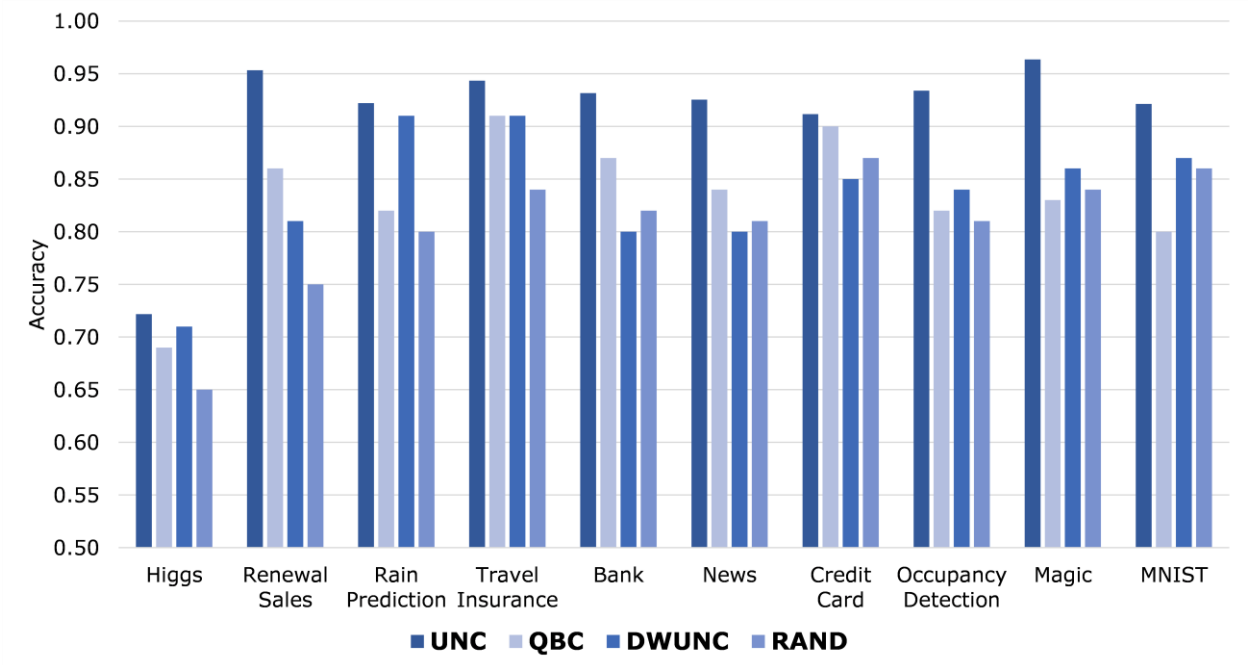
Asterisk also achieves higher values than UNC in all the problems with 6.20% enhancement on average.

Also, as shown in the table, active learning consumed more labels than the proposed method in all the tasks. The values of L_C shown in the table demonstrate that the size of D_{LC} remained less than the value of B through all the experiments. Hence, the data-driven learner in the proposed method stopped when there were no more points with low confidence labels to resolve in D_{LC} instead of exceeding the labeling budget B. As a result, Asterisk reduces the labeling cost in all the problems with the highest reduction of 53% in the credit card task. In short, the superiority of the proposed method over the AL process can be traced to two main reasons. First, the data-driven learner component in the proposed method starts with a larger seed since it employs both the data points with high confidence D_{HC} and D_L to form the seed. As a result, this enhances the initial accuracy of the end model and reduces the labeling cost. Second, the size of the unlabelled pool D_{LC} is much smaller than the size of the unlabeled pool used in the baseline active learning strategies as D_{LC} only represents the points with low confidence labels rather than the entire instance space. As a result, this helps reduce the labeling cost since it makes the data-driven learner converge faster.

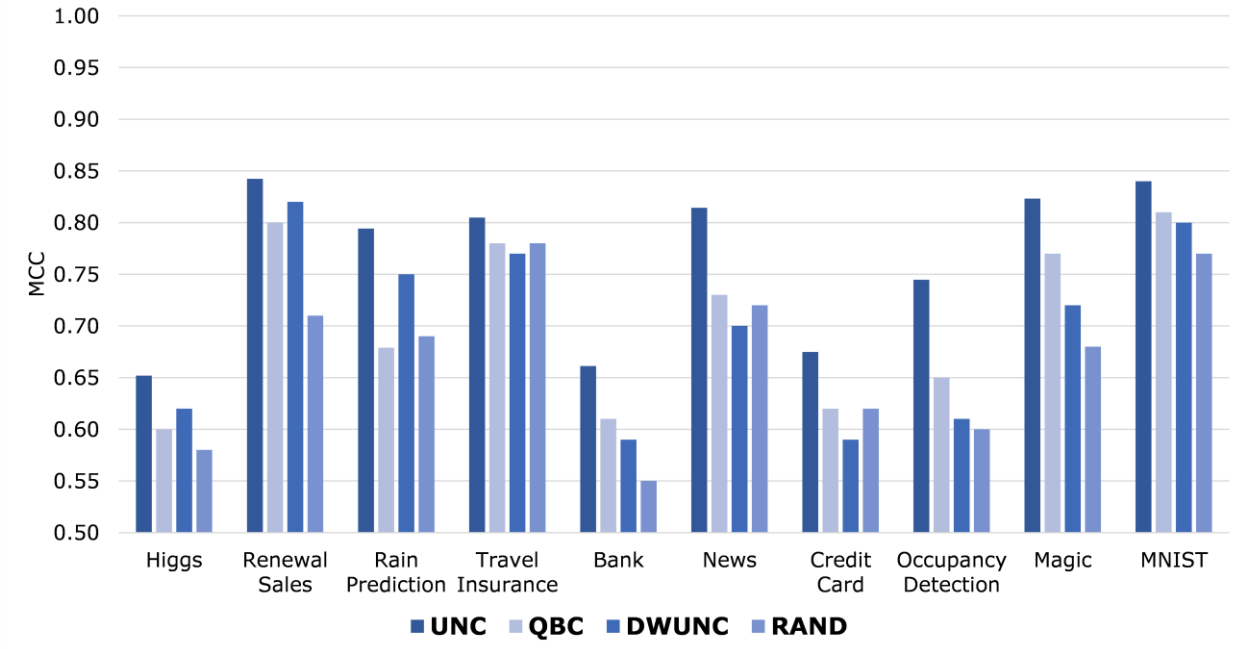
4.4.3. Experimental Results of Micro-Benchmarking

In this part, we assess the effect of the individual components of the proposed method, more specifically, the heuristics generator component and the data-driven learner component. The section shows that disabling either component can deteriorate the classification accuracy by up to 10.07% and decline the labeling accuracy by up to 61.26%. To create the Asterisk-manual modified version, we disable the automatic creation of the weak heuristics. Instead, we use the same labeling functions used for the approaches of user-defined heuristics (Table 4.2). Alternatively, in Asterisk-AL, the data-driven learner component is replaced by uncertainty sampling to choose the most informative points in D_{LC} for which true labels are provided. The results obtained using both versions to generate labels for the ten tasks are illustrated in Table 4.6. The table shows that, for each modified version, the evaluation metrics for the end model along with the labeling accuracy.

The results show that, with regards to Asterisk-Manual, disabling the heuristics generator component negatively affects the performance of the end models in most of the tasks, with the



(a)



(b)

Figure 4.4: Performance of end models in active learning experiments for (a) Classification accuracy and (b) MCC

highest reduction in the credit card dataset with 10.07% decrease in accuracy. However, in the datasets where user-defined labeling functions are more accurate than the generated heuristics,

Table 4.6: Performance of Asterisk-Manual and Asterisk-AL

Dataset	Asterisk-Manual				Asterisk-AL			
	End-model Performance			Labeling Accuracy	End-model Performance			Labeling Accuracy
	Acc	MCC	F1	(%)	Acc	MCC	F1	(%)
Higgs	0.78	0.69	0.76	72.08	0.85	0.69	0.80	61.34
Renewal Sales	0.96	0.93	0.89	83.19	0.79	0.77	0.87	83.10
Rain Prediction	0.83	0.80	0.77	84.15	0.83	0.69	0.77	81.55
Travel Insurance	0.89	0.79	0.71	75.26	0.71	0.76	0.69	72.14
Bank	0.91	0.85	0.83	81.35	0.88	0.87	0.84	76.71
News	0.95	0.93	0.94	78.15	0.95	0.93	0.92	56.60
Credit Card	0.83	0.90	0.91	41.23	0.87	0.89	0.91	34.31
Occupancy Detection	0.88	0.94	0.90	72.26	0.91	0.94	0.90	68.30
Magic	0.91	0.86	0.73	86.39	0.91	0.86	0.84	82.13
MNIST	0.91	0.91	0.86	67.45	0.89	0.92	0.88	59.04

Performance reported by the end models (Accuracy (ACC), MCC, and F1 measure (F1)) and the accuracy of the generated labels (Labeling Accuracy) compared to the ground truth.

Asterisk-Manual improves the performance of end models. For example, in the renewal sales and the rain prediction datasets, Asterisk-Manual enhances the classification accuracy by 11.76% and 9.36%, respectively, in MCC values. However, Asterisk-Manual achieves less labeling accuracy than the proposed method in all the problems with a maximum reduction of 53.02% in the credit card dataset. Overall, the results empirically posit that the heuristics generator component enhances the overall classification accuracy and the accuracy of the generated labels.

Alternatively, disabling the data learner component decreases the classification performance in all datasets. As the results show, Asterisk-AL achieved less accuracy in all datasets with an average of 5.10% decrease in accuracy, 3.94% decrease in MCC, and 4.54% decrease in F1 measure. Moreover, since the active learning part does not incorporate the outcome of the generative model in deciding on the points for which correct labels should be provided, this limits the capability of the proposed method to enhance the accuracy of the generated labels. Thus, Asterisk-AL achieved less labeling accuracy than Asterisk in all the problems with the highest decrease in the credit card dataset. Finally, the results empirically show the importance of the data learner component in enhancing the labeling accuracy and achieving better classification performance.

4.5. Related Work

The scarcity of labeled training data has been an abiding problem for machine learning developers and data scientists, which has motivated researchers to explore different labeling techniques. Therefore, in this section, we provide an overview of the methods that aim at automating the process of generating training labels.

Generating Noisy labels. Previous research [56]–[58] utilized weak supervision sources to provide high-level supervision in the form of noisy labels for massive datasets. For example, one framework [56] formulates the process of aggregating different weak supervision sources as a matrix completion problem for multi-task learning. Another work [57] focuses on the multiple instance learning paradigm and proposes a system that casts weak labels as an optimization scheme to identify the most discriminative instances. Also, Stewart and Ermon [58] introduce an approach to supervise machine learning models with weak supervisions sources by specifying constraints that hold over the output space.

Nevertheless, one of these approaches [56] is specific to multi-task weak supervision settings where diverse labeling sources have different granularities and related to sub-tasks of a problem. Alternatively, our settings are different since we have a set of weak supervision sources solving the same task; and hence, abstain, overlap, and conflict. Also, unlike our proposed method, other research [57], [58] focuses on applying weak supervision with specific models. For example, one of these techniques [57] aims at improving the predictive accuracy of Latent SVM for image and text classification tasks. Likewise, another approach [58] tries to enhance the capability of neural network models to handle weakly labeled datasets.

Furthermore, other research [8], [9], [59] has applied weak supervision to generate massive labeled datasets. However, unlike Asterisk, most of these approaches are only applicable to specific domains. For example, Gurjar *et al.* [8] introduce an approach to retrain the high performance of convolutional neural networks with weak supervision for tasks of handwriting recognition. Similarly, Chaidaroon *et al.* [9] apply an unsupervised method to extract weak signals from training data and leverage these signals for text hashing. Cao *et al.* [59] provide an end-to-end solution to the pattern classification problem in medical imaging.

Combining Noisy Signals. Moreover, there is ample research [2], [6], [60] that focuses on using generative models to aggregate weak supervision sources without the use of labeled data. The success of these approaches heavily relies on the quality of the labeling functions the users encode [22], which can be problematic, especially when designing such sources requires a high level of domain experience [13], [22]. Moreover, although the concept of automating the weak supervision sources have been studied in the literature [13], [22], none of these approaches [2], [6], [13], [22], [60] employ any domain experience when denoising the weak supervision sources which makes it challenging to estimate the coverage and the accuracy of the generated labels.

Optimizing Annotation Cost and Labeling Quality. Similar to weak supervision, other techniques [32], [37], [61], [62] have been proposed to provide solutions to the increasing demand for large-scale, high-quality labeled data. For example, some research [37] aims to formalize the user strategies for selecting data points in the active learning process. The study concludes that user-centered strategies can be beneficial in the early phases of the labeling process to resolve the cold-start problem in active learning. Although their findings [37] are consistent with our proposed system, Asterisk tries to mitigate the bootstrap problem by leveraging weak supervision sources in the beginning. Li *et al.* [61] propose a new active learning method that learns associations from deep neural networks to enhance the batch mode in AL. Also, as for meta-learning, other research [62] introduces a model that uses meta-learning to learn active learning strategies. However, in contrast to Asterisk, most active learning algorithms [32], [61], [62] have been validated on small and medium-size datasets. For example, one of these techniques [62] was validated on two datasets with a maximum size of 5,000 training samples and 1,000 test samples. Also, another algorithm [61] was tested using two datasets varying in size with a maximum of 70,000 records.

Combining Weak Supervision and Active Learning. Other research [63]–[65] provides various labeling solutions that combine active learning with weak supervision [63], [64], and transfer learning [65]. While research [63] explores using active learning and weak supervision as an integrated solution to debug machine learning models, Carbonneau *et al.* [64] combine active learning with weakly labeled data to reduce the annotation cost. Nevertheless, none of these techniques have tried to employ domain experience to denoise the weak sources. On the other hand, Zhou *et al.* [65] employs a pre-trained convolutional neural network and gradually fine-tunes it using active learning. Although the method aims at reducing the annotation cost, unlike the

proposed method, it is specific to the application of convolutional neural networks and is only evaluated within the domain of biomedical imaging.

4.6. Conclusions

The chapter presents Asterisk, a framework for generating high-quality labeled datasets at scale. The technique employs an iterative process to automatically generate high accuracy heuristics to assign initial labels. Then, it applies a data-driven active learning process to further enhance the quality of the generated heuristics. The process learns the active learning strategy while considering the modeled accuracies of the produced heuristics and the noise in the generated labels. The framework applies the learned strategy to economically engage the user and enhance the quality of the generated labels. We evaluate the proposed framework by comparing its performance with other weak supervision techniques such as data programming and automated weak supervision, along with active learning strategies. The empirical results show that the proposed framework can significantly enhance the learned accuracy of the generated heuristics by up to 44%, while producing high coverage labels for up to 91% of the unlabeled dataset. Also, comparing to the weak supervision techniques, the results show that the framework improves the quality of the generated labels by 28% on average. As well, the framework can reduce the annotation effort by up to 53% when compared to the baseline active learning strategies.

References

[1] W. Zhao, G. Guan, L. Chen, X. He, D. Cai, B. Wang, and Q. Wang, "Weakly-Supervised Deep Embedding for Product Review Sentiment Analysis," *IEEE Trans. Knowl. Data Eng*., vol. 30, no. 1, pp. 185–197, 2018.

[2] A. J. Ratner, C. M. De Sa, S. Wu, D. Selsam, and C. Ré, "Data Programming: Creating Large Training Sets, Quickly," *Advances in Neural Information Processing Systems*, pp. 3567-3575, 2016.

[3] V. S. Sheng, J. Zhang, B. Gu, and X. Wu, "Majority Voting and Pairing with Multiple Noisy Labeling," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 7, pp. 1355–1368, 2019.

[4] P. Cheng, X. Lian, X. Jian, and L. Chen, "FROG: A Fast and Reliable Crowdsourcing Framework," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 5, pp. 894–908, 2019.

[5] C. De Sa, A. Ratner, C. Ré, J. Shin, F. Wang, S. Wu, and C. Zhang, "DeepDive: Declarative

Knowledge Base Construction," *SIGMOD Rec.*, vol. 45, no. 1, pp. 60–67, 2016.

[6] A. Ratner, S. H. Bach, H. Ehrenberg, J. Fries, S. Wu, and C. Ré, "Snorkel: rapid training data creation with weak supervision," *Proc. VLDB Endow.*, vol. 11, no. 3, pp. 269–282, 2017.

[7] Y. Fu, X. Zhu, and B. Li, "A survey on instance selection for active learning," *Knowl. Inf. Syst.*, vol. 35, no. 2, pp. 249–283, 2013.

[8] N. Gurjar, S. Sudholt, and G. A. Fink, "Learning Deep Representations for Word Spotting under Weak Supervision," *International Workshop on Document Analysis Systems*, pp. 7-12, 2018.

[9] S. Chaidaroon, T. Ebesu, and Y. Fang, "Deep Semantic Text Hashing with Weak Supervision," *ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1109-1112, 2018.

[10] A. H. Akbarnejad and M. S. Baghshah, "An Efficient Semi-Supervised Multi-label Classifier Capable of Handling Missing Labels," *IEEE Trans. Knowl. Data Eng.*, vol. 31, pp. 229–242, 2019.

[11] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved Techniques for Training GANs," *Advances in neural information processing systems*, pp. 2234-2242, 2016.

[12] S. H. Bach, B. He, A. Ratner, and C. Ré, "Learning the Structure of Generative Models without Labeled Data," *Proc. the 34th International Conference on Machine Learning*, pp. 273-282, 2017.

[13] P. Varma and C. Ré, "Snuba: automating weak supervision to label training data," *Proc. VLDB Endow.*, pp. 223–236, 2018.

[14] E.-C. Huang, H.-K. Pao, and Y.-J. Lee, "Big active learning," *IEEE International Conference on Big Data*, Boston, MA, USA, pp. 94-101, 2017.

[15] Z.-H. Zhou, "A brief introduction to weakly supervised learning," *National Science Review*, vol. 5, no. 1, pp. 44–53, 2017.

[16] M.-F. Balcan, S. Hanneke, and J. W. Vaughan, "The true sample complexity of active learning," *Machine learning*, vol. 80, no. no. 2-3, pp. 111-139, 2010.

[17] B. Settles, "Active Learning Literature Survey," 2009.

[18] N. Dalvi, A. Dasgupta, R. Kumar, and V. Rastogi, "Aggregating Crowdsourced Binary

Ratings," *Proc. International Conference on World Wide Web*, pp. 285-294, 2013.

[19] M. Joglekar, H. Garcia-Molina, and A. Parameswaran, "Comprehensive and reliable crowd assessment algorithms," *IEEE International Conference on Data Engineering*, pp. 195-206, 2015.

[20] P. Varma, D. Iter, C. De Sa, and C. Ré, "Flipper: A Systematic Approach to Debugging Training Sets," *Proc. the 2nd Workshop on Human-In-the-Loop Data Analytics*, pp. 1–5, New York, USA, 2017.

[21] P. Varma, B. He, D. Iter, P. Xu, R. Yu, C. De Sa, C. Ré, "Socratic Learning: Augmenting Generative Models to Incorporate Latent Subsets in Training Data," *ArXiv161008123 Cs Stat*, 2016.

[22] N. Das, S. Chaba, S. Gandhi, D. H. Chau, and X. Chu, "GOGGLES: Automatic Training Data Generation with Affinity Coding," *ArXiv190304552 Cs*, 2019.

[23] J. Zhu, H. Wang, B. K. Tsou, and M. Ma, "Active Learning with Sampling by Uncertainty and Density for Data Annotations," *IEEE Trans. Audio Speech Lang. Process.*, vol. 18, no. 6, 2010.

[24] R. B. C. Prudencio and T. B. Ludermir, "Active Meta-Learning with Uncertainty Sampling and Outlier Detection," *IEEE International Joint Conference on Neural Networks*, pp. 346-351, 2008.

[25] K. Konyushkova, R. Sznitman, and P. Fua, "Introducing Geometry in Active Learning for Image Segmentation," *ArXiv150804955 Cs*, 2015.

[26] Y. Yang, Z. Ma, F. Nie, X. Chang, and A. G. Hauptmann, "Multi-Class Active Learning by Uncertainty Sampling with Diversity Maximization," *International Journal of Computer Vision*, vol. 113, no. 2, pp.113-127, 2015.

[27] M. Liu, W. Buntine, and G. Haffari, "Learning How to Actively Learn: A Deep Imitation Learning Approach," *Proc. Annual Meeting of the Association for Computational Linguistics*, vol. 1, pp. 1874-1883, 2018.

[28] M. E. Ramirez-Loaiza, M. Sharma, G. Kumar, and M. Bilgic, "Active learning: an empirical study of common baselines," Data Min. Knowl. Discov., vol. 31, no. 2, pp. 287–313, 2017.

[29] M. Fang, Y. Li, and T. Cohn, "Learning how to Active Learn: A Deep Reinforcement Learning Approach," *ArXiv170802383 Cs*, Aug. 2017.

[30] K. Konyushkova, R. Sznitman, and P. Fua, "Learning Active Learning from Data,"

Advances in Neural Information Processing Systems, 2017.

[31] H. Chu and H. Lin, "Can Active Learning Experience be Transferred?," *IEEE International Conference on Data Mining*, pp. 841-846, 2016.

[32] K. Pang, M. Dong, Y. Wu, and T. Hospedales, "Meta-learning transferable active learning policies by deep reinforcement learning," *ArXiv Prepr. ArXiv180604798*, 2018.

[33] A. Niculescu-Mizil and R. Caruana, "Predicting Good Probabilities with Supervised Learning," *Proc. International Conference on Machine Learning*, pp. 625-632, 2005.

[34] O. Sagi and L. Rokach, "Ensemble learning: A survey," *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.*, vol. 8, no. 4, 2018.

[35] B. Desharnais, F. Camirand-Lemyre, P. Mireault, and C. D. Skinner, "Determination of Confidence Intervals in Non-normal Data: Application of the Bootstrap to Cocaine Concentration in Femoral Blood," *J. Anal. Toxicol.*, vol. 39, no. 2, pp. 113-117, 2015.

[36] R. Xia, Y. Pan, H. Lai, C. Liu, and S. Yan, "Supervised Hashing for Image Retrieval via Image Representation Learning," *AAAI Conference on Artificial Intelligence*, pp. 2156–2162, 2014.

[37] J. Bernard, M. Zeppelzauer, M. Lehmann, M. Müller, and M. Sedlmair, "Towards User-Centered Active Learning Algorithms," *Comput. Graph. Forum*, vol. 37, no. 3, pp. 121-132. 2018.

[38] M. Nashaat, A. Ghosh, J. Miller, S. Quader, C. Marston, J. F. Puget, "Hybridization of Active Learning and Data Programming for Labeling Large Industrial Datasets," *IEEE International Conference on Big Data*, pp. pp. 46-55, 2018.

[39] M. Nashaat, A. Ghosh, J. Miller, S. Quader, and C. Marston, "M-Lean: An end-to-end development framework for predictive models in B2B scenarios," *Inf. Softw. Technol.*, vol. 113, pp. 131–145, 2019.

[40] S. Moro, P. Cortez, and P. Rita, "A data-driven approach to predict the success of bank telemarketing," *Decis. Support Syst.*, vol. 62, pp.22-3, 2014.

[41] I.-C. Yeh and C. Lien, "The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients," *Expert Syst. Appl.*, vol. 36, no. 2, 2009.

[42] P. Baldi, P. Sadowski, D. Whiteson, "Searching for exotic particles in high-energy physics with deep learning." *Nature communications*, 2014.

[43] L. M. Candanedo and V. Feldheim, "Accurate occupancy detection of an office room from light, temperature, humidity and CO 2 measurements using statistical learning models," *Energy Build.*, vol. 112, pp. 28–39, 2016.

[44] R.K. Bock, A. Chilingarian, M. Gaug, F. Hakl, T. Hengstebeck, M. Jiřina, J. Klaschka, E. Kotrč, P. Savický, S. Towers, A. Vaiciulis, and W. Wittek, "Methods for multidimensional event classification: a case study using images from a Cherenkov gamma-ray telescope," *Nucl. Instrum. Methods Phys. Res. Sect. Accel. Spectrometers Detect. Assoc. Equip.*, vol. 516, no. 2, pp. 511–528, 2004.

[45] K. Fernandes, P. Vinagre, and P. Cortez, "A Proactive Intelligent Decision Support System for Predicting the Popularity of Online News," *Conference on Artificial Intelligence,* pp. 535–546, 2015.

[46] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms," *ArXiv170807747 Cs Stat*, 2017.

[47] P. Varma, B. He, P. Bajaj, I. Banerjee, N. Khandwala, D. L. Rubin, and C. Ré, "Inferring Generative Model Structure with Static Analysis," *ArXiv170902477 Cs Stat*, 2017.

[48] G. Beatty, E. Kochis, and M. Bloodgood, "The Use of Unlabeled Data Versus Labeled Data for Stopping Active Learning for Text Classification," *IEEE International Conference on Semantic Computing*, Newport Beach, CA, USA, pp. 287-294, 2019.

[49] M. Bloodgood and K. Vijay-Shanker, "A method for stopping active learning based on stabilizing predictions and the need for user-adjustable stopping," *Proc. the 13th Conference on Computational Natural Language Learning*, Boulder, Colorado, pp. 39-47, 2009.

[50] M. Nashaat, A. Ghosh, J. Miller, and S. Quader, "WeSAL: Applying Active Supervision to Find High-quality Labels at Industrial Scale," *the Hawaii International Conference on System Sciences*, submitted for publication.

[51] A. C. Tan and D. Gilbert, "An Empirical Comparison of Supervised Machine Learning Techniques in Bioinformatics," *Proc. Conference on Bioinformatics*, vol. 19, pp. 219-222, 2003.

[52] D. M. Powers, "Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation," *Journal of Machine Learning Technologies*, 2011.

[53] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," Proc. ACM SIGKDD, San Francisco, CA, USA, 2016, pp. 785-794.

[54] I. Teinemaa, M. Dumas, F. M. Maggi, and C. Di Francescomarino, "Predictive business process monitoring with structured and unstructured data," *International Conference on Business Process Management*, pp 401-417, 2016.

[55] J. Kremer, K. Steenstrup Pedersen, and C. Igel, "Active learning with support vector machines," *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.*, vol. 4, no. 4, pp. 313–326, 2014.

[56] A. Ratner, B. Hancock, J. Dunnmon, R. Goldman, and C. Ré, "Snorkel MeTaL: Weak Supervision for Multi-Task Learning," *Proc. the 2nd Workshop on Data Management for End-To-End Machine Learning*, Houston, TX, USA, 2018.

[57] T. Durand, N. Thome, and M. Cord, "SyMIL: MinMax Latent SVM for Weakly Labeled Data," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 12, pp. 6099–6112, 2018.

[58] R. Stewart and S. Ermon, "Label-Free Supervision of Neural Networks with Physics and Domain Knowledge," *AAAI Conference on Artificial Intelligence*, pp. 2576- 2582, 2017.

[59] L. Cao, W. Tao, S. An, J. Jin, Y. Yan, X. Liu, W. Ge, A. Sah, L. Battle, J. Sun, R. Chang, B. Westover, S. Madden, and M. Stonebraker, "Smile: A System to Support Machine Learning on EEG Data at Scale," *Proc. VLDB Endow.*, vol. 12, no. 12, pp. 2230-2241, 2019.

[60] S. Wu, L. Hsiao, X. Cheng, B. Hancock, T. Rekatsinas, P. Levis, C. Re, "Fonduer: Knowledge Base Construction from Richly Formatted Data," *Proc. International Conference on Management of Data*, pp. 1301–1316, 2018.

[61] Y. Li, Y. l Wang, D. Yu, Y. Ning, P. Hu, and R. Zhao, "ASCENT: Active Supervision for Semi-supervised Learning," *IEEE Trans. Knowl. Data Eng.*, 2019.

[62] P. Bachman, A. Sordoni, and A. Trischler, "Learning Algorithms for Active Learning," *Proc. International Conference on Machine Learning*, vol. 70, pp. 301-310, 2017.

[63] D. Kang, D. Raghavan, P. Bailis, and M. Zaharia, "Model assertions for debugging machine learning," *NeurIPS MLSys Workshop*, 2018.

[64] M. Carbonneau, E. Granger, and G. Gagnon, "Bag-Level Aggregation for Multiple-Instance Active Learning in Instance Classification Problems," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 5, pp. 1441-1451, 2019.

[65] Z. Zhou, J. Y. Shin, S. R. Gurudu, M. B. Gotway, and J. Liang, "AFT*: Integrating Active Learning and Transfer Learning to Reduce Annotation Efforts," *ArXiv180200912 Cs Stat*, 2018.

Chapter 5 : Semi-Supervised Ensemble Learning for Dealing with Inaccurate and Incomplete Supervision

5.1. Introduction

Supervised learning refers to the task of inducing a learning function from a set of labeled examples so the function can map between the input (features) and the output (target label) in these training examples. After training, the created model should be able to generalize and correctly predict class labels for unseen data points. Therefore, supervised learning algorithms require large sets of noise-free labeled data to train their models since using data points with noisy or missing class labels can produce distorted models that lead to incorrect predictions [1]. However, obtaining these ideal datasets forms a challenge in most real-world applications. Due to the considerable cost of manual labeling, acquiring fully labeled datasets can be difficult, economically infeasible, or even impossible [1]. Also, since obtaining hand-labeled training data can be prohibitively expensive, practitioners tend to rely on weak supervision [2] to collect labeled datasets. However, low-cost approaches, such as crowdsourcing [3] and user-defined heuristics [2], produce low-quality annotated data with label noise.

Therefore, many techniques [4], [5], [6] have been proposed to enable learning algorithms to work with weakly supervised datasets. In this research, we focus on two types of weak supervision, which are inaccurate supervision and incomplete supervision. Inaccurate supervision refers to situations in which a portion of the provided examples are incorrectly labeled. The problem of learning with inaccurate supervision is also known under different names such as "learning with class noise" and "learning from mislabeled examples" [7]. On the other hand, in incomplete supervision, only a subset of the training data is provided with labels while the rest are unlabeled. Hence, the amount of given labeled examples are not enough to produce adequate classifier.

Although prior research [4], [5], [6], [8] treats those two types as two separate problems, in real-world applications, they often occur simultaneously. To deal with inaccurate supervision, many approaches [4], [9], [10] focus on creating a clean version of training data by identifying and removing instances with class noise; subsequently, a classification model is built using this clean

dataset. However, eliminating noisy instances can have a negative impact since these instances may contain useful information for the model. Moreover, most of these approaches [4], [9] apply a simple threshold to decide for each instance whether it should be considered as noise or not. Deciding on this threshold can be challenging, especially when there are a lot of misclassified points. Alternatively, other methods [8] try to modify existing algorithms to create learners that are more robust to class noise. However, some research [11] states that these approaches may not be effective when the noise level becomes relatively significant.

Alternatively, there are many techniques [5], [6] proposed to deal with situations of incomplete supervision. Some of these approaches [2], [5], [12] utilize semi-supervised learning (SSL) techniques to exploit unlabeled data without any human intervention. These approaches make assumptions about the underlying data distribution, such as its dimensional structure and smoothness. Many SSL techniques [5], [12] utilize the concept of generative models to estimate the probability that a given data point belongs to each class. Alternatively, active learning (AL) [6] is a special kind of semi-supervised learning which aims to achieve a satisfactory level of accuracy with minimal annotation cost. In AL, a human oracle is asked to provide labels for the most valuable unlabeled points. The selection of these valuable points is made by a query strategy [6], which is an algorithm that measures the informativeness of the data points and ranks them accordingly. However, several questions regarding these techniques remain to be addressed. For example, semi-supervised approaches that depend on learning a generative model have scalability problems when dealing with complex dependency structures [13]. Also, AL can be expensive with high-dimensional datasets in which the ranking process can be time-consuming, especially when the number of unlabeled points is significant.

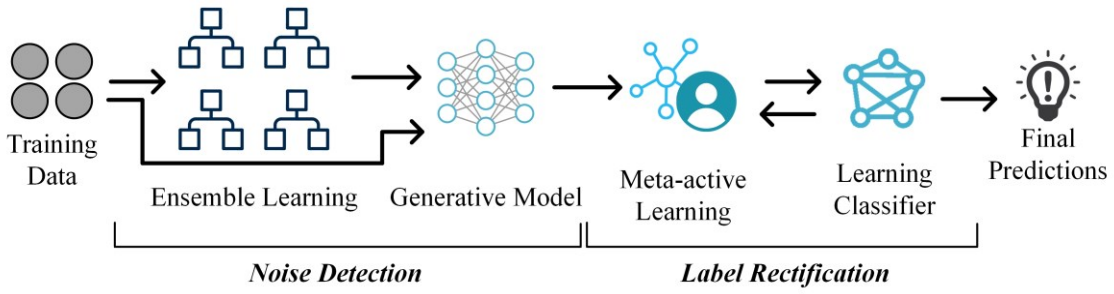


Figure 5.1: A component overview of the proposed method

Nevertheless, a few recent studies [14], [15] try to address the problem of learning with incomplete and inaccurate supervision simultaneously. However, most of these approaches assume specific configuration regarding the problem settings. For example, Guo *et al.* [14] presented an instance reweighting strategy to assign lower weights for noisy labels. The approach [14] also suggests robust criteria that use AUC instead of classification accuracy to mitigate the problem of the bias label distribution. Similarly, Zhang *et al.* [15] propose a framework to learn with inaccurate and incomplete supervision. However, the framework [15] assumes that data only has one-sided instance-dependent noise. In such settings, labels from one class are flipped into the other class while the other class stays free of noise. Additionally, these approaches only consider binary classification problems. Also, they are evaluated within specific domains such as ride-sharing [14] and the detection of software bugs [15].

Therefore, to overcome these challenges, we propose Smart Mendr, a new classification **M**odel that applies **En**semble Learning and **D**ata-driven **R**ectification to handle both scenarios of inaccurate and incomplete supervision. An overview of the proposed method is illustrated in Figure 5.1. As the figure shows, the method has two phases. In the first phase, Smart Mendr applies a preliminary stage of ensemble learning to estimate the probability of each instance being mislabeled and produce initially weak labels for unlabeled data. However, to overcome the challenges of noise detection using ensemble learning, we apply a semi-supervised learning approach to combine the output of the ensemble and report the noisy points. After that, the proposed method, in the second phase, applies a smart correcting procedure using meta-active learning to provide true labels for both noisy and unlabeled points. The source code of the proposed framework is available at https://github.com/MonaNashaat/SmartMendr.

To evaluate the proposed method, we compare its performance with state-of-the-art techniques dealing with inaccurate and incomplete supervision. During the experiments, we evaluate the classification performance, noise detection, and the accuracy of the corrected labels. The experiments explore a wide range of classification tasks, including binary and multi-classification problems, with 15 datasets that vary in size and dimensionality.

The rest of the chapter is structured as follows: Section 5.2 discusses the background related to the research. Section 5.3 describes the proposed approach. Section 5.4 presents the experiments

conducted to evaluate the proposed method, along with the obtained results. While Section 5.5 discusses related work, and Section 5.6 concludes the chapter.

5.2. Background

In this section, we review existing methods to deal with learning with weak supervision, more specifically, learning with incomplete and inaccurate supervision. In the first subsection, we discuss learning with inaccurate supervision, which includes filtering methods and noise-robust classifiers. In the second subsection, we discuss different approaches for incomplete supervision, such as semi-supervised learning and active learning.

5.2.1. Learning with inaccurate supervision

In inaccurate supervision, the task is to learn a classifier $f: \boldsymbol{X} \rightarrow Y$ from a training data $\{(\mathbf{x_i}, \mathrm{y_i})\}_{\mathrm{i=1}}^{\mathrm{N}}$ where $\mathrm{y_i}$ is incorrect for a portion of the training set. Existing techniques for classification with inaccurate supervision can be classified into 1) Filtering methods; and 2) noise-robust classifiers. Filtering techniques [4], [10], [16] are data-oriented methods that perform some preprocessing steps to identify and remove noisy data. Some filtering techniques [4], [10], [16] use an ensemble of classifiers to detect data with noisy labels. In these approaches, a set of classifiers is used to produces labels for the points in the training data. Then, the disagreements between the output of these classifiers are used to decide on noisy points. On the other hand, some filtering approaches [17] depend on the neighborhood information of data points in the training set; these approaches [17] iteratively employ k-NN classifiers to detect points whose labels are not consistent with their neighbors. Then, those examples are marked as noise and eliminated.

Likewise, Guan *et al.* [10] present an ensemble-based filter that adopts a soft majority voting to output voting results and the confidence values of the labels. Alternatively, Saman *et al.* [17] propose a preprocessing filtering phase to train conventional neural networks for image classification. The technique employs a rough set-based k-NN algorithm to eliminate noisy data before applying the neural networks. However, previous studies [11] state that, since filtering approaches can misidentify correct points as noise, this can deteriorate the classification performance. Also, other research [18] points out that, as ensemble-based filters are trained using noisy data, their results cannot be trusted. For instance, while the classifiers with one of these filters [10] achieved perfect accuracy values in the noisy version of the mushroom dataset, the

technique degraded the classification accuracy when applied to the noise-free version of the data. Similarly, some of these approaches [16] did not consider learning the noise rate in the data, which may result in limited improvements in noise detection.

Additionally, since k-NN noise filters depend on creating relative neighborhood graphs for training examples to estimate the labeling confidence, they are less reliable in high-dimensional feature spaces. Also, many studies [19] show that selecting the value of k depends on the noise ratio. Therefore, as the class noise increases, the value of k monotonically increases, which affects the scalability of these techniques.

Alternatively, noise-robust techniques are algorithm-oriented approaches that create learning models that can maintain their performance in the presence of noise. For example, although classical decision trees are known to be sensitive to class noise, C4.5 [20] is considered to be a robust decision tree algorithm. Moreover, many researchers [11], [21] recommend using ensembles of classifiers to create robust models. For instance, Miao *et al.* [21] modified the Adaboost algorithm by optimizing a nonconvex loss function of the classification margin to make it more robust to noise. However, most of these approaches rely on the classification algorithm, and thus, the achieved performance is inapplicable to other algorithms. Finally, other research [22] states that the performance of noise-robust techniques can differ when the noise ratios vary in each class.

5.2.2. Learning with incomplete supervision

Learning with incomplete supervision aims at creating a classifier $f: \boldsymbol{X} \rightarrow Y$ from a training data where only a small amount of data is labeled. Based on the level of interaction with domain experts, existing approaches, proposed to deal with incomplete supervision, can be classified into semi-supervised learning and active learning.

Semi-supervised learning [23] tries to employ both labeled and unlabeled data to create better models without human intervention. To accomplish this goal, some studies [23], [24] employ the concept of generative models to impute missing labels in the data. Generative models [24] assume that a joint probability model could be learned based on some assumptions about the underlying data distribution. For instance, Jain *et al.* [24] present a generative approach for multi-label learning that learns a latent factor model for labeling matrix to account for missing labels. Also, Liu *et al.* [23] develop a technique that applies a generative model with any supervised learning,

so the classification performance can be improved using unlabeled data. However, the process of learning the structure of such models can be expensive, especially when modeling a higher number of dependencies [13]. Since the learning complexity scales exponentially for higher degree dependencies, this limits the ability of the model to learn complex dependency structures.

Other SSL approaches [25], [26] try to represent the semi-supervised learning as a graph-based problem in which the graph nodes represent both the labeled and unlabeled examples. Then, the similarity between the nodes is measured to represent the graph edges. For instance, Du *et al.* [25] propose a graph-based approach that depends on the maximum correntropy criterion to learn a robust model. However, since these methods rely on building graphs, they do not scale well to large datasets [27]. They also cannot accommodate new data without reconstructing the graph.

Active learning, on the other hand, includes the user in the loop to provide ground-truth labels. In the standard setting of pool-based AL [6], a classifier is trained with a small labeled dataset. Then, the query strategy is applied to select additional points from the unlabeled pool and query the user to provide ground-truth labels for these points. After that, the obtained labels are added to the labeled dataset and used to retrain the model. The model performance is then reevaluated with the test set, and the procedure is repeated until a target performance is achieved or a maximum labeling budget is reached. Since the query strategy plays an essential role in the AL process, many query strategies have been proposed for different classification tasks [6], such as uncertainty sampling, query-by-committee, and density-based uncertainty sampling. However, previous studies [28] have proved that these heuristic-based strategies have limited performance when applied to different data distributions. As they use a static formula to measure either the informativeness or the representativeness of unlabeled points, their performance can be significantly impacted by several factors such as label noise and imbalanced classes.

Therefore, to deal with these situations, recent research [29], [30], [31] proposes meta-AL as an alternative solution. In meta-AL, the problem of selecting, or even designing, the query strategy is treated as a learning task to realize the best selection algorithm for the given data distribution. On the one hand, some research [29], [31] has extended existing query strategies to make them more robust to class noise and different distributions. For instance, one of these techniques [31] merges uncertainty sampling with diversity maximization to enforce diversity in the selected points and avoid overfitting. On the other hand, other studies have applied machine learning to learn the query

strategy. For example, Lin *et al.* [30] propose a technique that switches between different query strategies to deal with imbalanced classes. However, most of these techniques [29] focus on binary classification tasks. Also, since some of these approaches [30], [31] utilize existing strategies such as uncertainty sampling, they do not perform well with high rates of noise.

5.3. Smart Mendr: The Proposed Approach

In the following subsections, we discuss the design of the proposed system. In Section 5.3.1, we discuss some notation and formulate the problem. Then, in Section 5.3.2 and Section 5.3.3, we describe, in detail, the phases of Smart Mendr.

5.3.1. Problem Formulation

Formally, let D be an incomplete noisy dataset of size N, which can be split into two datasets: a dataset with class noise D_p and an unlabeled dataset D_u. The noisy labeled dataset $D_p \subset D$ consists of $\{(\mathbf{x_i}, y_{n_i})\}_{i=1}^{N_p}$, particularly, the data points in D_p comprise $\mathbf{x_i} \in \mathbf{X}$ and $y_{n_i} \in Y$. While $\mathbf{X} = R^d$ is a d-dimensional feature space, $y_{n_i} \in Y$ where $Y = \{y_1, y_2, \ldots, y_m\}$, which is the output space with m class labels. Let us also denote a noise rate p to be associated with the output labels y_n in D_p. It is assumed that $p \in [0,0.5)$, so there are more correctly labeled instances than mislabeled instances. Additionally, let $D_u \subset D$ denotes a subset of data points with unknown labels consisting of $D_u = \{(x_i)\}_{i=1}^{N_u}$ where $N_u = N - N_p$.

In the proposed method, dealing with inaccurate and incomplete supervision involves providing the correct labels to the points in D_u and relabeling the noisy data in D_p. Therefore, handling inaccurate supervision can be seen as a preliminary phase of identifying data points with noisy labels before proceeding with the classification problem. Thus, in our problem setting, it can be reduced to a special case of learning with incomplete supervision.

The proposed method aims at inducing a classifier $f: \mathbf{X} \rightarrow Y$ using D as the training data. It seeks to create highly generalizable learning models, even when a large proportion of the training data is mislabeled or unlabeled. To achieve this goal, the proposed framework is divided into two phases, namely *noise detection via ensemble learning* and *iterative label rectification using meta active learning*. An overview of the two phases of the proposed system is illustrated in Figure 5.2.

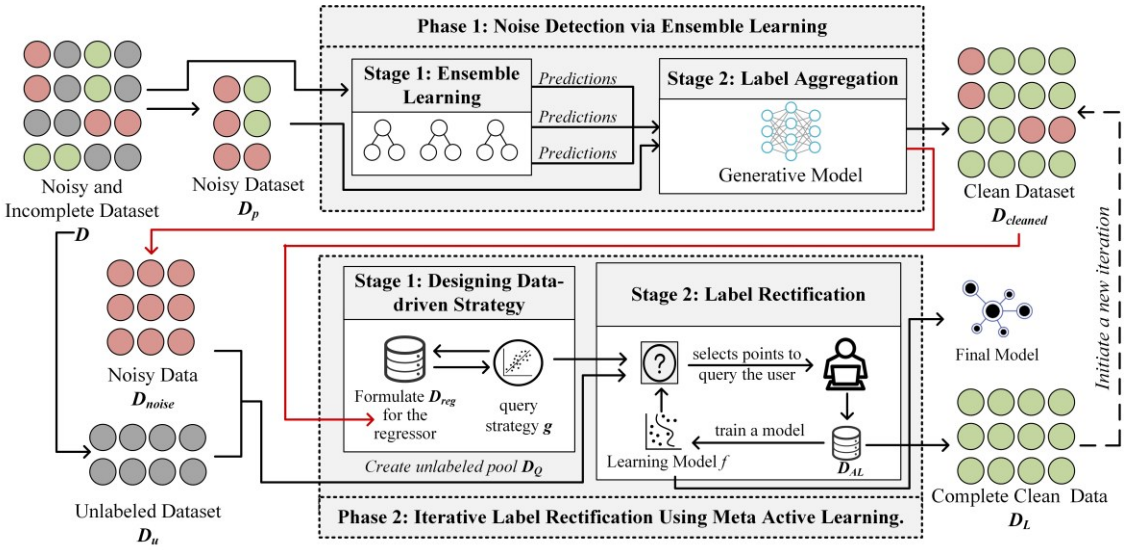


Figure 5.2: Overview of the two phases of the proposed approach

As the figure shows, in the first phase (Section 5.3.2), the proposed system exploits different bootstrap samples from the noisy dataset D_p to create a set of base classifiers. In filtering approaches [9], the misclassified instances are assumed to be noisy and removed. However, deleting noisy instances can be unfavorable, especially when the data is expensive to acquire or misidentified as noisy. Therefore, in Smart Mendr, the ensemble predictor is combined with the original data using a generative model to estimate the labeling confidence of each data point in D_p, and produce a set of initial probabilistic labels for the unlabeled points in D_u.

Consequently, in the second phase (Section 5.3.3), the proposed method tackles the problem of selecting which data points should be labeled by an oracle. The phase starts by designing a query strategy that is customized to consider the underline data distribution and the labeling confidence results obtained from the first phase. Finally, the query strategy is applied to rectify the data points with the noisy labels, provide correct labels for the unlabeled points in D_u, and improve the classifier performance to make predictions for unseen instances.

5.3.2. Phase 1: Noisy Label Detection via Ensemble Learning

In this phase, the proposed method aims at detecting data points with noisy labels in D_p and producing initial labels for the unlabeled points in D_u. Therefore, the phase employs a set of ensembles in two stages. In the first stage, a set of base learners are built to produce predictions for the data points in D. Then, the ensemble predictor is utilized in the second stage to detect noisy

points D_{noise} in D_p.

The stage takes D_p as an input along with an out-of-bag dataset to estimate the generalization error of the ensembles. As for creating the ensemble, we consider randomized ensembles, specifically, random forests, in which each classifier is trained on bootstrap samples of D_p. A detailed description of this stage is illustrated in Algorithm 1. As the first part of the algorithm shows (steps 2-11), the stage starts with deciding on the sampling rate and building the ensembles. Many studies [32], [33] verified that having a small sampling rate can make the ensemble more robust to label noise. Therefore, the sampling rate r is chosen from a range $r \in \{r_{min}, \dots r_{max}\}$ where $r_{min} = 0.1$ and $r_{max} = 0.4$ [33]. Next, for each sampling rate, a set of base learners H is iteratively created. The algorithm uses the out-of-bag dataset to evaluate the generalization error of the ensemble. Then, the set with the least generalization error H_{best} is selected for the next stage. Therefore, an unbiased selection of the ensembles is made regardless of the amount of data noise. Although the ensemble is trained using noisy data, the phase utilizes a robust model such as random forests and builds the ensemble with a small sampling rate to reduce the noise effect. The ensemble predictor of H_{best} is described as:

$$Y_{H_{best}} = \arg\max_y \sum_{j=1}^{T} \mathbf{I}(h_{best_j}(x) = y) \tag{5.1}$$

where $h_{bestj}(x)$ is the prediction of the response variable at x using the j^{th} base classifier in the ensemble H_{best}, and T is the ensemble size.

Accordingly, the second stage, shown in algorithm 1 (steps 12-16), utilizes H_{best} to produce labels to the data points in D and detect noise in D_p. To detect noisy data, filtering approaches must decide on a threshold of erroneous ensemble predictions to classify a given instance as noise. In other words, data points for which the fraction of misclassified predictions given by the classifiers in the ensemble exceeds this threshold are filtered as noise. However, previous studies [19], [32] attest that the optimal value of the threshold is problem-dependent, and therefore needs to be estimated for each classification task. Hence, to avoid the overhead of having to determine the filtering threshold for each classification problem, Smart Mendr formulates the problem, at this point, using weak supervision settings. As mentioned before, learning with weak supervision is based on dealing with low-quality but large-scale training examples. And since both the output of the ensemble Y_{Hbest} and the original labels in D contain label noise and hence can be considered as low-quality sources, they can be treated as two sources of weak supervision.

Algorithm 5.1: Noise Detection via Ensemble Learning

Input: $D_p = \{(\mathbf{x_i}, y_{n_i})\}_{i=1}^{N_p}$ % noisy training data

D_{oob} % Out-of-bag data

r % samplying_rate_range

T % ensemble size

Output: D_{noise} % detected noise

$D_{cleaned}$ % a cleaned version of D_P

Y_{gen} % probabilistic labels

1: $E_{min} = \infty$

2: **for each** sampling_rate **in** r **do**

3: $N_r \leftarrow$ sampling_rate * N_p

4: take a bootstrap sample D_r of size N_r from D_p.

5: create a randomized ensemble H of size T with D_r

6: estimate the generalization error E of H using D_{oob}

7: **if** $E < E_{min}$ **then**

8: $E_{min} = E$

9: $H_{best} = H$

10: **end**

11: **end**

12: construct the matrix of weak sources S (Equation (5.3))

13: learn $\emptyset$ for a generative model m_{Gen} (Equation (5.2))

14: obtain labels y_{gen} using m_{Gen} for the points in D

15: estimate the threshold value θ (Equation (5.5))

16: using θ, detect noise from D_p and construct D_{noise} ((Equation (5.6))

17: construct the cleaned data set $D_{cleaned} = D_p - D_{noise}$

18: return D_{noise}, $D_{cleaned}$, and Y_{gen}

As mentioned in Section 5.2, to integrate training labels from multiple weak sources, previous studies [2], [13] have used generative models to estimate the accuracy of each source and any statistical dependency between their outputs. As the generative model treats the true label as a latent variable, after fitting the generative model, the distribution of the true label Y is estimated as a set of probabilistic labels. Therefore, in this stage, the proposed method learns a generative model m_{Gen} to estimate the accuracy of the ensemble predictions for the data points in D and the noisy data points in D_p before combining these two sources. The generative model can be formally defined as:

$$m_{Gen} : \pi_{\varphi}(S, Y) = \frac{1}{Z_{\emptyset}} e^{\emptyset^T S Y} \tag{5.2}$$

where S is a matrix denoting the output of the weak sources, Ø is the accuracy of each source in S, and $Z_{Ø}$ is a partition function to ensure π is a distribution. The proposed method tries to address the scalability issue of learning a generative model for higher degree dependencies by limiting the number of weak sources to include Y_{Hbest} and y_n. Hence, the model can learn the structure for these sources with a sample complexity that only scales sublinearly with the number of binary dependencies [13]. As a result, the matrix S can be defined as:

$$s_{i,j} = \begin{cases} y_{n_i} & if\ j = 1 \\ y_{H_{best_i}} & if\ j = 2 \end{cases} \quad \text{where } 1 \leq i \leq N_p, 1 \leq j \leq 2 \tag{5.3}$$

where y_n is the noisy class label in D_p, and Y_{Hbest} is the ensemble predictor. The generative model outputs a vector of probabilistic labels $y_{gen} = P[y = 1]$ which denotes how confident the generative model is about each class label in D. For example, for data points that are misclassified by the ensemble, and therefore their labels differ from y_n, the generative model would output probabilistic labels for these points that are close to 0.5. Thus, we formally define the points with noise labels as:

$$|P[y_i = 1] - 0.5| \leq \theta \tag{5.4}$$

where $P[y_i = 1]$ is the probabilistic label assigned by the generative model, and θ is a threshold to ensure that the definition of low confidence changes according to the number of the weak sources with which the generative model operates. Since the number of weak sources remains constant regardless of the problem in question, we avoid the overhead of recalculating the filtering threshold for every problem. Moreover, since the generative model learns the underlying data distribution,

its output can be treated as the labeling confidence and used to detect noisy points. Therefore, the threshold θ can be denoted as:

$$\theta = \psi - (1/e^{\sqrt{k+1}}) \tag{5.5}$$

where k is the number of weak sources (in this case k=2), and ψ is the initial value before measuring the exponential decay as k increases (default $\psi = 1/3$). In other words, we expect to have fewer data points with labeling confidence close to 0.5 when the number of weak sources grows. Thus, the phase uses (5.4) to detect the points with noisy labels in D_p as:

$$D_{noise} \subseteq D_n, \forall\, x_{n_i} \in D_{noise}\, \{x_i |\, |P[y_i = 1] - 0.5| \leq \theta\} \tag{5.6}$$

The phase applies the formula above to eliminate the noisy data points from D_p in a new dataset D_{noise} containing all the detected noise. The phase outputs D_{noise} and $D_{cleaned} = D_p - D_{noise}$ and sends both datasets to the second phase. In the second phase, the proposed method aims at providing the correct labels for both these noisy labels in D_{noise} and the unlabeled dataset D_u.

5.3.3. Phase 2: Label Rectification using Meta-AL

As the first phase eliminates the data with noisy labels in D_{noise} and utilizes m_{Gen} to produce initial (noisy) predictions to D_u, the second phase has three goals, 1) to rectify the noisy labels in D_{noise}, 2) to give accurate labels to D_u, 3) to induce a classifier f that is trained with D. To accomplish these goals, the noisy points in D_{noise} are combined with D_u to form unlabeled pool $D_Q = D_u \cup D_{noise}$. The problem at this point can be considered as a task of AL, where the goal is to give labels to the points that are expected to improve the model performance.

However, the phase cannot apply traditional query strategies such as uncertainty sampling [6] because the problem settings in our case differ from the traditional scenario of AL. While in pool-based AL, we start with a small set of labeled points (seed) and an unlabeled pool, alternatively, in our setting, we start with a bigger seed D_{clean} and a set of unlabeled data D_Q along with a vector of labeling confidence Y_{gen} produced by the generative model for each point in D. Also, as mentioned before, traditional query strategies can provide sub-optimal solutions with different data distributions and noise levels [28].

Hence, for the above reasons, a meta-active learning approach is adopted in this phase to design the query strategy. We articulate the design process as a regression problem, in which we train a

model to estimate the reduction in the generalization error associated with labeling the points in D_Q. Then, only the data points with the highest reduction in the generalization error are selected and rectified by an oracle. Similar to the first phase, this phase has two main stages. In the first stage, a meta-AL query strategy is designed, while in the second stage, the obtained strategy is applied to rectify the labels.

In the first stage, the design process of the query strategy is framed as a regression problem. This step aims at creating a regression model g that is supposed to, when applied to D_Q, to choose the points that result in the maximum reduction ∇ to the generalization error. To start the process, we use $D_{cleaned}$ to create a set of labeled observations D_g needed to train and test a regressor g. Therefore, $D_{cleaned}$ is split into a training set D_{train} and testing set D_{test}. Then, we use the data points in D_{train} to iteratively train a classifier c and record the corresponding reduction to the generalization error of the produced model.

To accomplish this task, we further split D_{train} into a labeled training dataset $D_{labeled}$ of size w and a data pool D_{pool} containing the remaining points. Then, we use $D_{labeled}$ to train c and produce a model m_d that is used to provide predictions to the points in D_{test} and estimate the corresponding classification loss L_d. After that, we randomly select another data point x from the pool D_{pool}, and add it to $D_{labeled}$, and form a new dataset $D_x = D_{labeled} \cup \{x\}$. After that, we utilize D_x to train c again, create a new model m_x, and test this model using D_{test}. Similarly, the new classification Loss L_x is calculated and the reduction in the classification loss ∇_x for adding x to $D_{labeled}$ is estimated as:

$$\nabla_x = L_d - L_x \tag{5.7}$$

Additionally, as we are recording the reduction in the generalization error ∇_x associated with adding each point x from D_{pool} to $D_{labeled}$, we need to associate these reductions to a set of features φ that reflect the data distribution and the labeling confidence. Thus, we consider that each point that is added to $D_{labeled}$ can be characterized by a set of parameters φ_x that includes the value of its labeling confidence y_{gen}, the distance to the closest point in the dataset, and the distance to the closest labeled point. Also, as we collect these observations (φ_x, ∇_x), we iteratively build D_g using different samplings of $D_{labeled}$ with different sizes $w \in \{w_{min}, \dots w_{max}\}$. Based on the insights obtained from the experiments (Section 5.4), we repeat the process with different sizes equal to 30%, 50%, 70% 90% of the total size of $D_{labeled}$, as we found this range can result in enough

observations (φ_x, ∇_x) to train an adequate regressor without affecting the time complexity (The time to learn the AL strategy for a dataset of 78k records was less than 5 seconds on an Intel i7 machine with 32 GB RAM).

Therefore, during each iteration, we randomly sample w points from D_{train} and record both the features φ of w points in D_{train} along with their corresponding reduction ∇ to the generalization error. Finally, D_g is used to train a regression function g to predict the error reduction of annotating the points in D_Q. The complete process of designing the query strategy is explained in Algorithm 2.

In the second stage of this phase, the trained regression function g is applied as the query strategy to rank the points in D_Q. The model then selects data points from D_Q that are expected to result in the highest error reduction using the following formula:

$$x^* = \underset{x \in D_Q}{\arg\max}\ g(\varphi_x) \tag{5.8}$$

Moreover, to overcome the cold-start problem in AL [6], the component uses $D_{cleaned}$ as the initial seed. Initially, a probabilistic classifier f is trained using $D_{cleaned}$. Then, in each iteration of AL, the points in D_Q are ranked using (5.8), and the regression function g selects the data points with the highest reduction in the generalization error. Next, the user is queried to provide true labels for the selected points, which are then added to $D_{cleaned}$. Finally, the updated $D_{cleaned}$ is then used to retrain the classifier f for the next iteration. Therefore, the process gradually creates a labeled dataset $D_L = D_{cleaned} \cup D_{AL}$, where $D_{AL} = \{\mathbf{x}_i, y_i^*\}_{i=1}^{\min(B,Q)}$ represents the data points that received true labels from the user during this stage, and Q is the number of data points in D_Q. The AL process terminates when either D_Q is completely labeled, or a predefined labeling budget B is exceeded. Therefore the size of D_{AL} is denoted as min(B, Q). Finally, the phase outputs D_L as the complete clean version of D along with the classifier f trained using D_L.

Moreover, as illustrated in Figure 5.2, phase 2 is iterative. Therefore, another iteration can be initialed by the user. In this iteration, $D_{cleaned}$ is replaced with D_L, and another round is executed. Hence, a new query strategy is designed using D_L to further enhance the final performance. However, the experiments (Section 5.4) show that running only one iteration of the process can help obtain an adequate level of classification performance for real-world tasks and outperform state-of-the-art techniques.

Algorithm 5.2: Designing the Query Strategy

Input: $D_{cleaned}$ % cleaned version of the data

D_{noise} % detected noise

D_u % the unlabeled points in D

y_{gen} % probabilistic labels produced by m_{Gen}

Output: g % regressor function (the query strategy)

1: initialize dataset $D_Q = D_u \cup D_{noise}$

2: create two datasets D_{train} and D_{test} by splitting $D_{cleaned}$

3: initialize an empty dataset D_{reg}

3: **for** w **in** $\{w_{min}, \ldots w_{max}\}$ **do**

5: Split D_{train} into $D_{labeled}$ of size w and D_{pool}

6: train a classifier *c* with $D_{labeled}$

7: calculate the classification loss L_d using D_{test}

8: **for** each point x in D_{pool} **do**

9: form a new dataset $D_x = D_{labeled} \cup \{x\}$

10: train the same classification algorithm *c* using D_x

11: calculate the new test loss L_x

12: calculate the error reduction $\nabla_x = L_d - L_x$

13: collect the data point parameters φ_x

14: add the labeled data point $\{\varphi_x, \nabla_x\}$ to D_{reg}

15: return D_{reg} of size Q as $\{\varphi_x, \nabla_x\}$

16: train a regressor g using D_{reg}

17: return g

5.4. Experimental Framework

In this section, we present the results of extensive experiments carried out to check the validity of

Table 5.1: Datasets statistics

Datasets	N	dim.	m	# noise			# unlabeled		
				low	mod.	high	easy	medium	hard
activity	42,240	6	6	10,560	12,672	16,896	21,120	27,456	33,792
APS failure	60,000	171	2	15,000	18,000	24,000	30,000	39,000	48,000
Avila	20,867	10	12	5,217	6,260	8,347	10,434	13,564	16,694
banana	5,300	2	3	1,325	1,590	2,120	2,650	3,445	4,240
census	48,842	14	2	12,211	14,653	19,537	24,421	31,747	39,074
connect-4	67,557	42	3	16,889	20,267	27,023	33,779	43,912	54,046
german	1,000	20	2	250	300	400	500	650	800
HTRU2	17,898	9	2	4,475	5,369	7,159	8,949	11,634	14,318
MoCap	78,095	38	5	19,524	23,429	31,238	39,048	50,762	62,476
penbased	1,0992	16	10	275	330	440	550	714	879
shoppers	12,330	18	2	3,083	3,699	4,932	6,165	8,015	9,864
shuttle	2,175	9	7	544	653	870	1,088	1,414	1,740
statlog	58,000	9	7	14,500	17,400	23,200	29,000	37,700	46,400
twonorm	7,400	20	3	1,850	2,220	2,960	3,700	4,810	5,920
yeast	1,484	8	10	371	445	594	742	965	1,187

N is the number of records each dataset has, dim. is the number of attributes, and m is the number of classes.

Smart Mendr. The section is divided into four subsections. In the first subsection (Section 5.4.1), we discuss the datasets used in the experiments. Then, in Section 5.4.2, we present the experimental setup and the techniques used in the comparison. Finally, Sections 5.4.3 and 5.4.4 discuss the obtained results of evaluating Smart Mendr in different scenarios of inaccurate and incomplete supervision, respectively.

5.4.1. Datasets

We consider 15 benchmark datasets from the UC Irvine Machine Learning repository[3] and the Kaggle data repository[4], that cover a range of classification tasks, including binary and multi-classification problems. Summary statistics of the datasets are provided in Table 5.1.

Furthermore, to simulate different scenarios of learning with class noise, we introduced different

[3] https://archive.ics.uci.edu/
[4] https://www.kaggle.com/datasets/

noise levels p into each dataset following a uniform class noise scheme [11]. Therefore, we randomly replaced class labels of a portion of the data points with labels of other classes. In the experiments, we consider three noise setups where 25% (low), 30% (moderate), and 40% (high) of the data instances in the original datasets are introduced with noisy class labels. After creating the noisy versions for each dataset, both the noisy and original versions were partitioned into five equal folds. Then the experiments used partitions from the noisy versions to train the classifiers, whereas the test partitions were collected from the original datasets to construct noise-free test datasets.

On the other hand, to replicate the situations of learning with unlabeled data, we randomly treated a portion of each dataset as unlabeled by removing the corresponding class labels. The experiments also consider three scenarios of incomplete supervision as 50% (easy), 65% (medium), and 80% (hard) of the labels in each dataset were removed. Table 5.1 also shows the simulation setups for each dataset, so it demonstrates the number of flipped instances in case of learning with class noise (#noise), and the number of unlabeled data points for learning with incomplete supervision (#unlabeled) for each setup.

5.4.2. Experimental Setup

The experiments compare Smart Mendr with other commonly used approaches. Specifically, for learning with noisy class labels, we consider the following techniques:

- **Filtering-based approach (Filtering)** [10]: the method depends on majority filtering while applying soft multiple majority voting to output a degree of trust for each filtered point.
- **Bagging sampling method (Bagging)** [8]: the approach randomly subsamples a portion of the data to build a bagging ensemble. The ensemble predictions are then combined using majority voting. Finally, the misclassified instances are marked as noise and eliminated.

As for learning with unlabeled data, the experiments compare the proposed approach against:

- **Semi-supervised learning (SSL):** we applied the generative model-based method proposed by Bach *et al.* [2]. The method assumes that labels of unlabeled instances can be treated as missing values of the model parameters, and thus, can be estimated using maximum pseudolikelihood estimation.

The experiments treat the classification results obtained using the original datasets as the gold standard **(Gold)**. Alternatively, the results using the altered datasets (by either introducing class noise or removing class labels) without applying any of the comparing approaches are considered as the baseline model **(Baseline)**.

The experiments calculate the accuracy, the Matthews correlation coefficient (MCC), and F1 measure achieved by Smart Mendr and the other approaches. For each dataset, we performed 20 runs of five-fold cross-validation. To conserve space, only F1 scores are reported here, the results of the other metrics (accuracy, and MCC) are presented in Appendix B. The experiments are designed to maintain the same percentage of label changes applied to each fold. Hence, the results are averaged (mean) across the total number of runs per dataset for each setup. The reported values reflect the average of each trial and the standard deviation. Regarding the labeling budget used in the experiments, we set a maximum budget of 7% of the total training set size. The value was determined based on our interaction with the industry as an acceptable labeling cost for real-world business applications [34], [35]. Moreover, the experiments consider three classification algorithms, namely radial kernel SVM (SVM), k-nearest neighbor (kNN), and logistic regression (Logit). To compare between repeated measurements, we applied the (nonparametric) Wilcoxon test [36] on the obtained results and reported the p-values.

5.4.3. Experiments of Inaccurate Supervision

The evaluation in this section is twofold. In Section 5.4.3.1, we assess the predictive performance of Smart Mendr along with the filtering and bagging techniques. Subsequently, in Section 5.4.3.2, we evaluate the noise detection capabilities by reporting the percentage of actual noisy instances that were detected as noise by each method.

5.4.3.1 Classification Performance

The experimental results obtained using the 15 datasets are shown in Tables 5.2- 5.3. The tables show the F1 score achieved by each classifier while using each of the comparing methods. The best results accomplished for each dataset within each noise level are highlighted in boldface. The results attest that the classification performance consistently declines across all the datasets when the noise level increases. For example, in the penbased dataset, the performance of the three models

Table 5.2: F1 measure with different noise levels (Inaccurate Supervision) (I)

	SVM	KNN	Logit	SVM	KNN	Logit	SVM	KNN	Logit
	activity (low)			activity (moderate)			activity (high)		
Gold	0.85 ± 0.0	0.80 ± 0.0	0.82 ± 0.1	0.85 ± 0.0	0.80 ± 0.0	0.82 ± 0.1	0.85 ± 0.0	0.80 ± 0.0	0.82 ± 0.1
Baseline	0.76 ± 0.1	0.70 ± 0.2	0.62 ± 0.0	0.69 ± 0.0	0.61 ± 0.0	0.32 ± 0.0	0.43 ± 0.0	0.24 ± 0.0	0.17 ± 0.0
S. Mendr	**0.83 ± 0.1**	**0.78 ± 0.0**	**0.80 ± 0.4**	**0.77 ± 0.1**	**0.65 ± 0.2**	**0.74 ± 0.4**	**0.61 ± 0.5**	**0.63 ± 0.3**	**0.59 ± 0.0**
Filtering	0.79 ± 0.4	0.73 ± 0.2	0.63 ± 0.4	0.73 ± 0.4	0.64 ± 0.0	0.60 ± 0.1	0.56 ± 0.3	0.35 ± 0.1	0.48 ± 0.2
Bagging	0.77 ± 0.0	0.71 ± 0.1	0.75 ± 0.0	0.70 ± 0.1	0.61 ± 0.1	0.69 ± 0.0	0.55 ± 0.1	0.59 ± 0.5	0.51 ± 0.2
	APS failure (low)			APS failure (moderate)			APS failure (high)		
Gold	0.97 ± 0.0	0.93 ± 0.4	0.92 ± 0.0	0.97 ± 0.0	0.93 ± 0.4	0.92 ± 0.0	0.97 ± 0.0	0.93 ± 0.4	0.92 ± 0.0
Baseline	0.69 ± 0.2	0.66 ± 0.3	0.58 ± 0.0	0.53 ± 0.0	0.51 ± 0.0	0.48 ± 0.1	0.51 ± 0.0	0.47 ± 0.0	0.42 ± 0.0
S. Mendr	**0.88 ± 0.5**	**0.86 ± 0.0**	**0.84 ± 0.4**	**0.82 ± 0.3**	**0.75 ± 0.0**	**0.78 ± 0.1**	**0.73 ± 0.0**	**0.71 ± 0.2**	**0.72 ± 0.0**
Filtering	0.82 ± 0.4	0.71 ± 0.0	0.71 ± 0.2	0.79 ± 0.2	0.61 ± 0.2	0.68 ± 0.5	0.64 ± 0.0	0.58 ± 0.3	0.60 ± 0.3
Bagging	0.87 ± 0.2	0.73 ± 0.4	0.74 ± 0.0	0.77 ± 0.2	0.73 ± 0.3	0.67 ± 0.0	0.71 ± 0.3	0.65 ± 0.0	0.66 ± 0.2
	avila (low)			avila (moderate)			avila (high)		
Gold	0.98 ± 0.2	0.98 ± 0.0	0.97 ± 0.6	0.98 ± 0.2	0.98 ± 0.0	0.97 ± 0.6	0.98 ± 0.2	0.98 ± 0.0	0.97 ± 0.6
Baseline	0.91 ± 0.0	0.81 ± 0.0	0.82 ± 0.0	0.76 ± 0.4	0.70 ± 0.0	0.61 ± 0.2	0.32 ± 0.3	0.41 ± 0.2	0.12 ± 0.1
S. Mendr	**0.94 ± 0.0**	**0.93 ± 0.4**	0.95 ± 0.1	**0.85 ± 0.4**	**0.82 ± 0.2**	**0.82 ± 0.3**	**0.83 ± 0.1**	**0.76 ± 0.3**	**0.78 ± 0.4**
Filtering	0.93 ± 0.2	**0.93 ± 0.1**	0.83 ± 0.4	0.78 ± 0.0	0.83 ± 0.5	0.70 ± 0.1	0.65 ± 0.4	0.68 ± 0.0	0.66 ± 0.3
Bagging	0.93 ± 0.2	0.87 ± 0.2	**0.96 ± 0.2**	0.81 ± 0.0	0.78 ± 0.0	0.73 ± 0.2	0.70 ± 0.4	**0.76 ± 0.4**	0.58 ± 0.2
	banana (low)			banana (moderate)			banana (high)		
Gold	0.93 ± 0.2	0.80 ± 0.3	0.89 ± 0.1	0.93 ± 0.2	0.80 ± 0.3	0.89 ± 0.1	0.93 ± 0.2	0.80 ± 0.3	0.89 ± 0.1
Baseline	0.73 ± 0.1	0.69 ± 0.0	0.61 ± 0.1	0.51 ± 0.1	0.44 ± 0.0	0.11 ± 0.0	0.45 ± 0.0	0.31 ± 0.0	0.21 ± 0.0
S. Mendr	**0.88 ± 0.1**	**0.77 ± 0.3**	**0.87 ± 0.3**	**0.82 ± 0.4**	**0.68 ± 0.2**	**0.63 ± 0.2**	**0.71 ± 0.4**	**0.65 ± 0.4**	**0.60 ± 0.0**
Filtering	0.80 ± 0.4	0.72 ± 0.0	0.75 ± 0.1	0.79 ± 0.3	0.52 ± 0.4	0.49 ± 0.0	0.63 ± 0.3	0.49 ± 0.1	0.24 ± 0.0
Bagging	0.81 ± 0.2	0.70 ± 0.0	0.67 ± 0.0	0.76 ± 0.2	0.60 ± 0.3	0.60 ± 0.1	0.65 ± 0.4	0.59 ± 0.4	0.49 ± 0.3
	census (low)			census (moderate)			census (high)		
Gold	0.90 ± 0.1	0.89 ± 0.1	0.86 ± 0.1	0.90 ± 0.1	0.89 ± 0.1	0.86 ± 0.1	0.90 ± 0.1	0.89 ± 0.1	0.86 ± 0.1
Baseline	0.78 ± 0.1	0.79 ± 0.2	0.64 ± 0.0	0.51 ± 0.1	0.77 ± 0.0	0.52 ± 0.1	0.34 ± 0.0	0.61 ± 0.0	0.49 ± 0.0
S. Mendr	**0.85 ± 0.2**	**0.84 ± 0.2**	**0.85 ± 0.3**	**0.79 ± 0.4**	**0.81 ± 0.4**	**0.82 ± 0.0**	**0.72 ± 0.3**	**0.80 ± 0.1**	**0.79 ± 0.3**
Filtering	0.79 ± 0.0	0.81 ± 0.0	0.69 ± 0.1	0.74 ± 0.1	0.79 ± 0.1	0.64 ± 0.3	0.64 ± 0.1	0.63 ± 0.2	0.54 ± 0.4
Bagging	0.79 ± 0.0	0.83 ± 0.1	0.84 ± 0.0	0.70 ± 0.4	0.77 ± 0.1	0.73 ± 0.3	0.65 ± 0.3	0.67 ± 0.0	0.59 ± 0.0
	connect4 (low)			connect4 (moderate)			connect4 (high)		
Gold	0.67 ± 0.2	0.56 ± 0.2	0.61 ± 0.1	0.67 ± 0.2	0.56 ± 0.2	0.61 ± 0.1	0.67 ± 0.2	0.56 ± 0.2	0.61 ± 0.1
Baseline	0.50 ± 0.0	0.43 ± 0.0	0.41 ± 0.0	0.37 ± 0.0	0.21 ± 0.0	0.39 ± 0.0	0.31 ± 0.0	0.17 ± 0.0	0.27 ± 0.1
S. Mendr	**0.63 ± 0.2**	**0.52 ± 0.2**	**0.59 ± 0.2**	**0.56 ± 0.2**	**0.49 ± 0.1**	**0.58 ± 0.2**	**0.51 ± 0.0**	**0.57 ± 0.0**	**0.55 ± 0.1**
Filtering	0.56 ± 0.4	0.47 ± 0.1	0.47 ± 0.4	0.48 ± 0.1	0.38 ± 0.0	0.44 ± 0.2	0.33 ± 0.0	0.29 ± 0.3	0.29 ± 0.3
Bagging	0.52 ± 0.1	0.49 ± 0.3	0.51 ± 0.1	0.42 ± 0.2	0.38 ± 0.2	0.41 ± 0.1	0.31 ± 0.4	0.35 ± 0.3	0.31 ± 0.1
	german (low)			german (moderate)			german (high)		
Gold	0.95 ± 0.0	0.92 ± 0.1	0.94 ± 0.0	0.95 ± 0.0	0.92 ± 0.1	0.94 ± 0.0	0.95 ± 0.0	0.92 ± 0.1	0.94 ± 0.0
Baseline	0.78 ± 0.1	0.78 ± 0.1	0.71 ± 0.1	0.68 ± 0.0	0.69 ± 0.0	0.51 ± 0.0	0.55 ± 0.0	0.55 ± 0.0	0.39 ± 0.2
S. Mendr	**0.90 ± 0.4**	**0.82 ± 0.1**	**0.89 ± 0.1**	**0.80 ± 0.4**	**0.79 ± 0.0**	**0.83 ± 0.3**	**0.80 ± 0.1**	**0.73 ± 0.3**	**0.74 ± 0.0**
Filtering	0.80 ± 0.5	0.78 ± 0.3	0.81 ± 0.1	0.77 ± 0.2	0.74 ± 0.2	0.65 ± 0.4	0.66 ± 0.4	0.68 ± 0.2	0.60 ± 0.4
Bagging	0.83 ± 0.0	0.81 ± 0.2	0.79 ± 0.0	0.75 ± 0.2	0.78 ± 0.3	0.75 ± 0.0	0.70 ± 0.2	0.70 ± 0.1	0.65 ± 0.0

(SVM, KNN, and Logit) deteriorated by 20.83%, 29.21%, and 42.55%, respectively, in the moderate noise setup when compared to the gold models. Similarly, the performance of the same

Table 5.3: F1 measure with different noise levels (Inaccurate Supervision) (II)

	SVM	KNN	Logit	SVM	KNN	Logit	SVM	KNN	Logit
	HTRU2 (low)			HTRU2 (moderate)			HTRU2 (high)		
Gold	0.95 ± 0.5	0.91 ± 0.0	0.93 ± 0.3	0.95 ± 0.5	0.91 ± 0.0	0.93 ± 0.3	0.95 ± 0.5	0.91 ± 0.0	0.93 ± 0.3
Baseline	0.41 ± 0.0	0.27 ± 0.0	0.22 ± 0.1	0.33 ± 0.2	0.20 ± 0.0	0.19 ± 0.0	0.18 ± 0.2	0.07 ± 0.3	0.11 ± 0.0
S. Mendr	**0.85 ± 0.3**	**0.80 ± 0.2**	**0.79 ± 0.7**	**0.81 ± 0.0**	**0.75 ± 0.0**	**0.70 ± 0.0**	**0.71 ± 0.3**	**0.69 ± 0.0**	**0.66 ± 0.0**
Filtering	0.69 ± 0.0	0.76 ± 0.0	0.72 ± 0.0	0.61 ± 0.0	0.70 ± 0.0	0.67 ± 0.0	0.53 ± 0.0	0.61 ± 0.0	0.51 ± 0.0
Bagging	0.72 ± 0.0	0.57 ± 0.5	0.63 ± 0.2	0.65 ± 0.1	0.52 ± 0.5	0.55 ± 0.3	0.57 ± 0.0	0.44 ± 0.3	0.50 ± 0.0
	MoCap (low)			MoCap (moderate)			MoCap (high)		
Gold	0.92 ± 0.0	0.90 ± 0.1	0.93 ± 0.0	0.92 ± 0.0	0.90 ± 0.1	0.93 ± 0.0	0.92 ± 0.0	0.90 ± 0.1	0.93 ± 0.0
Baseline	0.79 ± 0.1	0.80 ± 0.6	0.61 ± 0.1	0.62 ± 0.0	0.73 ± 0.0	0.59 ± 0.0	0.54 ± 0.0	0.61 ± 0.0	0.47 ± 0.0
S. Mendr	**0.91 ± 0.2**	**0.88 ± 0.3**	0.84 ± 0.5	**0.78 ± 0.4**	**0.80 ± 0.4**	0.70 ± 0.4	**0.67 ± 0.2**	**0.68 ± 0.0**	**0.69 ± 0.3**
Filtering	0.80 ± 0.2	0.81 ± 0.2	0.75 ± 0.2	0.71 ± 0.1	0.76 ± 0.4	0.68 ± 0.0	0.57 ± 0.3	0.63 ± 0.1	0.57 ± 0.2
Bagging	0.81 ± 0.4	0.83 ± 0.4	**0.92 ± 0.0**	0.65 ± 0.1	0.74 ± 0.2	**0.72 ± 0.0**	0.63 ± 0.0	0.64 ± 0.3	0.52 ± 0.1
	penbased (low)			penbased (moderate)			penbased (high)		
Gold	0.96 ± 0.0	0.89 ± 0.0	0.94 ± 0.0	0.96 ± 0.0	0.89 ± 0.0	0.94 ± 0.0	0.96 ± 0.0	0.89 ± 0.0	0.94 ± 0.0
Baseline	0.81 ± 0.0	0.80 ± 0.0	0.72 ± 0.0	0.76 ± 0.0	0.63 ± 0.0	0.54 ± 0.0	0.64 ± 0.0	0.37 ± 0.0	0.24 ± 0.3
S. Mendr	**0.91 ± 0.3**	**0.84 ± 0.2**	**0.90 ± 0.3**	**0.84 ± 0.3**	**0.73 ± 0.0**	**0.72 ± 0.2**	**0.77 ± 0.0**	**0.69 ± 0.1**	**0.67 ± 0.0**
Filtering	0.83 ± 0.1	0.81 ± 0.0	0.73 ± 0.4	0.78 ± 0.4	0.70 ± 0.1	0.61 ± 0.5	0.67 ± 0.4	0.50 ± 0.0	0.56 ± 0.0
Bagging	0.86 ± 0.3	0.82 ± 0.2	0.81 ± 0.0	0.77 ± 0.4	0.69 ± 0.0	0.70 ± 0.0	0.66 ± 0.2	0.67 ± 0.2	0.50 ± 0.0
	shoppers intention (low)			shoppers intention (moderate)			shoppers intention (high)		
Gold	0.94 ± 0.1	0.91 ± 0.2	0.90 ± 0.0	0.94 ± 0.1	0.91 ± 0.2	0.90 ± 0.0	0.94 ± 0.1	0.91 ± 0.2	0.90 ± 0.0
Baseline	0.81 ± 0.1	0.72 ± 0.0	0.61 ± 0.0	0.79 ± 0.0	0.61 ± 0.0	0.57 ± 0.0	0.61 ± 0.0	0.51 ± 0.0	0.53 ± 0.0
S. Mendr	**0.93 ± 0.1**	**0.86 ± 0.1**	**0.82 ± 0.0**	**0.85 ± 0.3**	**0.78 ± 0.0**	0.74 ± 0.2	**0.72 ± 0.1**	**0.74 ± 0.4**	**0.69 ± 0.2**
Filtering	0.82 ± 0.4	0.76 ± 0.4	0.77 ± 0.5	0.81 ± 0.4	0.72 ± 0.5	0.65 ± 0.2	0.62 ± 0.1	0.64 ± 0.4	0.61 ± 0.2
Bagging	0.84 ± 0.2	0.80 ± 0.1	0.80 ± 0.0	0.80 ± 0.2	0.73 ± 0.2	**0.79 ± 0.0**	0.64 ± 0.3	0.69 ± 0.1	0.64 ± 0.0
	shuttle (low)			shuttle (moderate)			shuttle (high)		
Gold	0.97 ± 0.0	0.91 ± 0.0	0.92 ± 0.0	0.97 ± 0.0	0.91 ± 0.0	0.92 ± 0.0	0.97 ± 0.0	0.91 ± 0.0	0.92 ± 0.0
Baseline	0.80 ± 0.0	0.81 ± 0.0	0.79 ± 0.0	0.75 ± 0.0	0.71 ± 0.3	0.61 ± 0.0	0.69 ± 0.0	0.53 ± 0.0	0.52 ± 0.0
S. Mendr	**0.93 ± 0.2**	**0.88 ± 0.3**	0.90 ± 0.4	0.80 ± 0.0	**0.72 ± 0.0**	**0.76 ± 0.1**	**0.78 ± 0.3**	**0.71 ± 0.1**	**0.71 ± 0.0**
Filtering	0.82 ± 0.4	0.83 ± 0.2	0.85 ± 0.3	**0.81 ± 0.1**	**0.72 ± 0.4**	0.64 ± 0.1	0.73 ± 0.0	0.65 ± 0.4	0.59 ± 0.3
Bagging	0.86 ± 0.3	0.81 ± 0.2	**0.91 ± 0.0**	0.79 ± 0.2	0.71 ± 0.4	0.67 ± 0.1	0.71 ± 0.2	0.70 ± 0.3	0.52 ± 0.0
	statlog (low)			statlog (moderate)			statlog (high)		
Gold	0.99 ± 0.0	0.97 ± 0.0	0.91 ± 0.0	0.99 ± 0.0	0.97 ± 0.0	0.91 ± 0.0	0.99 ± 0.0	0.97 ± 0.0	0.91 ± 0.0
Baseline	0.81 ± 0.1	0.77 ± 0.0	0.71 ± 0.4	0.70 ± 0.0	0.62 ± 0.7	0.69 ± 0.0	0.59 ± 0.1	0.43 ± 0.0	0.51 ± 0.0
S. Mendr	**0.91 ± 0.4**	**0.86 ± 0.4**	**0.90 ± 0.2**	**0.86 ± 0.2**	**0.82 ± 0.0**	**0.81 ± 0.2**	**0.78 ± 0.2**	**0.77 ± 0.3**	**0.75 ± 0.3**
Filtering	0.85 ± 0.2	0.81 ± 0.4	0.84 ± 0.2	0.82 ± 0.0	0.72 ± 0.4	0.76 ± 0.0	0.64 ± 0.1	0.53 ± 0.4	0.64 ± 0.2
Bagging	0.88 ± 0.3	0.85 ± 0.0	0.79 ± 0.0	0.79 ± 0.1	0.79 ± 0.1	0.77 ± 0.0	0.74 ± 0.1	0.75 ± 0.1	0.69 ± 0.0
	twonorm (low)			twonorm (moderate)			twonorm (high)		
Gold	0.98 ± 0.0	0.96 ± 0.0	0.95 ± 0.0	0.98 ± 0.0	0.96 ± 0.0	0.95 ± 0.0	0.98 ± 0.0	0.96 ± 0.0	0.95 ± 0.0
Baseline	0.73 ± 0.0	0.80 ± 0.2	0.81 ± 0.0	0.61 ± 0.0	0.72 ± 0.1	0.61 ± 0.0	0.56 ± 0.0	0.64 ± 0.0	0.23 ± 0.0
S. Mendr	**0.89 ± 0.2**	**0.95 ± 0.4**	**0.92 ± 0.4**	**0.88 ± 0.2**	**0.78 ± 0.2**	**0.80 ± 0.2**	**0.74 ± 0.1**	**0.75 ± 0.0**	**0.76 ± 0.1**
Filtering	0.85 ± 0.4	0.86 ± -0.	0.82 ± 0.3	0.78 ± 0.1	0.85 ± 0.1	0.68 ± 0.1	0.73 ± 0.0	0.67 ± 0.2	0.61 ± 0.4
Bagging	0.86 ± 0.4	0.85 ± 0.2	0.90 ± 0.0	0.79 ± 0.1	0.77 ± 0.1	0.78 ± 0.0	0.70 ± 0.1	0.74 ± 0.4	0.68 ± 0.1
	yeast (low)			yeast (moderate)			yeast (high)		
Gold	0.91 ± 0.1	0.92 ± 0.0	0.95 ± 0.4	0.91 ± 0.1	0.92 ± 0.0	0.95 ± 0.4	0.91 ± 0.1	0.92 ± 0.0	0.95 ± 0.4
Baseline	0.80 ± 0.1	0.80 ± 0.0	0.81 ± 0.0	0.69 ± 0.0	0.66 ± 0.1	0.63 ± 0.0	0.63 ± 0.1	0.38 ± 0.0	0.24 ± 0.1
S. Mendr	**0.87 ± 0.1**	**0.90 ± 0.4**	0.90 ± 0.3	**0.81 ± 0.2**	**0.76 ± 0.3**	**0.79 ± 0.4**	**0.67 ± 0.2**	**0.70 ± 0.0**	**0.70 ± 0.3**
Filtering	0.83 ± 0.3	0.82 ± 0.3	0.85 ± 0.2	0.74 ± 0.3	0.74 ± 0.3	0.72 ± 0.2	0.63 ± 0.1	0.65 ± 0.2	0.57 ± 0.4
Bagging	0.81 ± 0.2	0.84 ± 0.2	**0.91 ± 0.0**	0.71 ± 0.4	0.70 ± 0.2	0.73 ± 0.0	0.66 ± 0.4	**0.70 ± 0.3**	0.61 ± 0.0

models declines by 33.33%, 58.43%, and 74.47%, respectively, when the noise setup is changed

to high.

The filtering model manages to improve the prediction performance in almost all the datasets. However, with a high level of noise, it usually shows limited enhancement when compared to the baseline models. For example, in the shoppers, the penbased, and the statlog datasets, the filtering models with the SVM classifier improved the performance of the baseline models by 1.64%, 4.69%, and 8.47%, respectively. The reason for these limited improvements is that the filtering model tends to remove some correctly labeled data points that lie close to the decision boundary of the classifier. Hence, the generalizability of the classification models is affected.

As for the bagging model, it manages to achieve better performance than the filtering model in most cases, especially with the high level of noise. For example, it enhances the classification performance of Logit in the APS failure and the twonorm datasets by 10.04% and 11.48%, respectively, when compared to the filtering models.

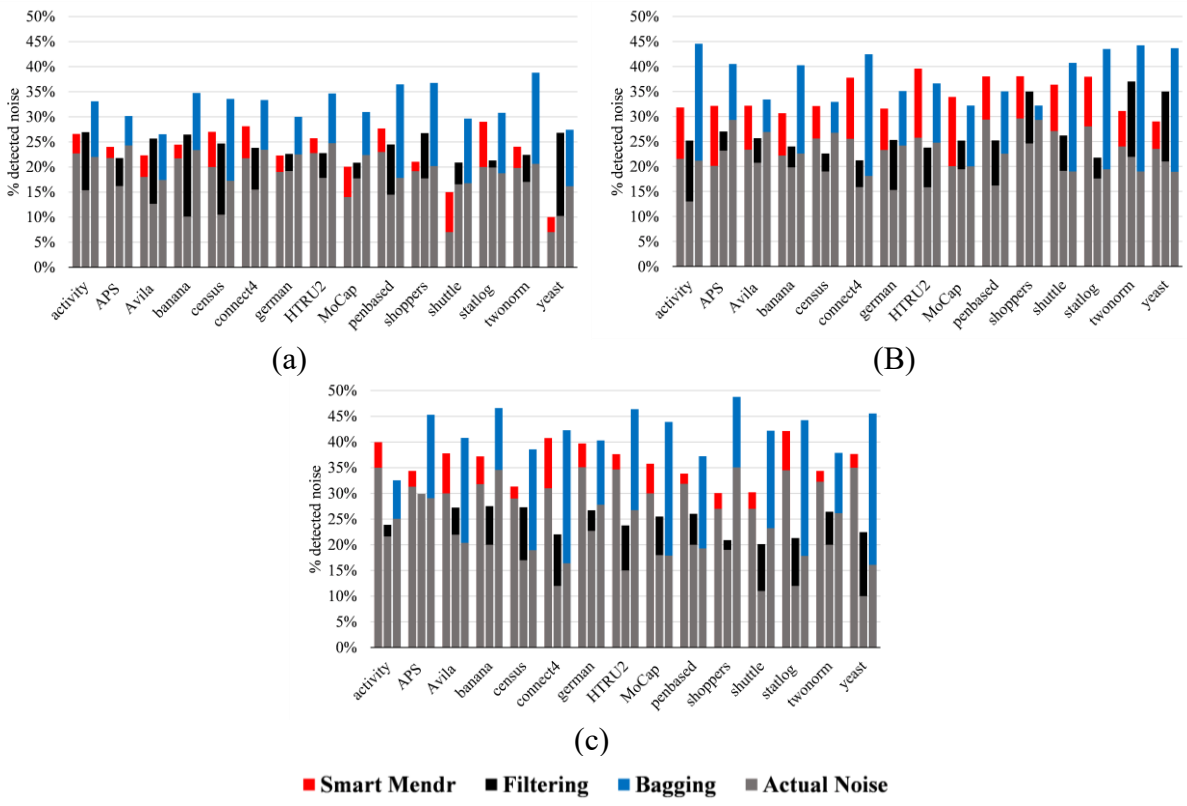


Figure 5.3: Percentage of noise detected by each method with different noise levels with (a) 25% injected noise (low), (b) 30% injected noise (moderate), and (c) 40% injected noise (high)

Table 5.4: P-values of paired Wilcoxon signed ranks test in inaccurate supervision experiments

	Classification Performance (low)			Classification Performance (moderate)			Classification Performance (high)		
	Baseline	Filtering	Bagging	Baseline	Filtering	Bagging	Baseline	Filtering	Bagging
Smart Mendr	$5.12\times e^{-9}$	$7.48\times e^{-9}$	$1.85\times e^{-7}$	$5.16\times e^{-9}$	$6.05\times e^{-8}$	$5.06\times e^{-8}$	$5.16\times e^{-9}$	$5.09\times e^{-9}$	$1.62\times e^{-8}$
	Noise detection (low)			Noise detection (moderate)			Noise detection (high)		
Smart Mendr	-	$2.16\times e^{-3}$	$1.79\times e^{-3}$	-	$3.77\times e^{-3}$	$6.41\times e^{-3}$	-	$1.99\times e^{-2}$	$6.55\times e^{-4}$

Alternatively, Smart Mendr achieves better results in most datasets, especially with the high setup. In some datasets, such as the connect4 and penbased datasets, it outperformed the filtering method by 89.66% and 19.64%, respectively. It also managed to surpass the bagging model in the same datasets by 77.42% and 34.02%, respectively. More formally, we applied the (nonparametric) Wilcoxon signed ranks test for performance comparison of the proposed method and the two other approaches. The p-values of the test are illustrated in Table 5.4 and show that the performance of Smart Mendr (in terms of F1 measure) is significantly different (better) than filtering and bagging models with all the noise setups.

5.4.3.2 Noise Detection

To estimate the effectiveness of Smart Mendr in detecting noise, we recorded the percentage of the noise identified by each technique with various noise setups. The percentage of the detected noise is calculated as the ratio between the number of data points identified as noise by each method to the total number of the instances in the dataset. The obtained results are depicted in Figure 5.3 for the three noise levels. In each chart in Figure 5.3, the average (mean) percentage of the noise detected by each technique is represented by the bars for all the datasets. Furthermore, from these percentages, the fraction of instances that resemble an actual injected noise are colored in grey.

The results show that the bagging model tends to aggressively mark more instances as noise than the other techniques. For example, in the shuttle dataset, the bagging model detects more noise than the proposed method by 12.01% and 39.65% in the moderate and high noise setups, respectively. Since the bagging model applies the majority voting to distinguish noisy data points, it discards more data points than the other approaches.

Alternatively, the filtering model does not detect high percentages of noise in most of the datasets.

However, in many tasks, the model removes a significant portion of clean points that are mistakenly identified as noise. For example, in the hard setup, 22.44% of the yeast dataset is detected as noise and hence removed from the training data. However, only 44.56% of these filtered instances are actual noise. As a result, the final model in this dataset shows a performance degradation (Table 5.3) of 30.77% with this setup when compared to the gold model.

As for Smart Mendr, the results show that, in most datasets, it manages to detect a high percentage of the injected noise without eliminating a high volume of noise-free instances. Although in some datasets, such as the HTRU2 and the german datasets in the moderate setup, the proposed method removed some noiseless data points (35.07% and 26.27% respectively), discarding these points does not seem to affect the performance of the final model (Table 5.3). Moreover, we report the p-values of the Wilcoxon signed ranks test for the comparison of noise detection in Table 5.4. The results show that the proposed method manages to achieve significantly better specificity across all noise setups.

5.4.4. Experiments of Incomplete Supervision

To compare the proposed method with the SSL technique [2], this section is divided into two subsections. In the first subsection, we estimate the predictive performance of the proposed method when compared to SSL. Second, in Section 5.4.4.2, we evaluate the accuracy of the predicted labels provided by each technique by comparing their outputs to the ground truth.

5.4.4.1 Classification Performance

Tables 5.5- 5.6 show the results obtained from applying the proposed method and SSL with different setups of incomplete supervision. As for the baseline models, the results show similar behavior as learning with noisy data since learning with missing labels severely affects the classification performance. For example, the performance of the baseline models deteriorated by 81.33% and 86.36% in the hard setup in the shoppers and the connect4 datasets, respectively, when compared to the gold models. As for SSL, the results show that it improves the classification performance when compared to the baseline models in almost all datasets. For example, with the medium setup, SSL, with the logit classifier, enhances the performance of the baseline models by 60.71% and 65.96% in the connect4 and statlog datasets, respectively.

Table 5.5: F1 measure for different levels of incomplete supervision (I)

	SVM	KNN	Logit	SVM	KNN	Logit	SVM	KNN	Logit
	activity (easy)			activity (medium)			activity (hard)		
Gold	0.85 ± 0.0	0.80 ± 0.0	0.82 ± 0.0	0.85 ± 0.0	0.80 ± 0.0	0.82 ± 0.0	0.85 ± 0.0	0.80 ± 0.0	0.82 ± 0.0
Baseline	0.79 ± 0.1	0.65 ± 0.1	0.67 ± 0.0	0.73 ± 0.0	0.52 ± 0.0	0.50 ± 0.0	0.45 ± 0.1	0.15 ± 0.1	0.36 ± 0.2
S. Mendr	**0.81 ± 0.1**	**0.79 ± 0.1**	**0.81 ± 0.0**	**0.77 ± 0.1**	0.69 ± 0.1	**0.71 ± 0.0**	**0.71 ± 0.1**	**0.68 ± 0.1**	**0.69 ± 0.0**
SSL	0.80 ± 0.0	0.76 ± 0.0	0.71 ± 0.0	0.74 ± 0.0	**0.71 ± 0.0**	0.65 ± 0.1	0.56 ± 0.0	0.54 ± 0.1	0.60 ± 0.0
	APS failure (easy)			APS failure (medium)			APS failure (hard)		
Gold	0.97 ± 0.0	0.93 ± 0.0	0.92 ± 0.0	0.97 ± 0.0	0.93 ± 0.0	0.92 ± 0.0	0.97 ± 0.0	0.93 ± 0.0	0.92 ± 0.0
Baseline	0.69 ± 0.0	0.76 ± 0.1	0.66 ± 0.1	0.53 ± 0.1	0.65 ± 0.0	0.65 ± 0.1	0.51 ± 0.0	0.23 ± 0.0	0.47 ± 0.1
S. Mendr	**0.95 ± 0.0**	**0.90 ± 0.0**	**0.91 ± 0.0**	0.86 ± 0.0	**0.87 ± 0.1**	**0.83 ± 0.0**	**0.84 ± 0.1**	**0.80 ± 0.1**	**0.78 ± 0.1**
SSL	0.90 ± 0.1	0.87 ± 0.1	0.87 ± 0.2	**0.87 ± 0.1**	0.79 ± 0.0	0.78 ± 0.0	0.72 ± 0.0	0.63 ± 0.0	0.70 ± 0.0
	avila (easy)			avila (medium)			avila (hard)		
Gold	0.98 ± 0.0	0.98 ± 0.1	0.97 ± 0.0	0.98 ± 0.0	0.98 ± 0.1	0.97 ± 0.0	0.98 ± 0.0	0.98 ± 0.1	0.97 ± 0.0
Baseline	0.89 ± 0.0	0.91 ± 0.2	0.87 ± 0.1	0.86 ± 0.0	0.82 ± 0.0	0.79 ± 0.0	0.78 ± 0.1	0.73 ± 0.0	0.63 ± 0.1
S. Mendr	**0.96 ± 0.1**	**0.97 ± 0.7**	0.93 ± 0.1	**0.90 ± 0.1**	**0.89 ± 0.0**	**0.87 ± 0.1**	**0.82 ± 0.1**	**0.85 ± 0.0**	**0.84 ± 0.1**
SSL	0.94 ± 0.1	0.91 ± 0.1	**0.94 ± 0.1**	0.86 ± 0.1	0.86 ± 0.0	0.83 ± 0.1	0.78 ± 0.0	0.74 ± 0.1	0.77 ± 0.0
	banana (easy)			banana (medium)			banana (hard)		
Gold	0.93 ± 0.1	0.80 ± 0.4	0.89 ± 0.0	0.93 ± 0.1	0.80 ± 0.4	0.89 ± 0.0	0.93 ± 0.1	0.80 ± 0.4	0.89 ± 0.0
Baseline	0.79 ± 0.1	0.71 ± 0.0	0.65 ± 0.0	0.51 ± 0.1	0.69 ± 0.1	0.52 ± 0.0	0.45 ± 0.1	0.56 ± 0.1	0.32 ± 0.0
S. Mendr	**0.91 ± 0.1**	**0.78 ± 0.1**	**0.86 ± 0.0**	**0.87 ± 0.0**	**0.74 ± 0.0**	**0.84 ± 0.2**	**0.80 ± 0.1**	**0.69 ± 0.2**	**0.78 ± 0.2**
SSL	0.88 ± 0.0	0.73 ± 0.1	0.79 ± 0.0	0.74 ± 0.0	0.70 ± 0.1	0.74 ± 0.0	0.66 ± 0.0	0.58 ± 0.0	0.63 ± 0.0
	census (easy)			census (easy)			census (easy)		
Gold	0.90 ± 0.3	0.89 ± 0.0	0.86 ± 0.1	0.90 ± 0.3	0.89 ± 0.0	0.86 ± 0.1	0.90 ± 0.3	0.89 ± 0.0	0.86 ± 0.1
Baseline	0.81 ± 0.0	0.79 ± 0.0	0.80 ± 0.0	0.77 ± 0.2	0.73 ± 0.1	0.76 ± 0.1	0.65 ± 0.1	0.42 ± 0.0	0.63 ± 0.1
S. Mendr	**0.86 ± 0.1**	**0.88 ± 0.0**	**0.85 ± 0.0**	**0.80 ± 0.0**	**0.78 ± 0.3**	**0.79 ± 0.0**	**0.79 ± 0.0**	**0.77 ± 0.1**	**0.75 ± 0.0**
SSL	0.83 ± 0.1	0.80 ± 0.1	0.83 ± 0.1	0.77 ± 0.0	0.74 ± 0.0	0.77 ± 0.0	0.71 ± 0.1	0.70 ± 0.1	0.65 ± 0.0
	connect4 (easy)			connect4 (medium)			connect4 (hard)		
Gold	0.67 ± 0.0	0.66 ± 0.2	0.61 ± 0.2	0.67 ± 0.0	0.66 ± 0.2	0.61 ± 0.2	0.67 ± 0.0	0.66 ± 0.2	0.61 ± 0.2
Baseline	0.56 ± 0.0	0.51 ± 0.0	0.39 ± 0.0	0.50 ± 0.0	0.48 ± 0.0	0.28 ± 0.0	0.23 ± 0.1	0.09 ± 0.1	0.07 ± 0.1
S. Mendr	**0.65 ± 0.0**	**0.66 ± 0.0**	**0.57 ± 0.1**	**0.60 ± 0.0**	**0.59 ± 0.1**	**0.54 ± 0.0**	**0.56 ± 0.1**	**0.42 ± 0.0**	**0.49 ± 0.0**
SSL	0.61 ± 0.1	0.53 ± 0.1	0.55 ± 0.0	0.51 ± 0.1	0.52 ± 0.0	0.45 ± 0.1	0.40 ± 0.1	0.29 ± 0.0	0.35 ± 0.2
	german (easy)			german (medium)			german (hard)		
Gold	0.95 ± 0.3	0.92 ± 0.0	0.94 ± 0.4	0.95 ± 0.3	0.92 ± 0.0	0.94 ± 0.4	0.95 ± 0.3	0.92 ± 0.0	0.94 ± 0.4
Baseline	0.78 ± 0.0	0.79 ± 0.0	0.74 ± 0.1	0.68 ± 0.2	0.74 ± 0.0	0.70 ± 0.0	0.55 ± 0.0	0.46 ± 0.0	0.63 ± 0.1
S. Mendr	**0.91 ± 0.0**	0.86 ± 0.0	**0.92 ± 0.0**	**0.84 ± 0.0**	**0.85 ± 0.1**	**0.83 ± 0.0**	**0.79 ± 0.0**	**0.80 ± 0.1**	**0.83 ± 0.0**
SSL	0.87 ± 0.2	**0.88 ± 0.0**	0.83 ± 0.1	0.76 ± 0.2	0.74 ± 0.0	0.81 ± 0.1	0.76 ± 0.0	0.73 ± 0.1	0.70 ± 0.1

However, the results illustrate that Smart Mendr outperformed SSL in most of the datasets. Even though SSL slightly surpassed the proposed approach in the APS failure and activity datasets by 1.16% and 2.90% in the medium setup, respectively, these improvements are not statistically significant (Table 5.7). Moreover, Smart Mendr manages to outperform SSL in the same datasets in the hard setting by 16.67% and 25.93%, respectively. Since Smart Mendr applies a preliminary phase of ensemble learning to produce predictions for the unlabeled points, the generative model

Table 5.6: F1 measure for different levels of incomplete supervision (II)

	SVM	KNN	Logit	SVM	KNN	Logit	SVM	KNN	Logit
	HTRU2 (easy)			HTRU2 (medium)			HTRU2 (hard)		
Gold	0.95 ± 0.5	0.91 ± 0.0	0.93 ± 0.3	0.95 ± 0.5	0.91 ± 0.0	0.93 ± 0.3	0.95 ± 0.5	0.91 ± 0.0	0.93 ± 0.3
Baseline	0.81 ± 0.0	0.87 ± 0.2	0.8 ± .1)	0.72 ± 0.1	0.72 ± 0.0	0.74 ± 0.0	0.41 ± 0.2	0.39 ± 0.4	0.32 ± 0.3
S. Mendr	**0.92 ± 0.1**	**0.97 ± 0.0**	**0.87 ± 0.1**	**0.83 ± 0.1**	**0.85 ± 0.1**	**0.91 ± 0.0**	**0.83 ± 0.1**	**0.79 ± 0.0**	**0.86 ± 0.0**
SSL	0.73 ± 0.0	0.85 ± 0.0	0.81 ± 0.0	0.76 ± 0.0	0.78 ± 0.2	0.83 ± 0.0	0.56 ± 0.0	0.66 ± 0.1	0.72 ± 0.1
	MoCap (easy)			MoCap (medium)			MoCap (hard)		
Gold	0.92 ± 0.1	0.90 ± 0.2	0.93 ± 0.1	0.92 ± 0.1	0.90 ± 0.2	0.93 ± 0.1	0.92 ± 0.1	0.90 ± 0.2	0.93 ± 0.1
Baseline	0.80 ± 0.0	0.75 ± 0.1	0.78 ± 0.0	0.77 ± 0.1	0.69 ± 0.0	0.76 ± 0.0	0.62 ± 0.0	0.22 ± 0.1	0.50 ± 0.1
S. Mendr	**0.91 ± 0.0**	**0.87 ± 0.0**	**0.92 ± 0.1**	**0.84 ± 0.1**	**0.80 ± 0.1**	0.82 ± 0.1	**0.81 ± 0.1**	**0.76 ± 0.0**	**0.82 ± 0.0**
SSL	0.83 ± 0.1	0.80 ± 0.1	0.87 ± 0.1	0.81 ± 0.0	0.77 ± 0.1	**0.83 ± 0.0**	0.67 ± 0.2	0.70 ± 0.1	0.72 ± 0.0
	penbased (easy)			penbased (medium)			penbased (hard)		
Gold	0.96 ± 0.0	0.89 ± 0.4	0.94 ± 0.2	0.96 ± 0.0	0.89 ± 0.4	0.94 ± 0.2	0.96 ± 0.0	0.89 ± 0.4	0.94 ± 0.2
Baseline	0.88 ± 0.0	0.79 ± 0.2	0.71 ± 0.0	0.86 ± 0.1	0.59 ± 0.0	0.48 ± 0.1	0.74 ± 0.0	0.32 ± 0.0	0.34 ± 0.0
S. Mendr	**0.95 ± 0.1**	**0.85 ± 0.1**	**0.93 ± 0.0**	**0.90 ± 0.2**	**0.80 ± 0.0**	**0.86 ± 0.0**	**0.85 ± 0.0**	**0.77 ± 0.1**	**0.83 ± 0.0**
SSL	0.89 ± 0.0	0.80 ± 0.0	0.90 ± .1)	0.87 ± 0.1	**0.80 ± 0.0**	0.76 ± 0.0	0.77 ± 0.1	0.63 ± 0.1	0.66 ± 0.0
	shoppers intention (easy)			shoppers intention (medium)			shoppers intention (hard)		
Gold	0.94 ± 0.0	0.91 ± 0.4	0.90 ± 0.0	0.94 ± 0.0	0.91 ± 0.4	0.90 ± 0.0	0.94 ± 0.0	0.91 ± 0.4	0.90 ± 0.0
Baseline	0.81 ± 0.0	0.78 ± 0.0	0.82 ± 0.0	0.79 ± 0.2	0.64 ± 0.0	0.74 ± 0.0	0.61 ± 0.0	0.17 ± 0.2	0.32 ± 0.0
S. Mendr	**0.92 ± 0.1**	**0.87 ± 0.0**	**0.89 ± 0.0**	**0.90 ± 0.1**	**0.82 ± 0.1**	**0.85 ± 0.1**	**0.83 ± 0.0**	**0.78 ± 0.0**	**0.74 ± 0.0**
SSL	0.88 ± 0.0	0.81 ± 0.1	0.86 ± 0.0	0.86 ± 0.0	0.73 ± 0.1	0.77 ± 0.1	0.65 ± 0.1	0.61 ± 0.0	0.61 ± 0.1
	shuttle (easy)			shuttle (medium)			shuttle (hard)		
Gold	0.97 ± 0.3	0.91 ± 0.4	0.92 ± 0.0	0.97 ± 0.3	0.91 ± 0.4	0.92 ± 0.0	0.97 ± 0.3	0.91 ± 0.4	0.92 ± 0.0
Baseline	0.87 ± 0.1	0.83 ± 0.0	0.73 ± 0.1	0.82 ± 0.1	0.80 ± 0.1	0.56 ± 0.1	0.78 ± 0.1	0.37 ± 0.0	0.31 ± 0.0
S. Mendr	**0.94 ± 0.0**	**0.87 ± 0.1**	0.88 ± 0.0	**0.86 ± 0.1**	**0.82 ± 0.2**	**0.85 ± 0.0**	**0.84 ± 0.1**	**0.76 ± 0.0**	**0.81 ± 0.1**
SSL	0.90 ± 0.0	0.85 ± 0.0	**0.90 ± 0.2**	0.83 ± 0.0	0.81 ± 0.1	0.80 ± 0.1	0.79 ± 0.1	0.69 ± 0.2	0.62 ± 0.0
	statlog (easy)			statlog (medium)			statlog (hard)		
Gold	0.99 ± 0.3	0.97 ± 0.3	0.91 ± 0.0	0.99 ± 0.3	0.97 ± 0.3	0.91 ± 0.0	0.99 ± 0.3	0.97 ± 0.3	0.91 ± 0.0
Baseline	0.81 ± 0.1	0.84 ± 0.1	0.65 ± 0.0	0.70 ± 0.0	0.80 ± 0.2	0.47 ± 0.0	0.59 ± 0.2	0.56 ± 0.1	0.16 ± 0.2
S. Mendr	**0.97 ± 0.1**	**0.93 ± 0.0**	**0.90 ± 0.0**	**0.89 ± 0.0**	**0.90 ± 0.0**	**0.82 ± 0.1**	**0.88 ± 0.2**	**0.81 ± 0.0**	**0.79 ± 0.0**
SSL	0.91 ± 0.0	0.91 ± 0.0	0.84 ± 0.0	**0.89 ± 0.0**	0.81 ± 0.2	0.78 ± 0.1	0.69 ± 0.0	0.69 ± 0.0	0.64 ± 0.1
	twonorm (easy)			twonorm (medium)			twonorm (hard)		
Gold	0.98 ± 0.2	0.96 ± 0.2	0.95 ± 0.3	0.98 ± 0.2	0.96 ± 0.2	0.95 ± 0.3	0.98 ± 0.2	0.96 ± 0.2	0.95 ± 0.3
Baseline	0.85 ± 0.1	0.81 ± 0.2	0.79 ± 0.1	0.71 ± 0.0	0.79 ± 0.0	0.54 ± 0.1	0.52 ± 0.0	0.74 ± 0.0	0.36 ± 0.1
S. Mendr	**0.95 ± 0.0**	**0.94 ± 0.0**	**0.92 ± 0.0**	**0.87 ± 0.0**	**0.89 ± 0.0**	**0.87 ± 0.0**	**0.87 ± 0.0**	**0.80 ± 0.1**	**0.79 ± 0.0**
SSL	0.94 ± 0.0	0.87 ± 0.1	0.89 ± 0.1	0.79 ± 0.2	0.81 ± 0.1	0.81 ± 0.0	0.77 ± 0.0	0.73 ± 0.1	0.74 ± 0.1
	yeast (easy)			yeast (medium)			yeast (hard)		
Gold	0.91 ± 0.0	0.92 ± 0.0	0.95 ± 0.0	0.91 ± 0.0	0.92 ± 0.0	0.95 ± 0.0	0.91 ± 0.0	0.92 ± 0.0	0.95 ± 0.0
Baseline	0.80 ± 0.0	0.79 ± 0.1	0.77 ± 0.1	0.71 ± 0.0	0.61 ± 0.0	0.70 ± 0.0	0.53 ± 0.0	0.48 ± 0.0	0.62 ± 0.0
S. Mendr	**0.87 ± 0.1**	**0.90 ± 0.0**	**0.93 ± 0.0**	**0.80 ± 0.1**	**0.83 ± 0.1**	**0.86 ± 0.0**	**0.75 ± 0.0**	**0.78 ± 0.0**	**0.80 ± 0.1**
SSL	0.86 ± 0.2	0.85 ± 0.2	0.84 ± 0.0	0.79 ± 0.0	0.73 ± 0.2	0.83 ± 0.0	0.65 ± 0.0	0.72 ± 0.1	0.69 ± 0.1

can operate with more accurate sources. Hence, the accuracy of the generated labels is enhanced so the classifiers can achieve better generalization.

Also, the results of comparing Smart Mendr with the baseline models and the SSL technique are tested using the Wilcoxon signed ranks test and shown in Table 5.7. The table demonstrates that the classification performance achieved by the proposed method is significantly statistically better

Table 5.7: P-values of paired Wilcoxon signed ranks test in incomplete supervision experiments

	Classification Performance (easy)		Classification Performance (medium)		Classification Performance (hard)	
	Baseline	SSL	Baseline	SSL	Baseline	SSL
Smart Mendr	$5.12\times e^{-9}$	$7.47\times e^{-9}$	$5.16\times e^{-9}$	$5.66\times e^{-8}$	$5.16\times e^{-9}$	$5.1\times e^{-9}$
	Labeling accuracy (easy)		Labeling accuracy (medium)		Labeling accuracy (medium)	
Smart Mendr	-	$6.40\times e^{-4}$	-	$6.42\times e^{-4}$	-	$6.52\times e^{-4}$

than the base models and SSL. Overall, the results show that the classifiers built using Smart Mendr are more robust and tend to maintain similar classification performance with different setups of missing labels.

5.4.4.2 Labeling Accuracy

Moreover, the experiments report the labeling accuracy calculated based on the ground truth provided in the original datasets. The labeling accuracy is measured as the ratio between the number of correctly labeled instances to the size of D_u. The average of labeling accuracies achieved by the proposed method and the SSL technique are presented in Table 5.8. The table shows that the proposed method manages to produce more accurate labels than SSL in all the datasets. Although SSL achieves a high level of accuracy in most of the datasets, when the number of unlabeled data points increases, the labeling accuracy tends to drop drastically. For example, in the yeast dataset, SSL manages to initially achieve a labeling accuracy of 85.17% with the easy setup. However, the labeling accuracy declines by 10.59% and 38.82% with medium and hard settings, respectively.

On the other hand, the labeling accuracy of the proposed method shows a mild deterioration as the number of unlabeled points increases. For instance, in the avila dataset, the labeling accuracy of the proposed method only declines by 7.32% and 6.10% in the medium and the hard settings, respectively, when compared to the easy setup. Also, in some datasets, such as the statlog and the shuttle datasets, the proposed method manages to produce more accurate labels than the SSL technique by 24.03% and 30.14% in the easy setting, 24.62% and 20.29% in the medium setting, and 16.92% and 30.36% in the hard setting. Table 5.7 also reports the results of the Wilcoxon test for the comparison of labeling accuracy achieved by the proposed method against the SSL

Table 5.8: Labeling accuracy with incomplete supervision

Dataset	easy		medium		hard	
	Smart Mendr	SSL	Smart Mendr	SSL	Smart Mendr	SSL
activity	0.87	0.83	0.74	0.72	0.69	0.56
APS failure	0.81	0.79	0.83	0.69	0.8	0.64
avila	0.82	0.8	0.76	0.7	0.77	0.69
banana	0.87	0.83	0.79	0.73	0.78	0.57
census	0.91	0.75	0.78	0.72	0.73	0.66
connect4	0.82	0.71	0.77	0.72	0.74	0.7
german	0.95	0.84	0.75	0.74	0.69	0.57
HTRU2	0.86	0.84	0.8	0.68	0.72	0.62
MoCap	0.95	0.82	0.89	0.78	0.76	0.66
penbased	0.8	0.73	0.79	0.76	0.75	0.61
shoppers	0.89	0.81	0.8	0.71	0.72	0.64
shuttle	0.95	0.73	0.83	0.69	0.73	0.56
statlog	0.93	0.75	0.81	0.65	0.76	0.65
twonorm	0.9	0.79	0.86	0.72	0.79	0.67
yeast	0.89	0.85	0.82	0.76	0.75	0.52

techniques, which shows significant differences in favor of the proposed method. In general, the results conform to the fact that the proposed method does not only learn a generative model to produce predictions for the unlabeled portion of the data. It also applies meta-active learning to enhance the accuracy of the output of the generative model and improve the overall classification performance.

5.5. Related Work

There have been numerous studies [37], [38], [39], [40] to investigate learning from inaccurate supervision. For instance, some approaches try to modify existing algorithms to create more robust learning models. Gao *et al.* [37] applied a set of independent corrections to the training examples and then exploited these corrections to enhance the robustness of the KNN algorithm. Also, Kumar and Sastry [38] present a new loss function to learn neural network models with inaccurately supervised training data. The function uses the mean absolute value of the error instead of the cross-entropy or the mean-squared error, which makes it more tolerant to class noise.

On the other hand, previous studies [39], [40] tried to modify bagging or boosting algorithms to detect noise. For instance, one ensemble learning technique is proposed [39] to deal with class

noise by adjusting the agreement and disagreement rates at which the points are considered noise. The adjusting process tries to estimate the noise level before using cross-validation. Likewise, Zhang *et al.* [40] proposed a meta-learning method that applies ensemble learning to learn from weakly supervised data. However, a closer look at these techniques reveals several shortcomings. For example, one of these methods [37] does not scale for high dimensional datasets and only consider binary classification. In their experiments [37], unlike our approach, the feature space has been scaled to $X = [0, 1]^2$, and the multi-classification problems were transformed into binary ones. Also, the loss function proposed by Kumar and Sastry [38] cannot learn the conditional probability distribution of the noise presented in the data.

Moreover, unlike the proposed method, ensemble learning techniques rely on either majority or consensus voting. Although many studies [41] show that majority filtering can outperform consensus voting, with small heterogeneous ensembles, majority voting may not be that effective since agreement rates become close to consensus filtering. Also, deciding on the agreement rate is known to be a challenge in most ensemble filtering techniques [39]. Therefore, Smart Mendr tries to address this challenge by leveraging semi-supervised learning techniques to automatically learn the accuracy of the ensemble predictor and choose the threshold for noise detection.

As for learning with incomplete supervision, previous research [42], [43], [44], [45] have handled missing labels by applying semi-supervised learning. For example, one technique [42] focuses on multi-label problems with incomplete supervision. The approach propagates provided labels to induce the missing ones by building a dependency graph that considers the semantic label hierarchy. Also, dealing with missing labels, Cong *et al.* [43] propose a semi-supervised learning model by integrating matrix factorization and attribute space. Furthermore, another technique [44] presents an embedding-based method to assign labels in large-scale datasets. Likewise, Dehghani *et al.* propose an approach [45] to learn neural network architectures with weakly supervised data. The approach trains two neural networks; the first one is used to estimate the labeling confidence. Then these scores are used to control the magnitude of the gradient updates to the second network. However, these techniques focus on different learning settings; for example, some approaches [42], [43] consider multi-task problems in which multiple tasks are solved by utilizing similarities and differences between the sub-tasks. However, our settings are different since we aim to learn from incomplete and inaccurate supervision simultaneously. Moreover, most of these approaches are model-specific. For example, some techniques [17], [45] only focuses on enhancing the capability

of neural networks to handle missing labels. Finally, although some research [29], [46] tries to control faulty results of crowdsourcing using active learning, we think that more work is required to enhance the robustness of active learning with class noise introduced in the training set, which is precisely what we are resolving in this research.

5.6. Conclusions

The chapter presents a classification framework that is designed to deal with weakly supervised data. First, the proposed technique employs ensemble learning in semi-supervised settings to detect noisy points and produce initial weak labels for unlabeled data. During this phase, both the ensemble predictor and the original data are treated as two weakly supervised sources. Hence, their accuracies are estimated using maximum likelihood estimation. The output of the generative model is then utilized to determine the labeling confidence of each data point. Then, to rectify the class labels of these points and resolve incomplete supervision, the method applies an iterative process of meta-active learning to select which points should be made correct by the user to improve the classification performance. The empirical results show that the proposed method can significantly statistically outperform state-of-the-art techniques while achieving high specificity, especially with high rates of noise. The proposed method manages to detect 33% more noisy data points than the comparing techniques on average. Also, when evaluating the proposed method within incomplete supervision scenarios, the results empirically demonstrate that the proposed method can produce high-accuracy labels for the unlabeled points and outperform the semi-supervised technique by up to 26% in classification performance.

References

[1] Y.F. Li, L.Z. Guo, and Z.H. Zhou, "Towards Safe Weakly Supervised Learning," IEEE Trans. Pattern Analysis and Machine Intelligence, 2019.

[2] S.H. Bach, B.He, A. Ratner, and C. Ré, "Learning the Structure of Generative Models without Labeled Data," *Proc. ICML'17,* vol. 70, pp. 273-282, 2017.

[3] P. Cheng, X. Lian, X. Jian, and L. Chen, "FROG: A Fast and Reliable Crowdsourcing Framework," *IEEE Trans. Knowl and Data Eng*, vol. 31, no. 5, pp. 894–908, 2019.

[4] J. Luengo, S.-O. Shim, S. Alshomrani, A. Altalhi, and F. Herrera, "CNC-NOS: Class Noise Cleaning By Ensemble Filtering And Noise Scoring," *Knowledge-Based Systems*, vol. 140, pp. 27–49, 2018.

[5] D. P. Kingma, S. Mohamed, D. Jimenez Rezende, and M. Welling, "Semi-Supervised Learning With Deep Generative Models," *Adv Neural Inf Process Syst,* pp. 3581–3589, 2014.

[6] Y. Fu, X. Zhu, and B. Li, "A Survey On Instance Selection For Active Learning," *Knowl and Inf Syst*, vol. 35, no. 2, pp. 249–283, 2013.

[7] M. Poel, "Detecting Mislabeled Data Using Supervised Machine Learning Techniques," *Augmented Cognition. Neurocognition and Machine Learning*, 2017.

[8] M. Sabzevari, G. Martínez-Muñoz, and A. Suárez, "A Two-Stage Ensemble Method For The Detection Of Class-Label Noise," *Neurocomputing*, pp. 2374–2383, 2018.

[9] L.P.F. Garcia, A.C. Lorena, S. Matwin, and A.C.P.L.F. de Carvalho, "Ensembles Of Label Noise Filters: A Ranking Approach," *Data Min Knowl Disc*, 2016.

[10] D. Guan, H. Wei, W. Yuan, G. Han, Y. Tian, M. Al-Dhelaan, and A. Al-Dhelaan, "Improving Label Noise Filtering by Exploiting Unlabeled Data," *IEEE Access*, vol. 6, pp. 11154–11165, 2018.

[11] S. García, J. Luengo, and F. Herrera, "Dealing with Noisy Data," *Data Preprocessing in Data Mining*, pp. 107–145, 2015.

[12] A. Oliver, A. Odena, C.A. Raffel, E.D. Cubuk, and I. Goodfellow, "Realistic Evaluation of Deep Semi-Supervised Learning Algorithms," *Advances in Neural Information Processing Systems*, pp. 3235–3246, 2018.

[13] P. Varma, B. He, P. Bajaj, I. Banerjee, N. Khandwala, D.L. Rubin, and C. Ré, "Inferring Generative Model Structure with Static Analysis," *Adv Neural Inf Process Syst,* 2017.

[14] L.-Z. Guo, F. Kuang, Z.-X. Liu, Y.-F. Li, N. Ma, and X.-H. Qie, "Weakly Supervised Learning Meets Ride-Sharing User Experience Enhancement," *arXiv preprint arXiv:2001.09027*, 2020.

[15] Z.-Y. Zhang, P. Zhao, Y. Jiang, and Z.-H. Zhou, "Learning from Incomplete and Inaccurate Supervision," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, New York, NY, USA, 2019, pp. 1017–1025.

[16] K. Chen, D. Guan, W. Yuan, B. Li, A. M. Khattak, and O. Alfandi, "A Novel Feature Selection-Based Sequential Ensemble Learning Method for Class Noise Detection in High-Dimensional Data," *Adv Data Mining and Applications*, pp. 55–65, 2018.

[17] R. Saman, A. Ali, and J. Licheng, "Rough-KNN Noise-Filtered Convolutional Neural Network for Image Classification," *Frontiers in Artificial Intelligence and Applications*, pp. 265–275, 2019.

[18] J. A. Sáez, M. Galar, J. Luengo, and F. Herrera, "INFFC: An Iterative Class Noise Filter Based On

The Fusion Of Classifiers With Noise Sensitivity Control," *Information Fusion*, vol. 27, pp. 19–32, 2016.

[19] B. Frenay and M. Verleysen, "Classification in the Presence of Label Noise: A Survey," *IEEE Trans. Neural Netw. Learn. Syst*, vol. 25, no. 5, pp. 845–869, 2014.

[20] C.J. Mantas, J. Abellán, and J.G. Castellano, "Analysis Of Credal-C4.5 For Classification In Noisy Domains," *Expert Systems with Applications*, 2016.

[21] Q. Miao, Y. Cao, G. Xia, M. Gong, J. Liu, and J. Song, "RBoost: Label Noise-Robust Boosting Algorithm Based on a Nonconvex Loss Function and the Numerically Stable Base Learners*," IEEE Trans. Neural Netw. Learn. Syst*, pp. 2216–2228, 2016.

[22] P. Yang, J.T. Ormerod, W. Liu, C. Ma, A.Y. Zomaya, and J.Y.H. Yang, "AdaSampling for Positive-Unlabeled and Label Noise Learning with Bioinformatics Applications," *IEEE Trans. Cybern*, vol. 49, 2019.

[23] X. Liu, D. Zachariah, J. Wågberg, and T.B. Schön, "Reliable Semi-Supervised Learning when Labels are Missing at Random," arXiv:1811.10947 [cs, stat], 2019.

[24] V. Jain, N. Modhe, and P. Rai, "Scalable Generative Models for Multi-label Learning with Missing Labels," *Proc. Machine Learning Research*, pp. 1636–1644, 2017.

[25] B. Du, T. Xinyao, Z. Wang, L. Zhang, and D. Tao, "Robust Graph-Based Semisupervised Learning for Noisy Labeled Data via Maximum Correntropy Criterion," *IEEE Trans. Cybern*, vol. 49, pp. 1440–1453, 2019.

[26] Y. Ding, S. Yan, Y. Zhang, W. Dai, and L. Dong, "Predicting The Attributes Of Social Network Users Using A Graph-Based Machine Learning Method," *Computer Communications*, vol. 73, pp. 3–11, Jan. 2016.

[27] Z.-H. Zhou, "A Brief Introduction To Weakly Supervised Learning," *National Science Review*, vol. 5, no. 1, pp. 44–53, 2017.

[28] M. E. Ramirez-Loaiza, M. Sharma, G. Kumar, and M. Bilgic, "Active Learning: An Empirical Study Of Common Baselines," *Data Mining and Knowledge Discovery,* vol. 31, no. 2, pp. 287–313, 2017.

[29] M.R. Bouguelia, S. Nowaczyk, K.C. Santosh, and A. Verikas, "Agreeing To Disagree: Active Learning With Noisy Labels Without Crowdsourcing," *Int. J. Mach. Learn. & Cyber.*, vol. 9, no. 8, pp. 1307–1319, 2018.

[30] C.H. Lin, M. Mausam, and D.S. Weld, "Active Learning with Unbalanced Classes and Example-Generation Queries," *Proc. Sixth AAAI Conf. on Human Computation and Crowdsourcing*, 2018.

[31] Y. Yang, Z. Ma, F. Nie, X. Chang, and A.G. Hauptmann, "Multi-Class Active Learning by Uncertainty Sampling with Diversity Maximization," *International Journal of Computer Vision*, vol. 113, pp. 113–127, 2015.

[32] R. C. Prati, J. Luengo, and F. Herrera, "Emerging Topics And Challenges Of Learning From Noisy Data In Nonstandard Classification: A Survey Beyond Binary Class Noise," *Knowl Inf Syst*, vol. 60, pp. 63–97, 2019.

[33] M. Sabzevari, G. Martínez-Muñoz, and A. Suárez, "Small Margin Ensembles Can Be Robust To Class-Label Noise," *Neurocomputing*, vol. 160, pp.18-33, 2015.

[34] M. Nashaat, A. Ghosh, J. Miller, S. Quader, and C. Marston, "M-Lean: An End-To-End Development Framework For Predictive Models In B2B Scenarios," *Information and Software Technology*, vol. 113, 2019.

[35] M. Nashaat, A. Ghosh, J. Miller, S. Quader, C. Marston, and J. Puget, "Hybridization of Active Learning and Data Programming for Labeling Large Industrial Datasets," *Proc. 2018 IEEE Conf. on Big Data*, 2018.

[36] J. Demšar, "Statistical Comparisons of Classifiers over Multiple Data Sets," *Machine Learning Research*, vol. 7, pp. 1–30, 2006.

[37] W. Gao, B.B. Yang, and Z.H. Zhou, "On the Resistance of Nearest Neighbor to Random Noisy Labels," arXiv:1607.07526 [cs], 2018.

[38] H. Kumar and P.S. Sastry, "Robust Loss Functions for Learning Multi-class Classifiers," *2018 IEEE Conf. on Systems, Man, and Cybernetics (SMC)*, pp. 687–692, 2018.

[39] M. Sabzevari, G. Martínez-Muñoz, and A. Suárez, "Vote-Boosting Ensembles," *Pattern Recognition*, vol. 83, pp. 119–133, Nov. 2018.

[40] J. Zhang, M. Wu, and V.S. Sheng, "Ensemble Learning from Crowds," *IEEE Trans. Knowl and Data Eng*, vol. 31, pp. 1506–1519, 2019.

[41] M. R. Smith and T. Martinez, "The Robustness Of Majority Voting Compared To Filtering Misclassified Instances In Supervised Classification Tasks," *Artif Intell Rev*, vol. 49, no. 1, pp. 105–130, 2018.

[42] B. Wu, F. Jia, W. Liu, B. Ghanem, and S. Lyu, "Multi-Label Learning With Missing Labels Using Mixed Dependency Graphs," *Int J Comput Vis*, pp. 875–896, 2018.

[43] Y. Cong, G. Sun, J. Liu, H. Yu, and J. Luo, "User Attribute Discovery With Missing Labels," *Pattern Recognition*, vol. 73, pp. 33–46, 2018.

[44] A.H. Akbarnejad and M.S. Baghshah, "An Efficient Semi-Supervised Multi-label Classifier Capable of Handling Missing Labels," *IEEE Trans. Knowl and Data Eng*, vol. 31, no. 2, pp. 229–242, Feb. 2019.

[45] M. Dehghani, A. Severyn, S. Rothe, and J. Kamps, "Learning to Learn from Weak Supervision by Full Supervision," *Proc. NIPS Workshop on Meta-Learning, 2017*.

[46] C. Li, L. Jiang, and W. Xu, "Noise Correction To Improve Data And Model Quality For Crowdsourcing," *Engineering Applications of Artificial Intelligence*, 2019.

Chapter 6 : Transformers Meet Tabular Data: Bidirectional Representation Model for Erroneous Data Detection

6.1. Introduction

Data and analytics have come one of the top growth opportunities for business. Data-driven decision making has proven to lead to better accountability for every organization. However, poor data quality can have adverse impacts on businesses and cause significant financial loss [1]. Thus, data cleansing has become an essential prerequisite for developing any business intelligence solutions. Data cleansing refers to the process of identifying and rectifying inaccurate records in databases. The procedure involves two practices intending to produce high-quality data. First, erroneous data cells are identified, and then data correction routines [2] are applied to fix or remove the corrupted data. Typically, errors originate from diverse sources such as syntax errors, type conversion, and duplicate values. Therefore, error detection can be challenging, especially when dealing with big data [1], which makes manual error detection prohibitively expensive. Data quality issues are considered as the main enemy for machine learning and analytics. Since "garbage-in, garbage-out" formed an ongoing threat for machine learning models, inaccurate data are proven to have severe consequences for businesses [3]. Therefore, error detection is considered as a critical step to maintain a stable analytics pipeline.

As a result, there have been numerous studies to investigate automating the process of detecting erroneous data. Much research is targeted to handle outlier detection [4]–[7], rule violations [8], [9], and duplicate data detection [10]. Rule-based systems [8], [9] count on the identification of a set of data quality rules using integrity constraints [11] to specify functional dependencies or other constraints that may define data quality in the given domain. Although these techniques are proven to be effective in many situations [12], they cannot be considered as conclusive for many reasons. First, each of these methods is customized only to detect specific types of erroneous data. Hence, their performance is not guaranteed in many situations in which diverse forms of errors coexist in the same database [13]. Second, some of these approaches are only effective with particular

configurations regarding the examined data. For example, most outlier detection methods are susceptible to imbalanced distributions or high dimensional datasets [7]. With large datasets in high-dimensional space, classifiers cannot separate outliers from the original data using the limited number of outliers available during training. Third, as previous research [13] points out, most of these systems are evaluated only using synthetic data, which might not be enough to test their suitability in real-world situations.

Finally, all of these solutions still require some input from the end-user. For example, rule-based systems [8], [9] oblige the user to write integrity constraints [9], such as denial constraints [2]. Then, these systems utilize these rules to detect violating cells that do not comply with these specified rules. However, writing integrity constrains requires an adequate level of domain knowledge alongside the technical expertise needed to write such regular expressions [9]. Also, outlier detection methods require precise identification of outlier thresholds. Existing thresholding techniques rely on statistics, which make them considerably biased when dealing with data with many outliers [14]. Hence, end-users input may be needed to evaluate the choice of these thresholds, which can be a time-consuming task.

Alternatively, while trying to address some of these challenges, some research [3], [15], [16] has recently investigated the effectiveness of applying machine learning to the problem of error detection. Since detecting erroneous cells can be seen as a binary classification problem, a learning model can be trained to differentiate faulty values from correct ones. Furthermore, the expressive power of sophisticated models such as neural networks can overcome the problem of error heterogeneity and detecting multiple classes of errors. Additionally, except for training data, learning models do not require additional input from the user. However, several challenges regarding applying machine learning to error detection remain to be addressed. For instance, previous techniques [16], [17] employ supervised learning and hence, require a considerable amount of labeled data to train such models. Alternatively, even though some techniques [17], [18] apply sampling strategies to reduce the volume of labeled examples, the burden required for feature engineering is believed to be substantial [16].

A closer look at the sources of errors, however, states that attention [19] matters. Attention mechanisms [19] is a recent technique that is mainly targeted at representation learning. Attention-based networks consider the dependency relationships between different parts of the input vector.

	ID	Airport Name	City	State
l_1	10551	Bethel Airport	Beathel	AK
l_2	14709	Deadhorse	Deadhorse	AM
l_3	12992	Adams Field	Little Rock	AR
l_4	10800	Bob Hope	Burbank	AR

Figure 6.1: An example dataset with errors

Thus, it learns interdependent representations, which are essential to solving many tasks such as speech recognition [20] and document summarization [21]. Comparably, when considering tabular data, attentive models can observe different levels of dependencies between the input features, which can be effectively employed to detect erroneous data.

As an example, Figure 6.1 shows a snippet of the Airports table from the Flights database [22]. Examples for misspelled values are shown in the figure (i.e., the city name in l_1 and the state name in l_2 should be spelled as "Bethel" and "AR", respectively). Also, a value swapping error appears in l_4 (i.e., Burbank is in California state (CA) not in Arkansas (AR)). Most of these errors go beyond traditional rule-based systems since the errors cannot be detected using traditional integrity constraints. One expensive solution to catch such errors is to provide the dataset with many constant conditional functional dependencies such as [Airport_Name ([airport name = "Bethel Airport"] → [city = "Beathel"]) and as [City ([city = "Burbank"] → [state = "AR"]).

Alternatively, since these errors are related to the data context, we believe that an attentive based network can employ data representation to reflect on inter-attribute dependencies and find these errors. Hence, inspired by the significant improvements that attention techniques have achieved in language understanding tasks [23], we introduce, TabReformer, a model that applies unsupervised representation learning to model attribute dependencies in tabular data. A component overview of the proposed framework is illustrated in Figure 6.2. As the figure shows, the model has two main phases. The first phase trains a bidirectional encoder representation model by a Masked Data Model (MDM) objective. In this phase, we randomly replace a percentage of the input features with a special masked token. Then, the model is trained to classify the masked cells. In the second phase, we fine-tune the system parameters with the task of erroneous data detection. To minimize

the manual effort in providing training data, the system applies data augmentation that takes a set of correct data points and returns erroneous synthetic examples.

To evaluate the proposed model, we compare its performance with four state-of-the-art techniques for error detection and data repairing [3], [9], [16], [17]. The primary contributions of this research can be summarized as follows:

- An end-to-end framework is introduced for self-supervised learning for structured data. The system applies bidirectional encoder representations to model the data and detect erroneous values. The architecture of the proposed method includes a novel learning objective for tabular data along with a data augmentation module. The system does not require any user-defined parameters; that is, it is fully-automated and assumes no domain-specific knowledge! Instead, the transformation functions and the augmentation strategy are concluded from the input data. The code of the framework, along with the trained models created during the evaluation are publicly available at https://github.com/MonaNashaat/TabReformer.
- We apply an extensive set of experiments to evaluate the proposed system against state-of-the-art techniques. The evaluation uses six datasets of varying size, dimensionality, and error

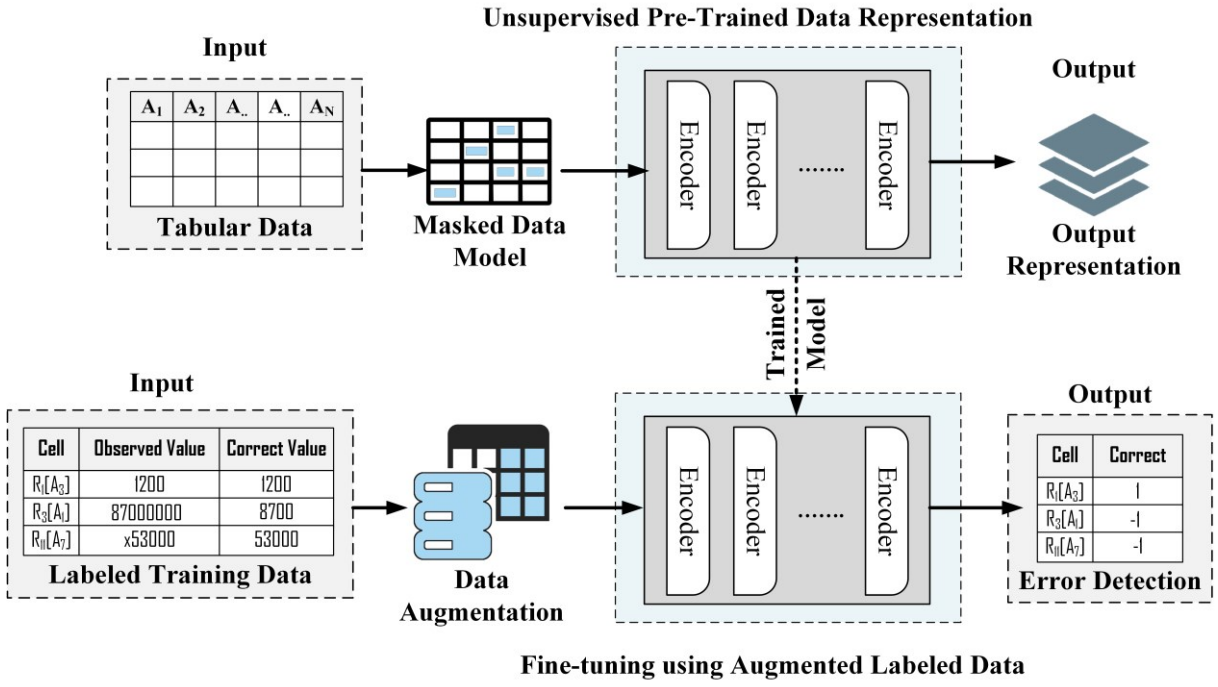


Figure 6.2: A component overview of TabReformer

distributions. The experiments also involve a micro-benchmark to evaluate the impact of different design decisions that are implemented in the proposed method.

The chapter is structured as follows: In Section 6.2, we present an overview of the background related to this research. We then describe the individual components of TabReformer (Section 6.3). Section 6.4 defines the evaluation setup and experimental results. While Section 6.5 reviews related work; and Section 6.6 concludes the chapter.

6.2. Background

In this section, we review methods for error detection; and discuss data augmentation for resolving data imbalance. Finally, we present transformers as a new architecture of attentive-based neural networks that have been gaining popularity in many applications, such as machine translation and language modeling.

6.2.1. Error Detection

There has been extensive research on error detection and data cleaning algorithms to identify and repair possible errors in data. According to the error sources, we categorize existing error detection methods into two main categories: (1) rule-based and pattern-based methods and (2) quantitative methods. Rule-based methods rely on a set of data quality rules and use them to specify which data cells violate these rules. Denial constraints [9] can be used to determine data quality rules in the form of first-order formulae that incorporate different types of integrity constraints. These constraints can be either supplied by domain experts [8], [24], or (potentially) automatically derived from the data [25].

Consequently, existing tools [8], [9] focus on analyzing these constraints and defining data inconsistencies with these rules. For example, Schelter *et al.* [8] propose a declarative API that allows a user to define database constraints. Then, the approach executes an algorithm for constraint validation to detect violating data. Similarly, Dallachiesa *et al.* [9] proposed NADEEF as a prototype that follows a similar pipeline of collecting user-specified constraints. Then, these constraints are compiled to detect erroneous data and select the most appropriate data repair algorithm.

Alternatively, qualitative and pattern-based methods characterize data by using pattern mining techniques. Pattern mining approaches attempt to discover the syntactic and semantic characterizations of the data. One technique for pattern discovery is inducing functional dependencies from the data [26]–[28]. Functional dependencies are considered a special form of denial constraints [26] and are commonly used to specify business rules. For example, tuples with the same value for longitude must share the same time zone. Existing research [27] has studied repeated patterns in the data, and formalize them into functional dependencies to suggest better repair solutions. Another study [28] focuses on deriving such dependencies with the presence of erroneous data; the method [28] introduces a new class of integrity constraints that can infer dependencies between data attributes, even if a portion of the attributes violates these dependencies.

Quantitative methods employ statistical techniques to identify unusual behavior in the data. One good example of such techniques is outlier detection. Existing research [4]–[7] applies data modeling approaches to detect outliers in numerical data, e.g., Gaussian mixture models [4] or histogram modeling [5]. Moreover, recent research has applied machine learning techniques, such as unsupervised learning [6] and active learning [7], to detect outliers in relational databases. For example, Riahi *et al.* [6] propose a technique to learn a model for outlier detection using Bayesian networks. The method integrates exception mining with statistical-relational learning to detect outliers in relational data.

However, there are vital questions that are still not addressed in these approaches. For example, all these techniques require end-users intervention in various time-consuming and non-trivial stages along the way. For example, rule-based systems require users to provide inputs such as algorithm configuration, data quality constraints, and the verification of final results [13]. Although some efforts [11], [29] try to derive denial constraints automatically, these approaches still depend on the user to provide an appropriate error threshold. Moreover, these tools can be computationally costly due to the enormous search space of the constraints [11]. Also, since each of these techniques is designed to deal with specific types of errors, real-world applications may require using a combination of these detection methods. However, integrating the outputs of these tools requires significant engineering, which often becomes the user's responsibility. Finally, the performance of these combinations depends on the weights assigned to each and every result for

each technique. Thus, the robustness of these methods to capture errors in real-world databases still requires to be verified, which provides the motivation of this research.

6.2.2. Data Augmentation

Data augmentation is an approach that allows practitioners to economically generate data to enhance the input variety (and volume) presented to machine learning models. Typically, neural networks require a massive amount of labeled data to model the underlying distribution of the general population. Training deep learning models with small training dataset can result in overfitting; in such a scenario, the model memorizes the input examples and their corresponding outputs. Therefore, adding more data (different) examples offers a broader description of the general population from which the model can be learned. Hence, data augmentation presents a reasonable solution for obtaining more training examples when acquiring real labeled data, which can be time-consuming or prohibitively expensive.

Data augmentation assumes that more information can be obtained by applying a set of transformations to the original dataset. Typically, data augmentation consists of two elements: (1) a set of transformation functions that, when applied to the original data, can generate additional examples; and, (2) a data augmentation strategy that determines how these functions should be applied to the data. Many approaches [30], [31] are proposed to specify augmentation policies for different classification tasks. For example, Cubuk *et al.* [31] present a search algorithm to find data augmentation strategies automatically. The algorithm applies reinforcement learning [32] to finds the optimal policy among a predefined set of geometric transformations. However, most of these techniques are focused on specific applications such as image analysis [31] or speech recognition [33].

Moreover, several questions are raised regarding the cost of these approaches [30], [31]. For instance, previous studies [34], [35] stated that these algorithms [31] require training a massive number of models, which can take thousands of GPU hours. As a result, this research aims to investigate the usefulness of data augmentation approaches for structured tabular data to reduce manual efforts in the context of erroneous data detection.

6.2.3. Transformers

Transformers are a novel (neural network) architecture that was recently presented [19]. The architecture applies an attention-mechanism [36] to enable transformers to understand complex structures such as natural language. The attention-mechanism was initially proposed for machine translation tasks, so it can process an input sentence and decide, for each input token, which other parts of the input are essential. Consequently, the mechanism extracts keywords that are important to sentence semantics. Thus, the network can execute translation more effectively. Moreover, self-attention [23] is an attention mechanism that aims to derive a representation of an input sequence by estimating relationships between items in this sequence. This mechanism has shown significant advances in natural language processing, such as abstractive summarization [21] and language modeling [23].

Similarly, transformers follow the same structure of sequence-to-sequence models [37] by utilizing an encoder-decoder architecture. The encoder processes the input and maps it to a single latent vector denoting the whole input sequence. The input first goes through a self-attention layer to allow the encoder to look at each word (token) in the input sequence. Then, the output of the self-attention layer is passed to feed-forward (neural) networks, which process each of these encodings individually [19]. The output of the encoder is then fed to a decoder, which unpacks the encoding into a target sequence (e.g., the same sentence translated in a different language). The decoder has a similar structure to the encoder; however, it has an additional attention layer that enables the decoder to focus on relevant parts of the input sentence.

Until now, transformers have shown improvements in many tasks, including question and answering, machine translation [38], and language understanding [23]. One example of such a transformer is BERT [23]. BERT applies an encoder representation using transformers to execute bidirectional training for language modeling. Motivated by BERT, recent research [39] has examined different configurations for transformer networks to enhance their capabilities. For example, Dai *et al.* [39] propose an enhancement that allows transformers to learn language model beyond fixed-length contexts. Also, other research [40], [41] have proposed some design changes to enhance BERT's performance. While some of these changes [40] aim to reduce the number of parameters to enhance memory consumption [41], others [40] modified BERT's hyperparameters to enhance the overall performance. All these models [23], [39]–[41] have focused on language

modeling – to the best of our knowledge – no previous research has investigated the effectiveness of attention-based models for error detection in tabular data.

6.3. TabReformer: The Proposed Framework

In the following subsections, we describe the architecture of TabReformer. Section 6.3.1 formulates the problem statement for error detection in databases; Section 6.3.2 describes in detail the phases of the proposed solution.

6.3.1. Problem Statement

TabReformer aims at classifying erroneous values in a database. Relational databases formally consist of a set of tables, while each table D comprises: a set of attributes $A = \{\mathrm{a_i}\}_{\mathrm{i=1}}^{\mathrm{N}}$ (columns), and tuples $L = \{l_\mathrm{i}\}_{\mathrm{i=1}}^{\mathrm{M}}$ (rows). Each tuple l contains a set of cells as $\mathrm{C}_l = \{l[\mathrm{a}_1], l[\mathrm{a}_2], .. l[\mathrm{a}_\mathrm{N}]\}$ where C_l represents the cells in l, and $l[\mathrm{a_i}]$ denotes the value of the i[th] attribute in l. Also, $\mathrm{C}_l \subset \mathrm{C}$ where $\mathrm{C} = \{\mathrm{c_i}\}_{\mathrm{i=1}}^{\mathrm{N\times M}}$ designates all the cells in D. Since erroneous entries originate from assigning incorrect values (including missing values) to table cells, we assume that each cell $c_i \in \mathrm{C}$ has a correct value $\bar{\mathrm{v}}_{\mathrm{c_i}}$ and an existing observed value $\mathrm{v}_{\mathrm{c_i}}$. Then, for each cell $c_i \in \mathrm{C}$, a cell c_i is said to be erroneous if $\bar{\mathrm{v}}_{\mathrm{c_i}} \neq \mathrm{v}_{\mathrm{c_i}}$.

Moreover, the model employs a training dataset $\mathrm{D_t}$ in the second phase. The training dataset is denoted as $\{\mathbf{x}_\mathrm{i}, \mathrm{y_i}\}_{\mathrm{i=1}}^{\mathrm{K}}$, where $\mathbf{x}_\mathrm{i}$ depicts a set of features representing a given cell as $\{c_i, \mathrm{v}_{\mathrm{c_i}}, \bar{\mathrm{v}}_{\mathrm{c_i}}\}$. The features include a reference to each cellc_i where $\{\mathbf{c}_\mathrm{i}\}_{\mathrm{i=1}}^{\mathrm{k}} \subset \mathrm{C}$, and $\bar{\mathrm{v}}_{\mathrm{c_i}}, \mathrm{v}_{\mathrm{c_i}}$ express the correct and the observed values for c_i, respectively. Additionally, $\mathrm{y_i} \in \{-1, 1\}$ represents the output label as a binary flag of a given cell (i.e., correct or erroneous). Generally, given a database table D and

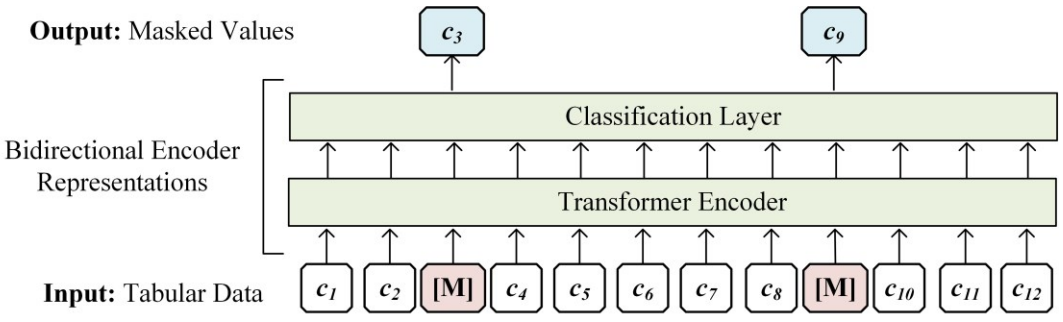


Figure 6.3: Masked Data Model task in TabReformer

a training dataset D_t, the goal of the proposed model is to classify each cell value in C by assigning a label to denote c_i having a correct value {1} or an erroneous one {-1}.

6.3.2. Model Design

The model architecture uses bidirectional encoder representations [19] with Gaussian Error Linear Unit (GELU) activation functions [42]. The model uses the encoder architecture with multi self-attention layers to capture the dependency relationships between the cells and seize the tuple-level representation. The encoder transforms the input data into another structured sequence. The input is internally altered using attention mechanisms [19] and position-aware connected layers. In our implementation, the number of transformer blocks (layers) is denoted as B=6, and the number of self-attention heads is S =12. First, the model applies a self-supervised learning task during the first phase to model the data representation. To train the model, we propose a Masked Data Model pre-training objective in which a fraction of the input cells is masked with a special token. An example of MDM is shown in Figure 6.3. As the figure shows, the input tuple C_l has 12 cells with the cells c_3 and c_9 being replaced with a mask [M]. Then, the model is trained to detect these cells. Finally, for supervised fine-tuning, the model learns the task of erroneous data detection with the help of the labeled dataset D_t. To tackle the problem of imbalanced data, we introduce a data augmentation approach to generate more synthetic examples. In this stage, a generative process applies a set of transformations to the training examples in D_t. These transformations are executed on the correct values of each cell to create more erroneous values.

Although transformers are usually coupled with language modeling [23], [39]–[41], we show that using bidirectional transformer training can gain a deeper understanding of tabular data contexts. The following subsections offer further details for the implementation of TabReformer.

6.3.2.1. Bidirectional Transformers for Structured Data

For unsupervised pre-training, the proposed model operates on a sequence of values $\{l[a_1], l[a_2], .. l[a_N]\}$ representing the cells in a tuple l. Similar to Seq2seq models [43], the input sequence is processed by stacked encoder layers to output the encoded representation. However, to accommodate tabular data, we alter the structure of the first encoder input to process cells with continuous values without modification. Alternatively, cells with categorical value are mapped

using trainable embeddings [44]. Moreover, to accelerate the training phase, a preliminary step of instance normalization [45] is applied to standardize the input embeddings as:

$$E_{l[a_i]}^{norm} = \frac{E_{l[a_i]} - \mu_{in}(E)}{\sqrt{\sigma_{in}^2(E) + \epsilon}} \tag{6.1}$$

where $E_{l[a_i]}^{norm}$ is the normalized output of the input embedding $E_{l[a_i]}$, and $\mu_{in}(E)$ and $\sigma_{in}^2(E)$ are the instance means and variances [46]. The output of the normalization layer is then passed to an attentive transformer to model the dependencies between the attributes. Figure 6.4 shows an illustration of the transformer structure in the proposed framework. As the figure depicts, the input embeddings are passed to the first encoder. The output is then propagated to the following encoder layers as:

$$\mathrm{h_i} = \mathrm{encoder}_{\mathrm{block(h}_{i-1})}, \forall\, \mathrm{i} \in [1, \mathrm{B}] \tag{6.2}$$

Each encoder block consists of a multi-head attention layer followed by a layer of a feed-forward network. The multi-head attention layer applies, within each head, a set of transformations based on scaled dot product attention [19] to its input to capture the tuple related features as:

$$\mathrm{Z_i} = \mathrm{softmax}(\frac{Q_i \times K_i}{\sqrt{d_k}})V_i \;\forall\, \mathrm{i} \in [1, \mathrm{S}] \tag{6.3}$$

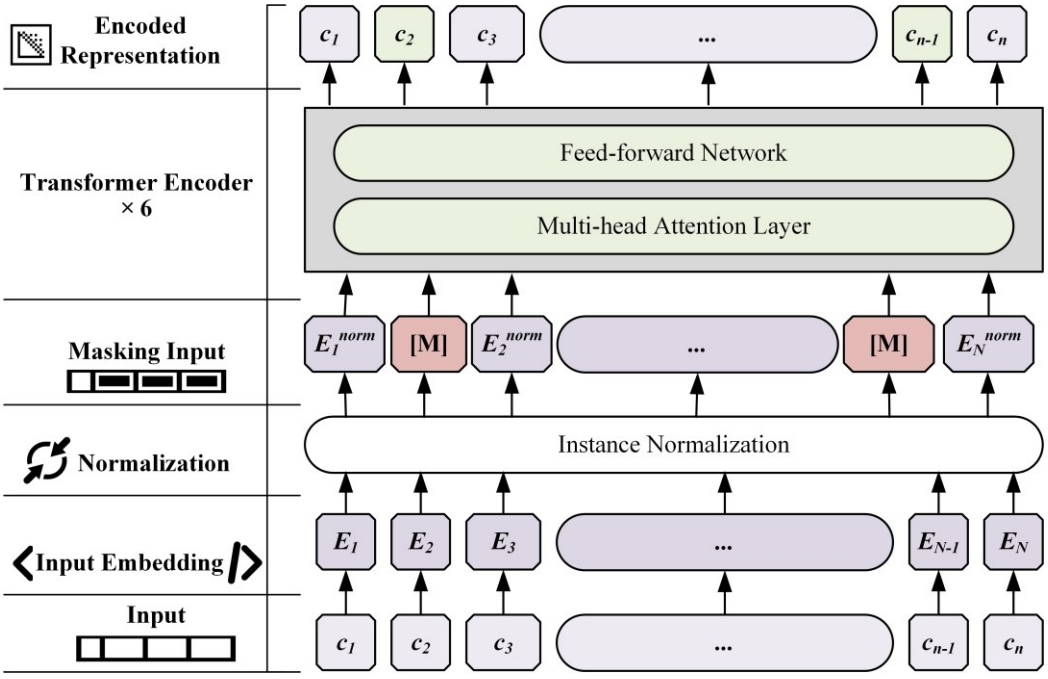


Figure 6.4: Self-supervised learning in TabReformer

where Q_i, , K_i, and V_i are the query, key, and value matrices, which are calculated for each head [19]. These matrices are multiplied together [19] to produce Z. These representations are then fed to a feed-forward neural network. As a self-supervised objective, we present the task of a masked data model. Similar to the Cloze task [23], [47], we mask 15% of the cells in each instance at random where each masked cell is replaced by a special symbol [M]. Then, the network is trained to predict the masked cells. To formalize the objective function, we use the log-likelihood as follows:

$$L_1(C_l;\ \theta) = \frac{1}{|C_l|}\sum_{c \in C_l} \log P(\mathrm{c_i} \mid c_{\neq i}; \theta) \tag{6.4}$$

where $C_l = \{l[\mathrm{a}_1], l[\mathrm{a}_2], .. l[\mathrm{a}_\mathrm{N}]\}$ contains all the cells in l, P is the conditional probability that is modeled using the network with parameters θ, and $c_{\neq i}$ denotes the cells that appear before and after position i. In other words, the model aims to predict the masked token, given the instance inputs appearing before and after that token, and calculate the loss function for the masked data model. Generally, the task of detecting masked cells is analogous to identifying erroneous data. Therefore, the objective allows us to obtain a bidirectional trained model with unsupervised representation. However, the mask tokens used in training may not appear during fine-tuning. Therefore, inspired by best-practice found in the literature [23], [40], we replace 80% of the masked cells with a masked token, 10% are left unchanged, while 10% are swapped with values from other tuples.

However, unsupervised representation learning in language modeling pre-trained systems [23] usually include multiple learning objectives. For example, the implementation of BERT [23] contains a Next Sentence Prediction (NSP) objective to train the model to infer the relationship between two sentences. For NSP, the model is pre-trained with pairs of sentences, and the goal is to predict if a given pair represents two consecutive sentences. However, previous studies [40] stated that removing the next sentence prediction loss can either match or improve the overall performance. Therefore, we decided not to accommodate the next tuple prediction objective while training TabReformer. Furthermore, we found that the next tuple prediction objective does not yield performance gains for erroneous data detection in tabular data (Section 6.5).

6.3.2.2. Parameter Fine-tuning to Erroneous Data Detection

After training the model, the second phase fine-tunes the system parameters to the target task of erroneous data detection. During this phase, the trained model accepts a labeled dataset to tune all the parameters. Therefore, $\{\mathbf{x}_i\}_{i=1}^{K}$ in D_t are supplied as inputs, and the ground-truth labels $\{\mathbf{y}_i\}_{i=1}^{K}$ are entered into an output layer for erroneous cell classification. As a result, a classifier is trained with minimal changes to the pre-trained model. To accomplish this task, the activation function of the final transformer block h_B goes through a linear adder layer to predict y as:

$$P(y|\boldsymbol{x_i}) = softmax(h_B W_y) \tag{6.5}$$

Where h_B is the output of the last encoder block, and W_y are the parameters for the linear output layer. Overall, the model aims to maximize the following objective function:

$$L_2(\boldsymbol{x}) = \sum_{(x,y)\in D_t} \log P(y \mid \boldsymbol{x}) \tag{6.6}$$

Moreover, to minimize the computational complexity, most of the model hyperparameters are kept the same except for W_y, the learning rate, and the number of training epochs. As mentioned before, the labeled dataset D_t is utilized in this step for supervised fine-tuning. Given the fact that errors in datasets are often limited compared to the number of correct cells, the collected data is usually highly imbalanced. Since learning models tend to treat the minority class as noise and ignore it, this can affect the classification performance. To mitigate the imbalance risk, we execute a preliminary stage of data augmentation before fine-tuning the model. During such a step, more synthetic labeled points are generated with minimal manual effort from end-users. A detailed description of this stage is explained next.

6.3.2.3. Data Augmentation for Tabular Data

Since training neural networks with small unbalanced datasets can lead to overfitting, we employ data augmentation to add more erroneous data points to D_t during fine-tuning. For this purpose, we aim to specify a set of transformation functions T, which, when applied to correct values, can generate erroneous ones. Moreover, to decide on which transformations should be used to which cell value, we need to derive a strategy of error generation S_{EG} [31]. Once both T and S_{EG} are specified, the model can start learning additional training examples from D_t.

The stage begins by applying pattern matching to determine T. The approach iteratively examines each of the labeled examples $\{c_i, \mathrm{v_{c_i}}, \overline{\mathrm{v}}_{\mathrm{c_i}}\, \mathrm{y_i}\}$ and extracts all possible transformations $f \in \mathrm{T}$ which could be applied to $\overline{\mathrm{v}}_{\mathrm{c_i}}$ to produce the observed value $\mathrm{v_{c_i}}$, so that $\mathrm{v_{c_i}} = f(\overline{\mathrm{v}}_{\mathrm{c_i}})$. Motivated by previous techniques for data augmentation [48], we consider that each $f \in \mathrm{T}$ employs one of the following operations:

- Random replacement: the process randomly selects a character in $\overline{\mathrm{v}}_{\mathrm{c_i}}$ and replaces it with another random character from the alphabet.
- Random insertion: the operation selects a position in $\overline{\mathrm{v}}_{\mathrm{c_i}}$ at random and inserts an additional character, chosen from the alphabet, in that position.
- Random deletion: the process picks a random character in $\overline{\mathrm{v}}_{\mathrm{c_i}}$ and removes it.
- Random swapping: the operation swaps two characters chosen at random in $\overline{\mathrm{v}}_{\mathrm{c_i}}$.

Given these operations, the process extracts a set of transformations from each erroneous example $\mathrm{D_t}$ (records with $\mathrm{y_i} = -1$). A detailed explanation of the extraction process is presented in Algorithm 1. As the algorithm shows, the learning process applies the Gestalt Pattern Matching algorithm [49] to find the similarity between $\mathrm{v_{c_i}}$ and $\overline{\mathrm{v}}_{\mathrm{c_i}}$ in each erroneous example in $\mathrm{D_t}$ (lines 4-6). The model returns the longest common substring found in $\mathrm{v_{c_i}}$ and $\overline{\mathrm{v}}_{\mathrm{c_i}}$. Then, it recursively returns the matching characters in the non-matching regions on both sides of that substring (lines 7-13). The model then extracts the transformations by examining the overlap between the matching substrings and fits them with the set of available operations. Finally, the algorithm merges all the sets derived from each example to produce a final multiset F.

Algorithm 6.1: The Process of Extracting Transformations (extract_f)

Input: A set $\mathrm{D_{err}}$ of training examples $\{\mathrm{c}_i, \mathrm{v_{c_i}}, \overline{\mathrm{v}}_{\mathrm{c_i}}, y_i\}$ from $\mathrm{D_t}$ where $\mathrm{y_i} = -1$

Output: A multiset F of transformations functions extracted from the examples in $\mathrm{D_{err}}$

1: initialize $\mathrm{F} = \emptyset$

2: **for each** point x **in** $\mathrm{D_{err}}$ **do**:

3: initialize $\tau = \{(f(\overline{v_x}) = v_x)\}$

4: initialize *str* as the longest common substring between v_x and $\overline{v_x}$

5: obtain $\overline{str_l}$ and $\overline{str_r}$ substrings as the left and the right substring surrounding $\overline{v_x} - str$

6: obtain str_l and str_r substrings as the left and the right substring surrounding $v_x - str$

7: if similarity_score($\overline{str_l}$, str_l) + similarity_score($\overline{str_r}$, str_r) > similarity_score($\overline{str_l}$, str_r) + similarity-score($\overline{str_r}$, str_l):

8: $\tau = \tau \cup \{(f(\overline{str_l}) = str_l), \{(f(\overline{str_r}) = str_r)\}$

9: $\tau = \tau \cup \{(extract\_f(\{str_l, \overline{str_l}\}, (extract\_f(\{str_r, \overline{str_r}\})$

10: else:

11: $\tau = \tau \cup \{(f(\overline{str_l}) = str_r), \{(f(\overline{str_r}) = str_l)\}$

12: $\tau = \tau \cup \{(extract\_f(\{str_r, \overline{str_l}\}, (extract\_f(\{str_l, \overline{str_r}\})$

13: end

14: F = F $\cup$ τ

15: **end**

16: return F

Moreover, the process still needs to learn the strategy of error generation S_{EG} which corresponds to the conditional probability distribution P(T | $\overline{v}_{c_i}$) for a given correct value $\overline{v}_{c_i}$ in D_t. Therefore, given the extracted multiset F from Algorithm 1 (line 14), the transformations set T is first constructed by removing the duplicated records in F. Then, the model learns S_{EG} by first calculating the empirical distribution of each function in F. Since F is expected to have duplicated transformation functions applied to different data points, the empirical distribution of each transformation function *f* in T can be formally denoted as:

$$P(f) = \frac{\sum_{x \in F} \mathbf{1}\{x = f\}}{|F|} \quad (6.7)$$

Where $|F|$ is the cardinality of F, and $\sum_{x \in F} \mathbf{1}\{x = f\}$ returns the number of times a function *f* appears in F. The process is further explained in Algorithm 2.

Algorithm 6.2: Learning the Empirical Distribution of Transformation Functions

---

Input: The multiset F of transformation functions

Output: A final set of transformations functions T, empirical distribution for each f in T as $\{P(f)\}_{f \in T}$

1: obtain $|F|$ as the number of elements in F

2: set T as all the unique transformation functions in F

3: **for each** f **in** T

4: compute $\sum_{x \in F} \mathbf{1}\{x = f\}$

5: use Equation (6.7) to calculate the empirical distribution $P(f)$

6: **end**

7: return T, $\{P(f)\}_{f \in T}$

Then, to derive the conditional probability distribution $P(T \mid \overline{v_{c_i}})$ given a correct cell value $\overline{v_{c_i}}$, the model finds all transformation functions in T as $f(\overline{str}) = str$, such that *str* can be seen as a subset of $\overline{v_{c_i}}$. Next, we consider the maximum and minimum of the empirical probabilities of these functions to normalize the empirical probability $P(f)_{f \in T}$. Furthermore, the conditional probability can be formally denoted as:

$$P(f \mid \overline{v_{c_i}}) = \frac{P(f) - min(P\left(f_{\overline{v_{c_i}}}\right)_{f \in T})}{norm} \tag{6.8}$$

structureWhere $f_{\overline{v_{c_i}}}$ is any transformation function over a substring of $\overline{v_{c_i}}$, and *norm* is calculated as:

$$norm = max(P\left(f_{\overline{v_{c_i}}}\right)_{f \in T}) - min(P\left(f_{\overline{v_{c_i}}}\right)_{f \in T}) \tag{6.9}$$

Finally, these normalized empirical probabilities can be used by S_{EG} to select which f should be applied to a given value $\overline{v_{c_i}}$.

Consequently, the model randomly selects an instance from the correct training examples in D_t (records with $y_i = 1$). Then, for each sampled data point, the conditional distribution $P(T \mid \overline{v_{c_i}})$ along with the learned transformations T are utilized to select appropriate transformation functions, and to add more training examples to D_t. The newly noisy value $\widetilde{v_{c_i}} = f(\overline{v_{c_i}})$ is then added to D_t as $D_t = D_T \cup \{c_i, \widetilde{v_{c_i}}, \overline{v_{c_i}}\}$. The algorithm takes a hyper-parameter γ which specifies the target ratio between correct and erroneous examples in the final training data D_t. During the experiments, the value of γ is determined with cross-validation using a held-out set taken from D_t.

6.4. Experimental Evaluation

The section presents empirical results obtained when comparing TabReformer against state-of-the-art alternatives on a variety of real-world datasets. The experimental evaluation seeks to validate the following claims:

- **Training a bidirectional transformer on structured data and fine-tuning it to the task of erroneous data detection can yield high-quality classification models.** We compare TabReformer to other error detection techniques that rely on machine learning [3], [16], [17]. The experimental results show that the proposed method outperforms other deep learning methods [3] by 45.86% on average (recall). Also, the experiments illustrate that the final trained model improves precision by 16.90% and the recall by 29.28% on average when compared to other machine learning techniques [16], [17].
- **Data augmentation represents an optimal approach for obtaining ample training data while minimizing the required human effort.** We compare the data augmentation module in TabReformer to other paradigms for collecting training data such as supervised learning and active learning [16]. Along with performance metrics, the experiments consider user effort to evaluate the model. The experimental results show that the proposed method can enhance the classification performance by 28.69% on average (F1 measure), while reducing the manual labeling effort by up to 48.86%.

Moreover, we perform a micro-benchmark to evaluate the individual design choices of the TabReformer, such as the effectiveness of data augmentation and adding other training objectives. The section is divided into four subsections. In the first subsection, we discuss the datasets and the baseline techniques, along with the evaluation setup. Next, we report the results of comparing the

Table 6.1: Datasets used in the evaluation

Dataset	Size	Dimensionality	K	Errors (# of cells)	Errors %
Adult	48,842	14	2,100	72384	12.30
Restaurants	28,788	16	1,439	19168	14.40
Flights	13,884	10	819	7297	13.10
Movies	7,390	17	318	14193	13.60
Hospital	4,561	19	283	2480	13.50
Beers	2,410	11	147	3152	11.80

proposed method to other error detection methods. Then, to validate the data augmentation claim, we compare data augmentation to traditional active learning and experiment with different values for the labeling cost. Finally, in the fourth subsection, we evaluate the individual components of the proposed system by experimenting with two variations of our model in which we investigate different design choices and learning paradigms.

6.4.1. Evaluation Setup

Datasets: The experiments utilize six datasets that explore a wide range of domains and vary in size, dimensionality, error types, and distributions. The summary statistics of these datasets are provided in Table 6.1. The table shows, for each dataset, the number of tuples (Size), the number of columns (Dimensionality), the initial size of training data D_t (K), the number of erroneous cells (Errors (# of cells), and the corresponding percentage of incorrect cells divided by the total number of cells in each dataset (Errors %). Although existing research [15], [16] has experimented with low ratios of injected errors (less than 2.5%), recent surveys [50], [51] show that the real-world datasets contain higher percentages of inaccurate entries and data errors (more than 10%) [50]. Therefore, the experiments consider the ratios reported in these surveys to set up more elevated rates of injected errors.

- The first dataset used in the experiments is the **Adult** dataset, which is a benchmark dataset [52] that is collected by Barry Becker from the 1994 Census database. The dataset contains various attributes for individuals such as their education level, age, gender, along with their annual income. Errors are introduced using BART [53], which include 39% typographical errors, and 61% value swaps across attributes.
- The experiments also consider the **Flights** dataset [22], which comprises departure and arrival information on domestic flights in the USA. The data is collected by the U.S. Bureau of Transportation Statistics. Errors in the dataset are manually injected to have 27% typos, 14% formatting errors, and 59% values violating data constraints.
- The third dataset in the experiments is the **Restaurant** dataset, which contains information about restaurants in the United States. Similarly, BART [53] is used in this dataset to inject errors with 63% values swaps, 13% duplicated values, and 24% typos.

- Another dataset is the **Movies** dataset that includes information about movies crawled from IMDB. To introduce errors, we manually injected 21% typographical errors, 17% duplicated error, and 62% values swaps.
- The **Beers** dataset is a benchmark database that is used in the literature to evaluate error detection models [16]. It encompasses information about different beer styles and brands. The data is crawled from CraftCans.com in 2017 and contains 12% missing values, 34% value swaps across tuples, and 54% typographical errors.
- Finally, the experiments include the **Hospital** dataset, which is a benchmark dataset used to evaluate several error detection tools [2], [15]. The dataset only contains typographical errors introduced by BART [53].

Competing methods. We compare TabReformer against the following baseline techniques:

- **HoloClean** [3]**:** is a state-of-the-art holistic data repairing technique that is driven by probabilistic inference. The current implementation of HoloClan is compatible with various types of error detection methods, which include denial constraints violation [11], outlier detections [7], and missing values detection. The experiments only evaluate the detection capabilities of HoloClean since data repairing is beyond the scope of this chapter.
- **ED2** [16]**:** is a two-stage example-driven error detection method. The method first applies a classification strategy to choose the cells that need to be tagged by the end-user as correct or erroneous. After collecting labeled data from the user, the method utilizes a wide range of features to represent the data and detect incorrect cells. When applied to datasets with relatively small error ratios, the model reports superior performance over the state-of-the-art solutions [9], [15], while reducing the effort of manual labeling.
- **NADEEF** [9]**:** is another error detection and data cleansing framework which allows users to define data quality rules that specify data problems using a programming interface. NADEEF compiles all these rules and examines the data to select violating cells. Furthermore, to repair corrupted data, NADEEF applies a mixture of data correcting algorithms and inputs provided by domain experts to achieve good repairing results.
- **ActiveClean** [17]**:** is an iterative data cleaning tool that applies statistical model training to detect erroneous data cells recursively. The approach employs a selection of convex loss

models to clean dirty data and improve classification performance iteratively. ActiveClean applies a sampling algorithm that selects data batches that need to be cleaned by end-users. Then it feeds this clean data into the model to retrain it and recommend other batches to the user.

Moreover, we also experiment with three variants of the TabReformer:

- **Reformer$_{\text{Supervised}}$:** In this variation, the module of data augmentation is disabled. Instead, the model is fine-tuned using the initial data points provided in D_t.
- **Reformer$_{\text{AL}}$:** Instead of the data augmentation module, we apply traditional active learning [54] to obtain additional training examples. First, the model is fine-tuned with D_t. Then, we employ uncertainty sampling for some *r* iterations. During every iteration, the user is queried to label a batch of 50 examples. Then, the model is retrained and evaluated.

Table 6.2: Evaluation metrics of different methods for error detection

Dataset (Size of D_t)	Evaluation Metric	TabReformer	HoloClean	ED2	NADEEF	ActiveClean
Adult (4.30%)	P	**0.97**	0.82	0.91	0.92	0.96
	R	**0.95**	0.59	0.83	0.93	0.61
	F_1	**0.96**	0.69	0.87	0.92	0.75
Restaurant (5.00%)	P	**0.92**	0.73	0.79	0.81	0.89
	R	0.87	0.67	**0.89**	0.77	0.58
	F_1	**0.89**	0.70	0.84	0.79	0.70
Flights (5.90%)	P	**0.93**	0.87	0.80	0.78	0.63
	R	**0.96**	0.61	0.89	0.67	0.65
	F_1	**0.94**	0.72	0.84	0.72	0.64
Movies (4.30%)	P	0.87	0.71	0.87	**0.93**	0.78
	R	**0.84**	0.55	0.65	0.49	0.62
	F_1	**0.85**	0.62	0.74	0.64	0.69
Hospital (6.20%)	P	**0.92**	0.92	0.81	0.91	0.89
	R	**0.91**	0.61	0.75	0.73	0.67
	F_1	**0.91**	0.73	0.78	0.81	0.76
Beers (6.10%)	P	**0.97**	0.81	0.91	0.93	0.55
	R	**0.90**	0.72	0.83	0.82	0.61
	F_1	**0.93**	0.76	0.87	0.87	0.58

- **Reformer$_{NTP}$:** In this version, the training phase is repeated with the next tuple prediction (NTP) objective enabled. Similar to the state-of-the-art models of natural language processing [23], [40], we train the model with pairs of tuples as input to predict if the second tuple follows the first one in the input table D. The training data has 50% of the input as consecutive tuples (with label y = 1), while the rest are separate tuples chosen randomly from D (with label y=-1). The training loss, in this case, is the sum of the mean masked data model likelihood and the mean next tuple prediction likelihood. In the literature [23], [41], the next sentence prediction helps the model to understand sentence-level representation. Therefore, we add this version of the model to investigate if this will improve the model's capability to capture table-level contexts.

Experimental Setup. To measure the effectiveness of error detection, we report Precision(P), Recall (R), and F1 measure. All the datasets used in the experiments have clean versions that are used as the ground truth. During the training phase, the existing ground truth is split to form the labeled training data D_t, an unlabeled pool for the active learning experiments, a test set for evaluation, and a held-out set for hyper-parameters tuning. To optimize TabReformer, we use ADAM [55] with a learning rate of 0.02. In the experiments, we used ED2 with the min certainty [16] as the column selection strategy and a learning batch size of 50 cells. The labeling cost consumed by ED2 is set to 4% of the total size of each dataset. The limit is determined since it is reported as the optimal cost in their experimental evaluation [16]. As for ActiveClean, the model is initially trained using D_t. Then, in each and every iteration, ActiveClean recommends a batch of 50 tuples to be cleaned by the user. After that, the approach updates (retrains) the current model using the obtained clean data and selects the next batch. This iterative clean-retrain process is repeated until an optimal clean model is realized [17]. In the experiments, we used ActiveClean with the SVM model [17] with the Adult and Flights datasets, while linear regression is applied to the rest. According to their experiments, the labeling budget is usually set to be around 2% to 10%. Therefore, our experimental evaluation considers a maximum number of iteration as $i/50$ where $i = 6\%$ of the size of each dataset. Moreover, Section 6.4.3 represents additional experiments to assess the sensitivity of the cost variable. Additionally, to evaluate the methods that rely on data sampling such as ED2 and Reformer$_{AL}$, we repeated the experiments ten times and reported the arithmetic mean.

6.4.2. End-to-end Performance

In this section, we compare the classification performance of TabReformer to detect data errors against the competing approaches in the six datasets. The experimental results achieved by different methods are presented in Table 6.2. The table shows, for each method, the precision, recall, and F1 measure; the table also represents the percentage of training data D_t of the total dataset. These percentages refer to the initial size of the training data before applying the data augmentation module (Section 6.3.2.3).

As Table 6.2 illustrates, TabReformer consistently achieves better F1 measure than other approaches in all of the datasets. For example, in the Beers and Adult datasets, TabReformer could enhance the performance (F1 measure) by 61.41% and 39.88%, respectively. Also, in most of the datasets, TabReformer attains the highest precision and recall values, especially in the datasets that contain multiple types of errors with different distributions. For instance, the results show that in the Beers dataset, the proposed approach outperforms Holoclean in precision and recall values by 19.75% and 25.13%, respectively. Since the performance of HoloClean highly depends on the quality of its error detection techniques, it shows imperfect results in those datasets that include several error types. On average, the results depict that TabReformer could improve the precision and recall values by 13.89% and 32.95%, respectively, when compared to the other techniques.

Alternatively, the results show that in the Movies dataset, NADEEF achieves better precision than TabReformer. Nevertheless, a closer look at the results shows that although NADEEF can detect cells violating predefined quality rules, it fails to report most of the value swaps in this dataset, which results in a significantly low recall (0.49). Additionally, in most of the datasets, results illustrate that data representation plays an essential role in detecting various types of errors. For example, in the Flights datasets, the majority of the errors came from cells violating integrity constraints and functional dependencies (Figure 6.1). Hence, modeling the data can substantially enhance classification performance in these tasks.

Consequently, approaches that depend on learning the data representation, such as TabReformer and ED2, managed to achieve higher recall in these situations. For instance, TabReformer attains better recall in the Flight dataset by a maximum improvement of 57.38% when compared to HoloClean. Similarly, ED2 outperforms HoloClean in the same dataset by 45.90% in recall values.

However, when dealing with a higher volume of errors, such as in the Movies dataset, ED2 reports poor recall values due to the diversity of error distributions and the limited labeling effort.

The results demonstrate that HoloClean shows an adequate performance in datasets with outliers, missing values, and constraints violations. For example, ignoring TabReformer, HoloClean enhances the precision values in the Hospital dataset by 6.02% on average when compared to the other tools. However, in the Movies and the Flights datasets, HoloClean reports poor recall values (0.55 and 0.61, respectively). Most of the reported errors are related to integrity constraints defined for these datasets. Nevertheless, since most of the errors injected in these datasets require exploiting inter-column relationships and functional dependencies, the performance of HoloClean was bounded by detecting cells violating denial constraints.

As for ActiveClean, the evaluation shows that it fails to capture the necessary tuple-level characterization to classify erroneous cells. For example, in the Restaurant dataset, ActiveClean reports the worst F1 measure due to is significantly lower recall (0.58) as it failed to detect any of the typographical errors. Moreover, in the Flights dataset, the tool reports a large number of false positives, which results in the smallest value of precision (0.63) among the competing techniques. Accordingly, we postulate that these results agree with our assumption that modeling data characteristics improve the classification of erroneous cells. Overall, the results empirically posit that, since TabReformer uses a bidirectional transformer to model the data, it manages to output more accurate results and detect a broader range of error types with different error distributions.

6.4.3. Data Augmentation versus Active Learning

To estimate the effectiveness of data augmentation, we compare it to traditional active learning and study the impact of the labeling cost to the performance of the competing methods. First, we validate the claim that data augmentation can optimize labeling effort while achieving a satisfactory performance for erroneous values classification. Therefore, we train a new model $\text{Reformer}_{\text{AL}}$ in which we disable the data augmentation step and replace it with uncertainty sampling [54]. In this version, the model is first trained via unsupervised learning over D (Section 6.3.2.1). Then, during fine-tuning, the model applies active learning with uncertainty sampling for several iterations. Uncertainty sampling ranks the output of the last layer to selects the point about which the network is most uncertain. In each iteration, the model acquires labels for a batch of 50

Table 6.3: Performance of Reformer$_{Supervised}$ and Reformer$_{NTP}$ with increasing sizes of training data

Dataset	(Size of D_t)%	TabReformer			Reformer$_{Supervised}$			Reformer$_{NTP}$		
		P	R	F1	P	R	F1	P	R	F1
Adult	5%	**0.97**	**0.95**	**0.96**	0.71	0.64	0.67	0.91	0.89	0.90
	10%	**0.97**	**0.98**	**0.97**	0.72	0.67	0.69	0.93	0.90	0.91
	15%	**0.97**	**0.97**	**0.97**	0.75	0.70	0.72	0.95	0.87	0.91
Restaurants	5%	**0.92**	**0.87**	**0.89**	0.68	0.65	0.66	0.83	0.87	0.85
	10%	**0.94**	**0.91**	**0.92**	0.75	0.69	0.72	0.87	0.90	0.88
	15%	**0.96**	**0.90**	**0.93**	0.79	0.71	0.75	0.89	0.90	0.89
Flights	5%	**0.93**	**0.95**	**0.94**	0.57	0.23	0.33	0.92	**0.95**	0.93
	10%	0.90	**0.97**	**0.93**	0.66	0.56	0.61	**0.93**	0.92	0.92
	15%	**0.96**	**0.92**	**0.94**	0.70	0.69	0.69	0.94	**0.92**	0.93
Movies	5%	**0.86**	**0.87**	**0.86**	0.68	0.64	0.66	0.90	0.81	0.85
	10%	0.83	**0.89**	**0.86**	0.70	0.65	0.67	**0.91**	0.83	**0.87**
	15%	0.92	**0.91**	**0.91**	0.76	0.70	0.73	**0.93**	0.88	0.90
Hospital	5%	**0.93**	**0.91**	**0.92**	0.58	0.55	0.56	0.87	0.83	0.85
	10%	**0.92**	**0.96**	**0.94**	0.67	0.59	0.63	0.91	0.87	0.89
	15%	**0.95**	**0.95**	**0.95**	0.77	0.68	0.72	0.90	0.93	0.91
Beers	5%	**0.96**	**0.90**	**0.93**	0.51	0.41	0.45	0.89	0.91	0.90
	10%	**0.95**	**0.94**	**0.94**	0.55	0.72	0.62	0.90	0.89	0.89
	15%	**0.98**	**0.96**	**0.97**	0.62	0.76	0.68	0.93	0.94	0.93

examples of erroneous cells and adds this batch to D_t. After obtaining these labels, the fine-tuning phase is repeated, and the model is evaluated using a test set. Finally, we compare the new model Reformer$_{AL}$ against the original implementation of TabReformer to validate the claim of data augmentation.

Moreover, to study the effect of the parameter of labeling cost, the experiments with ED2 and ActiveClean are repeated with different numbers of iterations $r \in \{5,10,20,50\}$. The initial labeled data D_t is set as in Table 6.1, and we report the F1 measure of each of the competing approaches with additional iterations. Also, since TabReformer does not utilize any labeled data from the user, the number of labeled examples obtained during each setup of r is computed and added to D_t. Then, TabReformer utilizes this updated version of D_t during the data augmentation module to generate more synthetic data points.

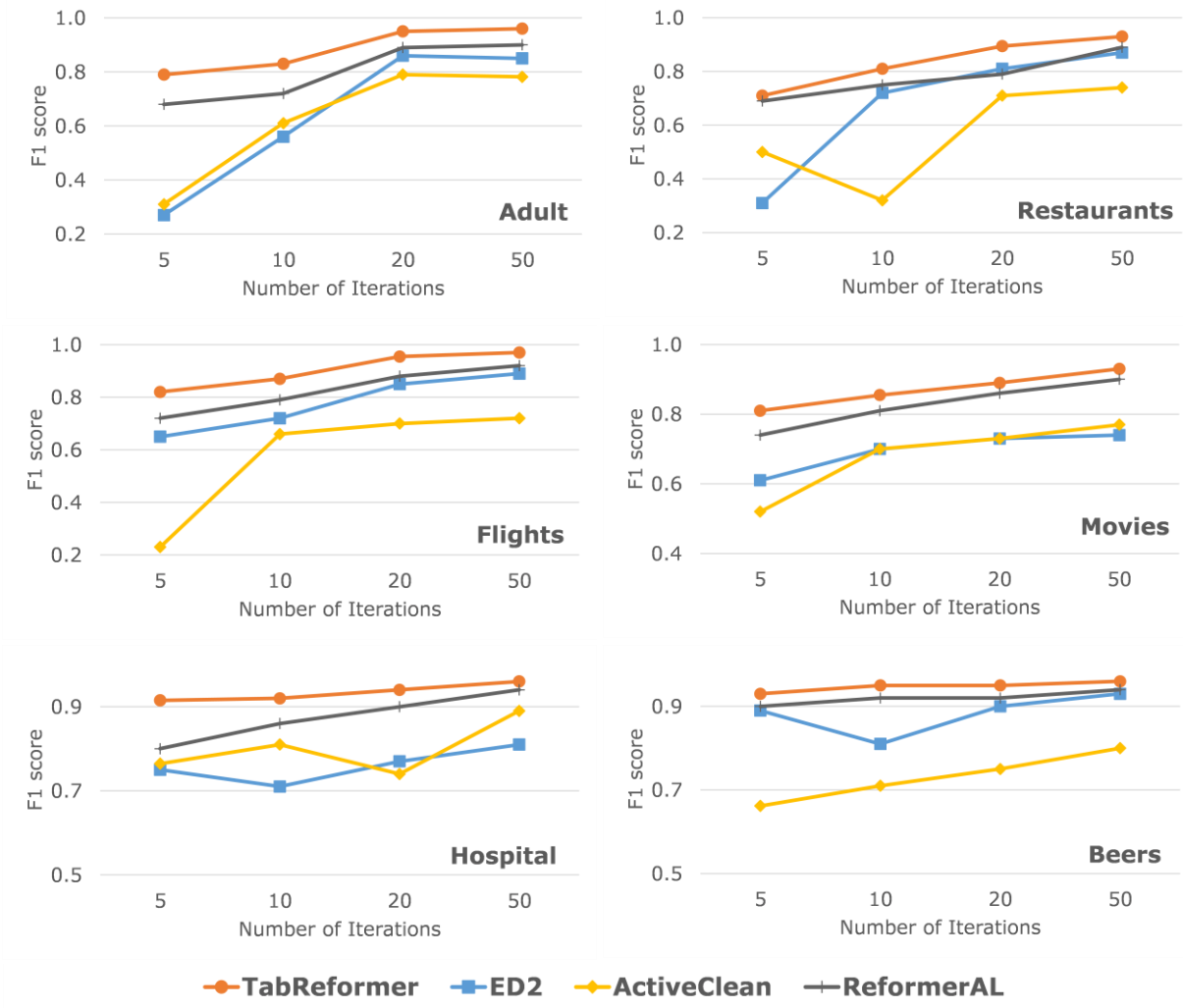


Figure 6.5: F1-score of detection methods with increasing labeling efforts

For each dataset, the F1 measure of each of the four models (TabReformer, ED2, ActiveClean, and Reformer_{AL}), is plotted against different r values in Figure 6.5. As the figure shows, TabReformer attains better F1 scores in all the datasets. Extending D_t with more examples allows TabReformer to achieve a higher F1 measure since the data augmentation algorithm can learn more training examples and represent different errors. For instance, in the Adult dataset, TabReformer initially attains an F1 measure value of 0.83 with a small number of iterations ($r = 10$). Subsequently, escalating D_t with more iterations, ($r = 20$ and $r = 50$), improves the performance of TabReformer by 10.84% and 4.34%, respectively.

Additionally, when comparing TabReformer with Reformer_{AL}, the results illustrate that data augmentation outputs better models than active learning. Although Reformer_{AL} outperforms ED2

and ActiveClean in all the datasets, the original implementation of TabReformer maintains a better classification performance with much less human effort. For instance, in large datasets such as the Adult and the Restaurant datasets, the data augmentation module improves the performance of the error detection by 15.28% and 8.01%, respectively, when compared to $\text{Reformer}_{\text{AL}}$ with ten iterations of active learning ($r = 10$).

Also, the results suggest that, in many situations, ActiveClean shows a faster converge than ED2. For example, in the Flights dataset, ActiveClean consumed fewer iterations (with $r = 20$) to reach global converge, while ED2 requires more than 40 iterations. Nevertheless, ED2 consistently outperforms ActiveClean in most datasets except for the Movies and the Hospital datasets in which ActiveClean achieves slightly better enhancements over ED2 (3.90% and 8.99% respectively with $r = 50$). In general, the results demonstrate that, since TabReformer applies data augmentation while modeling the data representation, the approach can realize high-quality data models with minimum manual efforts.

6.4.4. Micro-Benchmarking

To evaluate the effect of different design decisions implemented in TabReformer, we conduct an additional set of experiments where we compare different variations of the system. First, to assess the training objective, we repeat the training stage with an additional objective function to predict the next tuple. The new model, $\text{Reformer}_{\text{NTP}}$, combines the (arithmetic) mean of the two objective functions and uses it as the training loss. Second, we disable the data augmentation module, and the model is fine-tuned using the initial version of D_t. To assist with the problem of data imbalance, the new model $\text{Reformer}_{\text{Supervised}}$ resamples the points in D_t to make sure that both classes are presented corresponding to the hyper-parameter γ [56]. We repeat the experiments in Section 6.4.2 to compare between the original implementation TabReformer, and the two variations: $\text{Reformer}_{\text{NTP}}$ and $\text{Reformer}_{\text{Supervised}}$ with differing sizes of training data D_t. The experimental results are summarized in Table 6.3. The table shows for each dataset, the values of Precision, Recall, and F1 measure achieved by each model while increasing the size of D_t.

The table shows that, even with different setups of D_t, using data augmentation consistently results in higher-quality models. TabReformer outperforms the other two variations in all the datasets. With large datasets such as the Adult and Restaurants datasets, TabReformer manages to enhance

the F1 measure by 41.12% and 34.55%, respectively, when compared to $\text{Reformer}_{\text{Supervised}}$ with the small size of D_t (K=5%). Since $\text{Reformer}_{\text{Supervised}}$ relies on the examples provided in D_t, the model performance suffers from the effect of imbalanced data. Although resampling is applied to mitigate this risk, the error heterogeneity magnifies the impact of the imbalance problem. Thus, resampling could not represent different error types in the training data, which results in poor performance of the supervised learning model.

Furthermore, in datasets with different error distributions, $\text{Reformer}_{\text{Supervised}}$ produces unsatisfactory results. For instance, in the Flights dataset, the supervised version records a value of 0.33 for the F1 measure, since it only reports 23.81% of the errors injected in the Flights dataset (Recall). Moreover, increasing the size of the training data does not seem to help with the imbalance problem. With bigger training data, $\text{Reformer}_{\text{Supervised}}$ is also outperformed by the other models. For example, when compared to $\text{Reformer}_{\text{Supervised}}$, the original model (TabReformer) could enhance the detection quality by 31.54% (F1 measure) in the Hospital dataset, when training the models with 15% of the dataset. Likewise, $\text{Reformer}_{\text{NTP}}$ reports a 26.66% enhancement in the same dataset when compared to the supervised version. Overall, the empirical results confirm that data augmentation can form a reliable solution for alleviating different levels of imbalance and varying ratios of errors.

Additionally, the table shows the results of comparing the original model with $\text{Reformer}_{\text{NTP}}$ and depicts that, in most cases, training the model with NTP loss does not yields any performance improvements. Instead, removing the NTP objective can slightly improve the overall performance, especially with large datasets. For example, the original implementation of TabReformer improves the classification performance in the Adult and the Restaurant datasets by 5.76% and 4.53% on average when compared to $\text{Reformer}_{\text{NTP}}$ (F1 measure), respectively. Alternatively, adding the NTP loss results in approximately the same performance for many cases, such as the Flights and the Movies datasets (e.g., The only improvement $\text{Reformer}_{\text{NTP}}$ achieved is recorded within the Movies dataset with only 1.07% enhancement over the original implementation of TabReformer with D=10%). Generally, although training with the next sentence prediction can enhance modeling unstructured text [23], the situation is different for tabular data. We find that dataset-related representation does not depend on the relationships between the tuples. Instead, deep bidirectional representation of data features can expand the model performance for error classification.

6.5. Related Work

Given the fact that data-oriented approaches such as analytic systems are becoming critical for innovation in the enterprise, prior research explores a diverse set of techniques to detect and repair data quality issues. Also, since the focus of this research expands to different deep learning techniques, including data augmentation and unsupervised training, we survey existing effort regarding these areas with tabular data.

Learning models for erroneous data detection. Many studies [10], [57], [58] utilize machine learning techniques for error detection and data repairing tasks. As for error detection, recent research [57], [58] has applied machine learning for outlier detection. For example, Adeli *et al.* [58] propose a semi-supervised classification model to discriminate sample outliers and data noise. The model [58] estimates the noisy model using linear discriminant analysis and a labeled training dataset. Alternatively, Koumarelas *et al.* [10] have applied supervised learning to train a learning model for automatic duplicate detection. However, most of these efforts [10], [57], [58] only focus on specific error categories. Furthermore, most of these approaches are only applicable to certain domains, such as computer vision [57] and medical imaging [58].

Additionally, other research [59], [60] has applied different learning paradigms to extract functional dependencies [60] and discover denial constraints [59]. One example is the approach proposed by Eduardo and Sutton [59] to produce a probabilistic model that can induce functional dependencies. The model applies structural expectation-maximization to discover data rules and detect violating data. Unlike TabReformer, these techniques [59], [60] try to assist data analysts by formulating the integrity constrains from the data. Nevertheless, they do not provide fully automated error detection or data repairing systems.

Aiming to provide a more holistic detection system, HoloDetect [15] uses few-shot learning to build a neural network for erroneous data detection. The system integrates weak supervision with supervised learning to leverage noisy signals from data models and train machine learning models. Another example is SCODED [61], which leverages approximate statistical constraints from the data. After detecting the violating data, the model applies data partitioning to identify the minimum number of records that, if removed, could resolve detected violations. However, unlike the proposed model, these systems rely on many assumptions regarding the underlying data characteristics. For example, HoloDetect [15] assumes that the data can be described using the

concept of the probabilistic unclean database Model [62]. Likewise, SCODED [61] only considers multi-column dependencies, and hence can be limited to single-column errors.

Data augmentation. To overcome the scarcity of training data, prior research [33], [63]–[65] has applied data augmentation to prevent overfitting. Popular techniques [63] apply affine transformations such as translation, relation, cropping, etc., to infer synthetic labeled images from actual images. Furthermore, recent research [33], [64] has utilized generative models to interpolate augmented examples from training data. For example, Liu *et al.* [64] propose a technique in which a generative adversarial network is first trained using the original data. Then, the learning is transferred to generate additional images. Another study [65] integrates data augmentation with semi-supervised learning and learns a model to treat augmented data as noise. As a result, the final model [65] is trained to become robust to input noise. However, these approaches are only applied to specific domains such as computer vision [64], [65], unstructured data [65], and audio separation [33]. Furthermore, none of these techniques have applied data augmentation to structured databases.

Self-supervised learning. Unsupervised representation learning aims to discover data characteristics without labeled examples. For example, to learn image representation, self-supervised learning can help the model to infer different relations between images during training. An existing approach [66] applies autoencoder networks to encode input features into representations that preserve the structure of the original images. Then, the model is trained to predict the original label from the encoded features.

Furthermore, the same idea is employed in different domains, such as language modeling [23], [40], and audio classification [67]. However, unlike TabReformer, none of these approaches have considered tabular data or relational databases. Based on our literature survey, only one recent technique [68] has explored applying self-supervised learning with tabular data. Nevertheless, the technique mainly aims to learn decision-tree-like mappings of the data. As far as we know, no previous research has investigated the capability of attentive neural networks to classify erroneous data

6.6. Conclusions

The chapter presents TabReformer, a learning framework for detecting erroneous values in tabular data. The framework trains a bidirectional model to learn the data representation. To effectively achieve a deeper understanding of the data context, the model implements unsupervised representation learning using the Masked Data Model objective with GELU activation functions. Moreover, to fine-tune the model, TabReformer introduces a phase of data augmentation to generate synthetic labeled examples while optimizing manual labeling effort. In the data augmentation process, both the transformation functions and the augmentation strategy are inferred from the underlying data with no need for any user-defined parameters. We evaluate the proposed framework by comparing its performance with state-of-the-art techniques for error detection and data repairing. The empirical results show that TabReformer can significantly enhance the classification performance of erroneous values by up to 61.41% (F1 measure) while reducing the manual labeling budget by 31.77% on average. Also, the experimental evaluation depicts that the implemented data augmentation strategy outperforms other sampling techniques such as active learning strategies and traditional resampling approaches. Overall, the results empirically prove that TabReformer can detect a diverse set of errors, tolerate high noise ratios, and surpass existing error detection techniques.

References

[1] R. Lu, X. Jin, S. Zhang, M. Qiu, and X. Wu, "A Study on Big Knowledge and Its Engineering Issues," *IEEE Transactions on Knowledge and Data Engineering*, vol. 31, no. 9, pp. 1630–1644, 2019.

[2] X. Chu, I. F. Ilyas, and P. Papotti, "Holistic data cleaning: Putting violations into context," in *2013 IEEE 29th International Conference on Data Engineering (ICDE)*, Apr. 2013, pp. 458–469.

[3] T. Rekatsinas, X. Chu, I. F. Ilyas, and C. Ré, "HoloClean: holistic data repairs with probabilistic inference," *Proceedings of the VLDB Endowment*, vol. 10, no. 1, 2017.

[4] A. Reddy, M. Ordway-West, M. Lee, M. Dugan, J. Whitney, R. Kahana, B. Ford, J. Muedsam, A. Henslee, and M. Rao, "Using Gaussian Mixture Models to Detect Outliers in

Seasonal Univariate Network Traffic," in *2017 IEEE Security and Privacy Workshops (SPW)*, May 2017, pp. 229–234.

[5] C. Pit--Claudel, Z. Mariet, R. Harding, and S. Madden, "Outlier Detection in Heterogeneous Datasets using Automatic Tuple Expansion," 2016.

[6] F. Riahi and O. Schulte, "Model-based exception mining for object-relational data," *Data Mining and Knowledge Discovery*, vol. 34, no. 3, pp. 681–722, 2020.

[7] Y. Liu, Z. Li, C. Zhou, Y. Jiang, J. Sun, M. Wang, and X. He, "Generative Adversarial Active Learning for Unsupervised Outlier Detection," *IEEE Transactions on Knowledge and Data Engineering*, 2019.

[8] S. Schelter, D. Lange, P. Schmidt, M. Celikel, F. Biessmann, and A. Grafberger, "Automating large-scale data quality verification," *Proceedings of the VLDB Endowment*, vol. 11, no. 12, pp. 1781–1794, 2018.

[9] M. Dallachiesa, A. Ebaid, A. Eldawy, A, Elmagarmid, I. F. Ilyas, M. Ouzzani, and N. Tang, "NADEEF: a commodity data cleaning system," in *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, New York, USA, 2013, pp. 541–552.

[10] loannis Koumarelas, T. Papenbrock, and F. Naumann, "MDedup: Duplicate Detection with Matching Dependencies," *Proceedings of the VLDB Endowment*, vol. 13, no. 5, pp. 712–725, 2020.

[11] E. H. M. Pena, E. C. de Almeida, and F. Naumann, "Discovery of Approximate (and Exact) Denial Constraints," *Proceedings of the VLDB Endowment*, vol. 13, no. 3, pp. 266–278, 2019.

[12] X. Chu, I. F. Ilyas, S. Krishnan, and J. Wang, "Data Cleaning: Overview and Emerging Challenges," in *Proceedings of the 2016 International Conference on Management of Data*, New York, USA, 2016, pp. 2201–2206.

[13] Z. Abedjan, X. Chu, D. Deng, R. C. Fernandez, I. F. Ilyas, M. Ouzzani, P. Papotti, M. Stonebraker, and N. Tang, "Detecting Data Errors: Where Are We and What Needs to Be Done?," *Proceedings of the VLDB Endowment*, vol. 9, no. 12, pp. 993–1004, 2016.

[14] J. Yang, S. Rahardja, and P. Fränti, "Outlier Detection: How to Threshold Outlier Scores?," in *Proceedings of the International Conference on Artificial Intelligence, Information Processing and Cloud Computing*, New York, USA, 2019.

[15] A. Heidari, J. McGrath, I. F. Ilyas, and T. Rekatsinas, "HoloDetect: Few-Shot Learning for Error Detection," in *Proc. of the 2019 International Conference on Management of Data*, Netherlands, 2019, pp. 829–846.

[16] F. Neutatz, M. Mahdavi, and Z. Abedjan, "ED2: A Case for Active Learning in Error Detection," in *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, New York, USA, 2019, pp. 2249–2252.

[17] S. Krishnan, J. Wang, E. Wu, M. J. Franklin, and K. Goldberg, "ActiveClean: Interactive Data Cleaning for Statistical Modeling," *Proceedings of the VLDB Endowment*, vol. 9, no. 12, pp. 948–959, 2016.

[18] S. Krishnan, M. J. Franklin, K. Goldberg, and E. Wu, "BoostClean: Automated Error Detection and Repair for Machine Learning," *arXiv:1711.01299 [cs]*, 2017.

[19] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is All you Need," in *Advances in Neural Information Processing Systems*, Curran Associates, Inc., 2017, pp. 5998–6008.

[20] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-Based Models for Speech Recognition," in *Advances in Neural Information Processing Systems*, 2015, pp. 577–585.

[21] J. Krantz and J. Kalita, "Abstractive Summarization Using Attentive Neural Techniques," *arXiv:1810.08838 [cs]*, Oct. 2018.

[22] A. Sternberg, J. Soares, D. Carvalho, and E. Ogasawara, "A Review on Flight Delay Prediction," *arXiv:1703.06118 [cs]*, 2017.

[23] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," *arXiv:1810.04805 [cs]*, 2019.

[24] M. R. A. Rashid, G. Rizzo, M. Torchiano, N. Mihindukulasooriya, O. Corcho, and R. García-Castro, "Completeness and consistency analysis for evolving knowledge bases," *Journal of Web Semantics*, vol. 54, pp. 48–71, 2019.

[25] M. Farid, A. Roatis, I. F. Ilyas, H.-F. Hoffmann, and X. Chu, "CLAMS: Bringing Quality to Data Lakes," in *Proceedings of the 2016 International Conference on Management of Data*, San Francisco, California, USA, 2016, pp. 2089–2092.

[26] H. Saxena, L. Golab, and I. F. Ilyas, “Distributed Discovery of Functional Dependencies,” in *2019 IEEE 35th International Conference on Data Engineering*, Macao, 2019, pp. 1590–1593.

[27] E. K. Rezig, M. Ouzzani, W. G. Aref, A. K. Elmagarmid, and A. R. Mahmood, “Pattern-Driven Data Cleaning,” *arXiv:1712.09437 [cs]*, 2017.

[28] A. Qahtan, N. Tang, M. Ouzzani, Y. Cao, and M. Stonebraker, “Pattern functional dependencies for data cleaning,” *Proceedings of the VLDB Endowment*, vol. 13, no. 5, pp. 684–697, 2020.

[29] Z. Abedjan, L. Golab, and F. Naumann, “Profiling relational data: a survey,” *The VLDB Journal*, vol. 24, no. 4, pp. 557–581, 2015.

[30] E. D. Cubuk, B. Zoph, J. Shlens, and Q. V. Le, “RandAugment: Practical automated data augmentation with a reduced search space,” *arXiv:1909.13719 [cs]*, Nov. 2019.

[31] E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, and Q. V. Le, “AutoAugment: Learning Augmentation Strategies From Data,” presented at the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 2019, pp. 113–123.

[32] B. Zoph and Q. V. Le, “Neural Architecture Search with Reinforcement Learning,” *arXiv:1611.01578 [cs]*, 2017.

[33] D. Stoller, S. Ewert, and S. Dixon, “Adversarial Semi-Supervised Audio Source Separation Applied to Singing Voice Extraction,” in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Calgary, Alberta, Canada, 2018, pp. 2391–2395.

[34] S. Lim, I. Kim, T. Kim, C. Kim, and S. Kim, “Fast AutoAugment,” in *Advances in Neural Information Processing Systems*, Curran Associates, Inc., 2019, pp. 6665–6675.

[35] Y. Li, G. Hu, Y. Wang, T. Hospedales, N. M. Robertson, and Y. Yang, “DADA: Differentiable Automatic Data Augmentation,” *arXiv:2003.03780 [cs]*, 2020.

[36] T. Luong, H. Pham, and C. D. Manning, “Effective Approaches to Attention-based Neural Machine Translation,” in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, Lisbon, Portugal, 2015, pp. 1412–1421.

[37] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin, “Convolutional Sequence to Sequence Learning,” in *Proceedings of the 34th International Conference on Machine Learning*, 2017, pp. 1243–1252.

[38] K. Ahmed, N. S. Keskar, and R. Socher, “Weighted Transformer Network for Machine Translation,” *arXiv:1711.02132 [cs]*, Nov. 2017.

[39] Z. Dai, Z. Yang, Y. Yang, J. Carbonell, Q. Le, and R. Salakhutdinov, “Transformer-XL: Attentive Language Models beyond a Fixed-Length Context,” in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy, Jul. 2019, pp. 2978–2988.

[40] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, “RoBERTa: A Robustly Optimized BERT Pretraining Approach,” *arXiv:1907.11692 [cs]*, Jul. 2019.

[41] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, “ALBERT: A Lite BERT for Self-supervised Learning of Language Representations,” *arXiv:1909.11942 [cs]*, 2020.

[42] D. Hendrycks and K. Gimpel, “Gaussian Error Linear Units (GELUs),” *arXiv:1606.08415 [cs]*, 2018.

[43] J. Torres, C. Vaca, L. Terán, and C. L. Abad, “Seq2Seq models for recommending short text conversations,” *Expert Systems with Applications*, vol. 150, 2020.

[44] J. T. Hancock and T. M. Khoshgoftaar, “Survey on categorical data for neural networks,” *Journal of Big Data*, vol. 7, no. 1, 2020.

[45] H. Nam and H.-E. Kim, “Batch-Instance Normalization for Adaptively Style-Invariant Neural Networks,” in *Advances in Neural Information Processing Systems*, 2018, pp. 2558–2567.

[46] D. Ulyanov, A. Vedaldi, and V. Lempitsky, “Instance Normalization: The Missing Ingredient for Fast Stylization,” *arXiv:1607.08022 [cs]*, 2017.

[47] W. L. Taylor, “‘Cloze Procedure’: A New Tool for Measuring Readability,” *Journalism Quarterly*, vol. 30, no. 4, pp. 415–433, 1953.

[48] J. Wei and K. Zou, “EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks,” *arXiv:1901.11196 [cs]*, 2019.

[49] N. S. Tawfik and M. R. Spruit, “Evaluating sentence representations for biomedical text: Methods and experimental results,” *Journal of Biomedical Informatics*, vol. 104, Apr. 2020.

[50] Crane, David, "The Cost of Bad Data," Integrate, Inc, 201AD. [Online]. Available: https://demand.integrate.com/rs/951-JPP-414/images/Integrate_TheCostofBadLeads_Whitepaper.pdf.

[51] D. W. Cearley, "Top 10 Strategic Technology Trends for 2020," Gartner, 2020. [Online]. Available: https://www.gartner.com/en/publications/top-tech-trends-2020.

[52] D. Dua and C. Graff, *UCI Machine Learning Repository*. University of California, Irvine, School of Information and Computer Sciences, 2017.

[53] P. C. Arocena, B. Glavic, G. Mecca, R. J. Miller, P. Papotti, and D. Santoro, "Messing up with BART: Error Generation for Evaluating Data-Cleaning Algorithms," *Proceedings of the VLDB Endowment*, vol. 9, no. 2, pp. 36–47, Oct. 2015.

[54] Y. Fu, X. Zhu, and B. Li, "A survey on instance selection for active learning," *Knowledge and Information Systems*, vol. 35, no. 2, pp. 249–283, 2013.

[55] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *arXiv:1412.6980 [cs]*, Jan. 2017.

[56] A. Estabrooks, T. Jo, and N. Japkowicz, "A Multiple Resampling Method for Learning from Imbalanced Data Sets," *Computational Intelligence*, vol. 20, no. 1, pp. 18–36, 2004.

[57] S. Akcay, A. Atapour-Abarghouei, and T. P. Breckon, "GANomaly: Semi-supervised Anomaly Detection via Adversarial Training," in *Computer Vision – ACCV 2018*, 2019, pp. 622–637.

[58] E. Adeli, K-H Thung, L. An, G. Wu, F. Shi, T. Wang, and D. Shen, "Semi-Supervised Discriminative Classification Robust to Sample-Outliers and Feature-Noises," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 2, pp. 515–522, 2019.

[59] S. Eduardo and C. Sutton, "Data Cleaning using Probabilistic Models of Integrity Constraints," in *Neural Information Processing Systems*, 2016.

[60] G. Zhu, Q. Wang, Q. Tang, R. Gu, C. Yuan, and Y. Huang, "Efficient and Scalable Functional Dependency Discovery on Distributed Data-Parallel Platforms," *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, no. 12, pp. 2663–2676, 2019.

[61] J. N. Yan, O. Schulte, M. Zhang, J. Wang, and R. Cheng, "SCODED: Statistical Constraint Oriented Data Error Detection," presented at the SIGMOD'20, Portland, OR, USA, 2020.

[62] C. De Sa, I. F. Ilyas, B. Kimelfeld, C. Re, and T. Rekatsinas, “A Formal Framework For Probabilistic Unclean Databases,” in *International Conference on Database Theory (ICDT 2019)*, 2019.

[63] K. Chaitanya, N. Karani, C. F. Baumgartner, A. Becker, O. Donati, and E. Konukoglu, “Semi-supervised and Task-Driven Data Augmentation,” in *Information Processing in Medical Imaging*, 2019, pp. 29–41.

[64] S. Liu, J. Zhang, Y. Chen, Y. Liu, Z. Qin, and T. Wan, “Pixel Level Data Augmentation for Semantic Image Segmentation Using Generative Adversarial Networks,” in *2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 1902–1906.

[65] Q. Xie, Z. Dai, E. Hovy, M.-T. Luong, and Q. V. Le, *Unsupervised Data Augmentation for Consistency Training*. 2020.

[66] L. Zhang, G.-J. Qi, L. Wang, and J. Luo, “AET vs. AED: Unsupervised Representation Learning by Auto-Encoding Transformations Rather Than Data,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, USA, 2019, pp. 2547–2555.

[67] M. Freitag, S. Amiriparian, S. Pugachevskiy, N. Cummins, and B. Schuller, “auDeep: unsupervised learning of representations from audio with deep recurrent neural networks,” *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 6340–6344, 2017.

[68] S. O. Arik and T. Pfister, “TabNet: Attentive Interpretable Tabular Learning,” *arXiv:1908.07442 [cs, stat]*, Feb. 2020.

Chapter 7 : Conclusions and Future Studies

To better explore the challenges associated with data preparation tasks for tabular datasets, we have proposed several frameworks and classification algorithms. Some of these techniques have been applied to real-world situations to provide high-quality training datasets. Other proposed methods aim to find data quality issues in tabular databases to prepare them for the data analytics pipeline. In this chapter, we briefly review the major contributions presented in this research and point out what could be some interesting research topics for future studies.

7.1. Major Contributions

- We experimentally verified that existing techniques for labeling might not be feasible for big real-world data. For example, although active learning can result in accurate predictive models with minimum labeling effort when the amount of unlabeled data is large, active learning gets very expensive, especially with high dimensional data. Similarly, weak supervision approaches do not allow the end-user to evaluate and understand the level of noise in the output of the weak sources, which may deteriorate the final model performance.
- As a result, we present a new hybrid method for labeling massive training datasets. The technique uses traditional active learning within the data programming process to optimize user engagement. The experimental results show that the proposed technique can outperform data programming in labeling accuracy and predictive performance. Also, when compared to active learning, the proposed method can maintain less labeling cost, with a percentage decrease up to 53% compared to active learning.
- To further enhance the generated labels, we propose an end-to-end framework to produce high-quality, large-scale training datasets. The framework does not require the user to define any weak sources. Instead, it applies a novel process of automatic generation of labeling heuristics. Also, the framework employs a data-driven active learning phase to improve the accuracy of the weak labels. Instead of using traditional query strategies, the proposed system learns the selection policy according to the distribution of the underlying data. We evaluated the framework within ten datasets of varying sizes with a maximum size of 11 million records.

The results illustrate the effectiveness of the framework in producing high-quality labels and achieving high classification accuracy with minimal annotation efforts

- Moreover, to test the feasibility of the labeling framework and investigate different challenges for applying machine learning in the business domain, we propose M-Lean, an end-to-end development framework develop, evaluate and, deploy predictive products in business domains. We used M-Lean along with our labeling technique within a longitudinal case study with the help of our industrial partner. Over more than nine months, we worked to coordinate the application of the proposed frameworks. The results of the case study attest that M-Lean can help organizations utilize their stored datasets to build predictive models effectively.
- We pointed out that the different types of weak supervision may coexist in real-world situations. Therefore, machine learning algorithms must learn to deal with cases of compound weakly supervised learning. Specifically, we propose a classification algorithm to deal with inaccurate and incomplete data. To learn with the presence of noise, the model applies ensemble learning in semi-supervised settings to determine labeling confidence of each data point in the input data. Then, to correct the class labels of these points and resolve incomplete supervision, the method applies an iterative process of meta-active learning to select which points should be rectified by end-users. The results obtained from the experiments show that the proposed method can significantly statistically outperform state-of-the-art techniques, especially with high rates of noise.
- We verified that attention mechanisms and representation learning could help define dependency relationships between different attributes in tabular data. Thus, this technique can be further applied to detect various sources for data errors. We, therefore, propose a learning framework for detecting erroneous values in tabular data. The framework learns a bidirectional model to model the data representation. Also, to effectively fine-tune the model with the task of error detection, the model introduces a phase of data augmentation to generate synthetic labeled examples while optimizing manual labeling effort. When compared to state-of-the-art state techniques for error detection and data repairing, the proposed framework could enhance the classification performance of erroneous values while optimizing the manual labeling budget.

7.1. Future Studies

Although we have examined many important topics so far, we believe that there are many directions for future research that have been opened by the research presented here. We list some of these directions, which are of interest to be investigated in future studies.

- Database cleaning: Although we pointed out that modeling data representation can lead to satisfactory performance in erroneous data detection. The ultimate goal we aim to achieve is to get consistent query answering. Consistent query answering aims at obtaining meaningful answers to queries from inconsistent or noisy data. Therefore, in our work, we target at designing a framework that can provide end-to-end support for data manipulation during in-database data cleansing. Right now, most relational and document databases only allow operations such as search, insert, update, remove, workload analysis, and rollback activities to the data. However, for a next-generation database system, end-users would like to enable in-database machine learning and data preparation while maintaining speed and scalability.
- Handling unstructured data: The research verifies the effectiveness of the proposed frameworks when applied to structured and tabular data. However, the studies show that significant portions of data in many organizations are still in the unstructured format. Therefore, aiming to provide a complete data preprocessing solution, we investigate the effectiveness of applying the proposed techniques to unstructured data.
- Collaborative development of predictive models: we have studied some of the challenges that face machine learning applications in the business domain. However, we believe that applying the M-Lean framework to other case studies in different domains can lead to designing a holistic approach for designing, evaluating, and evaluating machine learning while incorporating the input from end-users. Such a collaborative approach should allow the domain experts with and without statistical background to understand the machine learning pipeline, interpret the rationale behind the decisions of the learning models, and give feedback to rectify any errors.

Bibliography

"What data scientists tell us about AI model training today," Alegion, 2019. [Online]. Available: https://content.alegion.com/dimensional-researchs-survey.

M. Vlachos, V. G. Vassiliadis, R. Heckel, and A. Labbi, "Toward interpretable predictive models in B2B recommender systems," *IBM Journal of Research and Development*, 2016.

A. J. Ratner, C. M. De Sa, S. Wu, D. Selsam, and C. Ré, "Data Programming: Creating Large Training Sets, Quickly," in *Advances in Neural Information Processing Systems*, 2016, pp. 3567–3575.

Y. Fu, X. Zhu, and B. Li, "A survey on instance selection for active learning," *Knowledge and Information Systems*, vol. 35, no. 2, pp. 249–283, 2013.

H. Zamani and W. B. Croft, "On the theory of weak supervision for information retrieval," in *ACM International Conference on Theory of Information Retrieval*, 2018.

A. Ratner, S. H. Bach, H. Ehrenberg, J. Fries, S. Wu, and C. Ré, "Snorkel: rapid training data creation with weak supervision," *VLDB Endowment*, vol. 11, pp. 269–282, 2017.

P. Varma and C. Ré, "Snuba: automating weak supervision to label training data," *VLDB Endowment*, vol. 12, 2018.

S. H. Bach, B. He, A. Ratner, and C. Ré, "Learning the Structure of Generative Models without Labeled Data," *ArXiv170300854 Cs Stat*, 2017.

G. V. Cormack and M. R. Grossman, "Scalability of Continuous Active Learning for Reliable High-Recall Text Classification," in *ACM International Conference on Information and Knowledge Management*, 2016.

H. Yu, X. Yang, S. Zheng, and C. Sun, "Active Learning from Imbalanced Data: A Solution of Online Weighted Extreme Learning Machine," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 4, 2019.

E.-C. Huang, H.-K. Pao, and Y.-J. Lee, "Big active learning," in *IEEE International Conference on Big Data*, Boston, MA, USA, 2017, pp. 94–101.

K. Wang, D. Zhang, Y. Li, R. Zhang, and L. Lin, "Cost-Effective Active Learning for Deep Image Classification," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 27, no. 12, pp. 2591–2600, 2017.

P. Jain and A. Kapoor, “Active learning for large multi-class problems,” in *IEEE Conference on Computer Vision and Pattern Recognition*, Miami, FL, USA, 2009, pp. 762–769.

S. Ertekin, J. Huang, L. Bottou, and L. Giles, “Learning on the border: active learning in imbalanced data classification,” in *ACM conference on information and knowledge management*, Lisbon, Portugal, 2007, pp. 127–136.

M. E. Ramirez-Loaiza, M. Sharma, G. Kumar, and M. Bilgic, “Active learning: an empirical study of common baselines,” *Data Mining and Knowledge Discovery*, vol. 31, no. 2, pp. 287–313, 2017.

P. Varma, D. Iter, C. De Sa, and C. Ré, “Flipper: A Systematic Approach to Debugging Training Sets,” in *Workshop on Human-In-the-Loop Data Analytics*, 2017.

P. Varma, B. He, D. Iter, P. Xu, R. Yu, C. D. Sa, and C. Ré, “Socratic Learning: Augmenting Generative Models to Incorporate Latent Subsets in Training Data,” *ArXiv161008123 Cs Stat*, 2017.

M. Liu, W. Buntine, and G. Haffari, “Learning How to Actively Learn: A Deep Imitation Learning Approach,” in *Annual Meeting of the Association for Computational Linguistics*, Melbourne, Australia, 2018, pp. 1874–1883.

G. Beatty, E. Kochis, and M. Bloodgood, “The Use of Unlabeled Data Versus Labeled Data for Stopping Active Learning for Text Classification,” in *IEEE International Conference on Semantic Computing*, 2019, pp. 287–294.

M. Bloodgood and K. Vijay-Shanker, “A method for stopping active learning based on stabilizing predictions and the need for user-adjustable stopping,” in *Conference on Computational Natural Language Learning*, Boulder, Colorado, 2009, pp. 39–47.

S. Hickson, A. Angelova, I. Essa, and R. Sukthankar, “Object category learning and retrieval with weak supervision,” *ArXiv Prepr. ArXiv180108985*, 2018.

J. Xu, A. G. Schwing, and R. Urtasun, “Learning to Segment Under Various Forms of Weak Supervision,” in *IEEE Conference on Computer Vision and Pattern Recognition*, Boston, MA, USA, 2015.

A. Ratner, B. Hancock, J. Dunnmon, R. Goldman, and C. Ré, “Snorkel MeTaL: Weak Supervision for Multi-Task Learning,” in *the Second Workshop on Data Management for End-To-End Machine Learning*, New York, NY, USA, 2018.

S. Wu, L. Hsiao, X. Cheng, B. Hancock, T. Rekatsinas, P. Levis, and C. Ré, “Fonduer:

Knowledge Base Construction from Richly Formatted Data," in *International Conference on Management of Data*, Houston, USA, 2018, pp. 1301–1316.

Y.-L. Tsou and H.-T. Lin, "Annotation cost-sensitive active learning by tree sampling," *Machine Learning*, 2019.

S. D. Bhattacharjee, W. J. Tolone, and V. S. Paranjape, "Identifying malicious social media contents using multi-view Context-Aware active learning," *Future Generation Computer Systems*, vol. 100, pp. 365–379, 2019.

L. F. S. Coletta, M. Ponti, E. R. Hruschka, A. Acharya, and J. Ghosh, "Combining clustering and active learning for the detection and learning of new image classes," *Neurocomputing*, vol. 358, pp. 150–165, 2019.

W. Fu, M. Wang, S. Hao, and X. Wu, "Scalable Active Learning by Approximated Error Reduction," in *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, New York, NY, USA, 2018, pp. 1396–1405.

D. Kang, D. Raghavan, P. Bailis, and M. Zaharia, "Model assertions for debugging machine learning," in *NeurIPS MLSys Workshop*, 2018.

M. Carbonneau, E. Granger, and G. Gagnon, "Bag-Level Aggregation for Multiple-Instance Active Learning in Instance Classification Problems," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 5, pp. 1441–1451, May 2019.

M. Bilal, L. O. Oyedele, J. Qadir, K. Munir, S. O. Ajayi, O. O. Akinade, H. A. Owolabi, H. A. Alaka, and M. Pasha, "Big Data in the construction industry: A review of present status, opportunities, and future trends," *Advanced engineering informatics*, 2016.

S. Yin and O. Kaynak, "Big data for modern industry: challenges and trends [point of view]," *Proceedings of the IEEE*, 2015.

X. Jin, B. W. Wah, X. Cheng, and Y. Wang, "Significance and challenges of big data research," *Big Data Research*, 2015.

I. A. T. Hashem, I. Yaqoob, N. B. Anuar, S. Mokhtar, A. Gani, and S. U. Khan, "The rise of 'big data' on cloud computing: Review and open research issues," *Information Systems*, 2015.

C. P. Chen and C.-Y. Zhang, "Data-intensive applications, challenges, techniques and technologies: A survey on Big Data," *Information Sciences*, 2014.

N. Gordini and V. Veglio, "Customers churn prediction and marketing retention strategies. An application of support vector machines based on the AUC parameter-selection technique in

B2B e-commerce industry," *Industrial Marketing Management*, 2017.

V. Tsoukalas and N. Fragiadakis, "Prediction of occupational risk in the shipbuilding industry using multivariable linear regression and genetic algorithm analysis," *Safety Science*, 2016.

M. Chen, Y. Hao, K. Hwang, L. Wang, and L. Wang, "Disease prediction by machine learning over big data from healthcare communities," *IEEE Access*, 2017.

S. Erevelles, N. Fukawa, and L. Swayne, "Big Data consumer analytics and the transformation of marketing," *Journal of Business Research*, 2016.

A. Y. L. Chong, E. Ch'ng, M. J. Liu, and B. Li, "Predicting consumer product demands via Big Data: the roles of online promotional marketing and online reviews," *International Journal of Production Research*, 2017.

M. Salehan and D. J. Kim, "Predicting the performance of online consumer reviews: A sentiment mining approach to big data analytics," *Decision Support Systems*, 2016.

E. Ries, The lean startup: How today's entrepreneurs use continuous innovation to create radically successful businesses. Crown Books, 2011.

R. Malhotra, "A systematic review of machine learning techniques for software fault prediction," *Applied Soft Computing*, 2015.

J. Wen, S. Li, Z. Lin, Y. Hu, and C. Huang, "Systematic literature review of machine learning based software development effort estimation models," *Information and Software Technology*, 2012.

M. Shepperd, D. Bowes, and T. Hall, "Researcher bias: The use of machine learning in software defect prediction," *IEEE Transactions on Software Engineering*, 2014.

Q. Song, X. Zhu, G. Wang, H. Sun, H. Jiang, C. Xue, B. Xu, and W. Song, "A machine learning based software process model recommendation method," *Journal of Systems and Software*, 2016.

Z. Kurtanović and W. Maalej, "Automatically classifying functional and non-functional requirements using supervised machine learning," in *Requirements Engineering Conference (RE), 2017 IEEE 25th International*, 2017.

A. Perini, A. Susi, and P. Avesani, "A machine learning approach to software requirements prioritization," *IEEE Transactions on Software Engineering*, 2012.

P. Avesani, A. Perini, A. Siena, and A. Susi, "Goals at risk? Machine learning at support of

early assessment," in *Requirements Engineering Conference (RE), 2015 IEEE 23rd International*, 2015.

S. Landset, T. M. Khoshgoftaar, A. N. Richter, and T. Hasanin, "A survey of open source tools for machine learning with big data in the Hadoop ecosystem," *Journal of Big Data*, 2015.

X. Meng, J. Bradley, B. Yavuz, E. Sparks, S. Venkataraman, D. Liu, J. Freeman, D. Tsai, M. Amde, S. Owen, D. Xin, R. Xin, M. J. Franklin, R. Zadeh, M. Zaharia, and A. Talw, "Mllib: Machine learning in apache spark," *The Journal of Machine Learning Research*, 2016.

E. R. Sparks, S. Venkataraman, T. Kaftan, M. J. Franklin, and B. Recht, "Keystoneml: Optimizing pipelines for large-scale advanced analytics," in *IEEE 33rd International Conference on Data Engineering*, 2017.

M. Vartak, H. Subramanyam, W-E. Lee, S. Viswanathan, S. Husnoo, S. Madden, and M. Zaharia, "Model DB: a system for machine learning model management," in *Proceedings of the Workshop on Human-In-the-Loop Data Analytics*, 2016.

B. R. Bodenmann and K. W. Axhausen, "Synthesis report on the state of the art on firmographics," *Institute for Transport Planning and Systems, ETH, Zurich*, 2010.

P. Sugimura and F. Hartl, "Building a Reproducible Machine Learning Pipeline," *arXiv preprint arXiv:1810.04570*, 2018.

R. L. Baskerville and A. T. Wood-Harper, "A critical perspective on action research as a method for information systems research," *Journal of information Technology*, 1996.

N. Carter, D. Bryant-Lukosius, A. DiCenso, J. Blythe, and A. J. Neville, "The use of triangulation in qualitative research.," in *Oncology nursing forum*, 2014.

R. B. Svensson, T. Gorschek, B. Regnell, R. Torkar, A. Shahrokni, and R. Feldt, "Quality requirements in industrial practice—an extended interview study at eleven companies," *IEEE Transactions on Software Engineering*, 2012.

I. Etikan, S. A. Musa, and R. S. Alkassim, "Comparison of convenience sampling and purposive sampling," *American Journal of Theoretical and Applied Statistics*, 2016.

E. Souza, A. Moreira, J. Araújo, S. Abrahão, E. Insfran, and D. S. da Silveira, "Comparing business value modeling methods: A family of experiments," *Information and Software Technology*, 2018.

I. Hadar, P. Soffer, and K. Kenzi, "The role of domain knowledge in requirements elicitation via interviews: an exploratory study," *Requirements Engineering*, 2014.

A. Sutcliffe and P. Sawyer, “Requirements elicitation: Towards the unknown unknowns,” in *Requirements Engineering Conference (RE), 2013 21st IEEE International*, 2013.

S. K. Kwak and J. H. Kim, “Statistical data preparation: management of missing values and outliers,” *Korean Journal of anesthesiology*, 2017.

M. Ortega, M. Pérez, and T. Rojas, “Construction of a systemic quality model for evaluating a software product,” *Software Quality Journal*, 2003.

H. Brink, J. W. Richards, M. Fetherolf, and B. Cronin, *Real-world machine learning*. Manning, 2017.

S. Easterbrook, J. Singer, M.-A. Storey, and D. Damian, “Selecting empirical methods for software engineering research,” in *Guide to advanced empirical software engineering*, Springer, 2008.

S. Liu, X. Wang, M. Liu, and J. Zhu, “Towards better analysis of machine learning models: A visual analytics perspective,” *Visual Informatics*, 2017.

D. Sculley, T. Phillips, D. Ebner, V. Chaudhary, and M. Young, “Machine learning: The high-interest credit card of technical debt,” 2014.

L. Li, S. Chen, J. Kleban, and A. Gupta, “Counterfactual estimation and optimization of click metrics in search engines: A case study,” in *Proceedings of the 24th International Conference on World Wide Web*, 2015.

P. Runeson and M. Höst, “Guidelines for conducting and reporting case study research in software engineering,” *Empirical software engineering*, 2009.

A. T. Bahill and A. M. Madni, “Discovering system requirements,” in *Tradeoff Decisions in System Design*, Springer, 2017.

S. Tueno, R. Laleau, A. Mammar, and M. Frappier, “The SysML/KAOS Domain Modeling Approach,” *arXiv preprint arXiv:1710.00903*, 2017.

S. Srinivas and A. R. Ravindran, “Optimizing outpatient appointment system using machine learning algorithms and scheduling rules: A prescriptive analytics framework,” *Expert Systems with Applications*, 2018.

R. Caruana, N. Karampatziakis, and A. Yessenalina, “An Empirical Evaluation of Supervised Learning in High Dimensions,” in *Proceedings of the 25th International Conference on Machine Learning*, 2008.

D. W. Hosmer Jr, S. Lemeshow, and R. X. Sturdivant, *Applied logistic regression*, vol. 398.

John Wiley & Sons, 2013.

P.-N. Tan, M. Steinbach, and V. Kumar, "Classification: alternative techniques," *Introduction to data mining*, 2005.

T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," in Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016.

D. M. Powers, "Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation," 2011.

T. Fawcett, "ROC graphs: Notes and practical considerations for researchers," *Machine learning*, 2004.

M. Nashaat, A. Ghosh, J. Miller, S. Quader, C. Marston, and J. Puget. "Hybridization of Active Learning and Data Programming for Labeling Large Industrial Datasets." In *2018 IEEE International Conference on Big Data (Big Data)*, 2018.

C. Wohlin and A. Aurum, "Towards a decision-making structure for selecting a research design in empirical software engineering," *Empirical Software Engineering*, 2015.

"Special Edition on Advanced Analytics in Banking," McKinsey&Company.

H. A. Akkermans and K. E. Van Oorschot, "Relevance assumed: a case study of balanced scorecard development using system dynamics," in *System Dynamics*, Springer, 2018.

F. Shull, J. Singer, and D. I. Sjøberg, *Guide to advanced empirical software engineering*. Springer, 2007.

W. Zhao, G. Guan, L. Chen, X. He, D. Cai, B. Wang, and Q. Wang, "Weakly-Supervised Deep Embedding for Product Review Sentiment Analysis," *IEEE Trans. Knowl. Data Eng*., vol. 30, no. 1, pp. 185–197, 2018.

V. S. Sheng, J. Zhang, B. Gu, and X. Wu, "Majority Voting and Pairing with Multiple Noisy Labeling," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 7, pp. 1355–1368, 2019.

P. Cheng, X. Lian, X. Jian, and L. Chen, "FROG: A Fast and Reliable Crowdsourcing Framework," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 5, pp. 894–908, 2019.

C. De Sa, A. Ratner, C. Ré, J. Shin, F. Wang, S. Wu, and C. Zhang, "DeepDive: Declarative Knowledge Base Construction," *SIGMOD Rec.*, vol. 45, no. 1, pp. 60–67, 2016.

N. Gurjar, S. Sudholt, and G. A. Fink, "Learning Deep Representations for Word Spotting under Weak Supervision," *International Workshop on Document Analysis Systems*, pp. 7-12, 2018.

S. Chaidaroon, T. Ebesu, and Y. Fang, “Deep Semantic Text Hashing with Weak Supervision,” *ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1109-1112, 2018.

A. H. Akbarnejad and M. S. Baghshah, “An Efficient Semi-Supervised Multi-label Classifier Capable of Handling Missing Labels,” *IEEE Trans. Knowl. Data Eng.*, vol. 31, pp. 229–242, 2019.

T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, “Improved Techniques for Training GANs,” *Advances in neural information processing systems*, pp. 2234-2242, 2016.

Z.-H. Zhou, “A brief introduction to weakly supervised learning,” *National Science Review*, vol. 5, no. 1, pp. 44–53, 2017.

M.-F. Balcan, S. Hanneke, and J. W. Vaughan, “The true sample complexity of active learning,” *Machine learning*, vol. 80, no. no. 2-3, pp. 111-139, 2010.

B. Settles, “Active Learning Literature Survey,” 2009.

N. Dalvi, A. Dasgupta, R. Kumar, and V. Rastogi, “Aggregating Crowdsourced Binary Ratings,” *Proc. International Conference on World Wide Web*, pp. 285-294, 2013.

M. Joglekar, H. Garcia-Molina, and A. Parameswaran, “Comprehensive and reliable crowd assessment algorithms,” *IEEE International Conference on Data Engineering*, pp. 195-206, 2015.

N. Das, S. Chaba, S. Gandhi, D. H. Chau, and X. Chu, “GOGGLES: Automatic Training Data Generation with Affinity Coding,” *ArXiv190304552 Cs*, 2019.

J. Zhu, H. Wang, B. K. Tsou, and M. Ma, “Active Learning with Sampling by Uncertainty and Density for Data Annotations,” *IEEE Trans. Audio Speech Lang. Process.*, vol. 18, no. 6, 2010.

R. B. C. Prudencio and T. B. Ludermir, “Active Meta-Learning with Uncertainty Sampling and Outlier Detection,” *IEEE International Joint Conference on Neural Networks*, pp. 346-351, 2008.

K. Konyushkova, R. Sznitman, and P. Fua, “Introducing Geometry in Active Learning for Image Segmentation,” *ArXiv150804955 Cs*, 2015.

Y. Yang, Z. Ma, F. Nie, X. Chang, and A. G. Hauptmann, “Multi-Class Active Learning by Uncertainty Sampling with Diversity Maximization,” *International Journal of Computer Vision*, vol. 113, no. 2, pp.113-127, 2015.

M. Fang, Y. Li, and T. Cohn, “Learning how to Active Learn: A Deep Reinforcement Learning Approach,” *ArXiv170802383 Cs*, Aug. 2017.

K. Konyushkova, R. Sznitman, and P. Fua, “Learning Active Learning from Data,” Advances in Neural Information Processing Systems, 2017.

H. Chu and H. Lin, “Can Active Learning Experience be Transferred?,” *IEEE International Conference on Data Mining*, pp. 841-846, 2016.

K. Pang, M. Dong, Y. Wu, and T. Hospedales, “Meta-learning transferable active learning policies by deep reinforcement learning,” *ArXiv Prepr. ArXiv180604798*, 2018.

A. Niculescu-Mizil and R. Caruana, “Predicting Good Probabilities with Supervised Learning,” *Proc. International Conference on Machine Learning*, pp. 625-632, 2005.

O. Sagi and L. Rokach, “Ensemble learning: A survey,” *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.*, vol. 8, no. 4, 2018.

B. Desharnais, F. Camirand-Lemyre, P. Mireault, and C. D. Skinner, “Determination of Confidence Intervals in Non-normal Data: Application of the Bootstrap to Cocaine Concentration in Femoral Blood,” *J. Anal. Toxicol.*, vol. 39, no. 2, pp. 113-117, 2015.

R. Xia, Y. Pan, H. Lai, C. Liu, and S. Yan, “Supervised Hashing for Image Retrieval via Image Representation Learning,” *AAAI Conference on Artificial Intelligence*, pp. 2156–2162, 2014.

J. Bernard, M. Zeppelzauer, M. Lehmann, M. Müller, and M. Sedlmair, “Towards User-Centered Active Learning Algorithms,” *Comput. Graph. Forum*, vol. 37, no. 3, pp. 121-132. 2018.

M. Nashaat, A. Ghosh, J. Miller, S. Quader, and C. Marston, “M-Lean: An end-to-end development framework for predictive models in B2B scenarios,” *Inf. Softw. Technol.*, vol. 113, pp. 131–145, 2019.

S. Moro, P. Cortez, and P. Rita, “A data-driven approach to predict the success of bank telemarketing,” *Decis. Support Syst.*, vol. 62, pp.22-3, 2014.

I.-C. Yeh and C. Lien, “The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients,” *Expert Syst. Appl.*, vol. 36, no. 2, 2009.

P. Baldi, P. Sadowski, D. Whiteson, “Searching for exotic particles in high-energy physics with deep learning.” *Nature communications*, 2014.

L. M. Candanedo and V. Feldheim, “Accurate occupancy detection of an office room from light, temperature, humidity and CO 2 measurements using statistical learning models,” *Energy*

Build., vol. 112, pp. 28–39, 2016.

R.K. Bock, A. Chilingarian, M. Gaug, F. Hakl, T. Hengstebeck, M. Jiřina, J. Klaschka, E. Kotrč, P. Savický, S. Towers, A. Vaiciulis, and W. Wittek, "Methods for multidimensional event classification: a case study using images from a Cherenkov gamma-ray telescope," *Nucl. Instrum. Methods Phys. Res. Sect. Accel. Spectrometers Detect. Assoc. Equip.*, vol. 516, no. 2, pp. 511–528, 2004.

K. Fernandes, P. Vinagre, and P. Cortez, "A Proactive Intelligent Decision Support System for Predicting the Popularity of Online News," *Conference on Artificial Intelligence,* pp. 535–546, 2015.

H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms," *ArXiv170807747 Cs Stat*, 2017.

P. Varma, B. He, P. Bajaj, I. Banerjee, N. Khandwala, D. L. Rubin, and C. Ré, "Inferring Generative Model Structure with Static Analysis," *ArXiv170902477 Cs Stat*, 2017.

M. Nashaat, A. Ghosh, J. Miller, and S. Quader, "WeSAL: Applying Active Supervision to Find High-quality Labels at Industrial Scale," *the Hawaii International Conference on System Sciences*, submitted for publication.

A. C. Tan and D. Gilbert, "An Empirical Comparison of Supervised Machine Learning Techniques in Bioinformatics," *Proc. Conference on Bioinformatics*, vol. 19, pp. 219-222, 2003.

I. Teinemaa, M. Dumas, F. M. Maggi, and C. Di Francescomarino, "Predictive business process monitoring with structured and unstructured data," *International Conference on Business Process Management*, pp 401-417, 2016.

J. Kremer, K. Steenstrup Pedersen, and C. Igel, "Active learning with support vector machines," *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.*, vol. 4, no. 4, pp. 313–326, 2014.

T. Durand, N. Thome, and M. Cord, "SyMIL: MinMax Latent SVM for Weakly Labeled Data," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 12, pp. 6099–6112, 2018.

R. Stewart and S. Ermon, "Label-Free Supervision of Neural Networks with Physics and Domain Knowledge," *AAAI Conference on Artificial Intelligence*, pp. 2576- 2582, 2017.

L. Cao, W. Tao, S. An, J. Jin, Y. Yan, X. Liu, W. Ge, A. Sah, L. Battle, J. Sun, R. Chang, B. Westover, S. Madden, and M. Stonebraker, "Smile: A System to Support Machine Learning on EEG Data at Scale," *Proc. VLDB Endow.*, vol. 12, no. 12, pp. 2230-2241, 2019.

Y. Li, Y. l Wang, D. Yu, Y. Ning, P. Hu, and R. Zhao, "ASCENT: Active Supervision for

Semi-supervised Learning," *IEEE Trans. Knowl. Data Eng.*, 2019.

P. Bachman, A. Sordoni, and A. Trischler, "Learning Algorithms for Active Learning," *Proc. International Conference on Machine Learning*, vol. 70, pp. 301-310, 2017.

Z. Zhou, J. Y. Shin, S. R. Gurudu, M. B. Gotway, and J. Liang, "AFT*: Integrating Active Learning and Transfer Learning to Reduce Annotation Efforts," *ArXiv180200912 Cs Stat*, 2018.

Y.F. Li, L.Z. Guo, and Z.H. Zhou, "Towards Safe Weakly Supervised Learning," IEEE Trans. Pattern Analysis and Machine Intelligence, 2019.

J. Luengo, S.-O. Shim, S. Alshomrani, A. Altalhi, and F. Herrera, "CNC-NOS: Class Noise Cleaning By Ensemble Filtering And Noise Scoring," *Knowledge-Based Systems*, vol. 140, pp. 27–49, 2018.

D. P. Kingma, S. Mohamed, D. Jimenez Rezende, and M. Welling, "Semi-Supervised Learning With Deep Generative Models," *Adv Neural Inf Process Syst,* pp. 3581–3589, 2014.

M. Poel, "Detecting Mislabeled Data Using Supervised Machine Learning Techniques," *Augmented Cognition. Neurocognition and Machine Learning*, 2017.

M. Sabzevari, G. Martínez-Muñoz, and A. Suárez, "A Two-Stage Ensemble Method For The Detection Of Class-Label Noise," *Neurocomputing*, pp. 2374–2383, 2018.

L.P.F. Garcia, A.C. Lorena, S. Matwin, and A.C.P.L.F. de Carvalho, "Ensembles Of Label Noise Filters: A Ranking Approach," *Data Min Knowl Disc*, 2016.

D. Guan, H. Wei, W. Yuan, G. Han, Y. Tian, M. Al-Dhelaan, and A. Al-Dhelaan, "Improving Label Noise Filtering by Exploiting Unlabeled Data," *IEEE Access*, vol. 6, pp. 11154–11165, 2018.

S. García, J. Luengo, and F. Herrera, "Dealing with Noisy Data," *Data Preprocessing in Data Mining*, pp. 107–145, 2015.

A. Oliver, A. Odena, C.A. Raffel, E.D. Cubuk, and I. Goodfellow, "Realistic Evaluation of Deep Semi-Supervised Learning Algorithms," *Advances in Neural Information Processing Systems*, pp. 3235–3246, 2018.

L.-Z. Guo, F. Kuang, Z.-X. Liu, Y.-F. Li, N. Ma, and X.-H. Qie, "Weakly Supervised Learning Meets Ride-Sharing User Experience Enhancement," *arXiv preprint arXiv:2001.09027*, 2020.

Z.-Y. Zhang, P. Zhao, Y. Jiang, and Z.-H. Zhou, "Learning from Incomplete and Inaccurate Supervision," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, New York, NY, USA, 2019, pp. 1017–1025.

K. Chen, D. Guan, W. Yuan, B. Li, A. M. Khattak, and O. Alfandi, "A Novel Feature Selection-Based Sequential Ensemble Learning Method for Class Noise Detection in High-Dimensional Data," *Adv Data Mining and Applications*, pp. 55–65, 2018.

R. Saman, A. Ali, and J. Licheng, "Rough-KNN Noise-Filtered Convolutional Neural Network for Image Classification," *Frontiers in Artificial Intelligence and Applications*, pp. 265–275, 2019.

J. A. Sáez, M. Galar, J. Luengo, and F. Herrera, "INFFC: An Iterative Class Noise Filter Based On The Fusion Of Classifiers With Noise Sensitivity Control," *Information Fusion*, vol. 27, pp. 19–32, 2016.

B. Frenay and M. Verleysen, "Classification in the Presence of Label Noise: A Survey," *IEEE Trans. Neural Netw. Learn. Syst*, vol. 25, no. 5, pp. 845–869, 2014.

C.J. Mantas, J. Abellán, and J.G. Castellano, "Analysis Of Credal-C4.5 For Classification In Noisy Domains," *Expert Systems with Applications*, 2016.

Q. Miao, Y. Cao, G. Xia, M. Gong, J. Liu, and J. Song, "RBoost: Label Noise-Robust Boosting Algorithm Based on a Nonconvex Loss Function and the Numerically Stable Base Learners*," IEEE Trans. Neural Netw. Learn. Syst*, pp. 2216–2228, 2016.

P. Yang, J.T. Ormerod, W. Liu, C. Ma, A.Y. Zomaya, and J.Y.H. Yang, "AdaSampling for Positive-Unlabeled and Label Noise Learning with Bioinformatics Applications," *IEEE Trans. Cybern*, vol. 49, 2019.

X. Liu, D. Zachariah, J. Wågberg, and T.B. Schön, "Reliable Semi-Supervised Learning when Labels are Missing at Random," arXiv:1811.10947 [cs, stat], 2019.

V. Jain, N. Modhe, and P. Rai, "Scalable Generative Models for Multi-label Learning with Missing Labels," *Proc. Machine Learning Research*, pp. 1636–1644, 2017.

B. Du, T. Xinyao, Z. Wang, L. Zhang, and D. Tao, "Robust Graph-Based Semisupervised Learning for Noisy Labeled Data via Maximum Correntropy Criterion," *IEEE Trans. Cybern*, vol. 49, pp. 1440–1453, 2019.

Y. Ding, S. Yan, Y. Zhang, W. Dai, and L. Dong, "Predicting The Attributes Of Social Network Users Using A Graph-Based Machine Learning Method," *Computer Communications*, vol. 73, pp. 3–11, Jan. 2016.

M.R. Bouguelia, S. Nowaczyk, K.C. Santosh, and A. Verikas, "Agreeing To Disagree: Active Learning With Noisy Labels Without Crowdsourcing," *Int. J. Mach. Learn. & Cyber.*, vol. 9, no. 8, pp. 1307–1319, 2018.

C.H. Lin, M. Mausam, and D.S. Weld, "Active Learning with Unbalanced Classes and Example-

Generation Queries," *Proc. Sixth AAAI Conf. on Human Computation and Crowdsourcing*, 2018.

R. C. Prati, J. Luengo, and F. Herrera, "Emerging Topics And Challenges Of Learning From Noisy Data In Nonstandard Classification: A Survey Beyond Binary Class Noise," *Knowl Inf Syst*, vol. 60, pp. 63–97, 2019.

M. Sabzevari, G. Martínez-Muñoz, and A. Suárez, "Small Margin Ensembles Can Be Robust To Class-Label Noise," *Neurocomputing*, vol. 160, pp.18-33, 2015.

J. Demšar, "Statistical Comparisons of Classifiers over Multiple Data Sets," *Machine Learning Research*, vol. 7, pp. 1–30, 2006.

W. Gao, B.B. Yang, and Z.H. Zhou, "On the Resistance of Nearest Neighbor to Random Noisy Labels," arXiv:1607.07526 [cs], 2018.

H. Kumar and P.S. Sastry, "Robust Loss Functions for Learning Multi-class Classifiers," *2018 IEEE Conf. on Systems, Man, and Cybernetics (SMC)*, pp. 687–692, 2018.

M. Sabzevari, G. Martínez-Muñoz, and A. Suárez, "Vote-Boosting Ensembles," *Pattern Recognition*, vol. 83, pp. 119–133, Nov. 2018.

J. Zhang, M. Wu, and V.S. Sheng, "Ensemble Learning from Crowds," *IEEE Trans. Knowl and Data Eng*, vol. 31, pp. 1506–1519, 2019.

M. R. Smith and T. Martinez, "The Robustness Of Majority Voting Compared To Filtering Misclassified Instances In Supervised Classification Tasks," *Artif Intell Rev*, vol. 49, no. 1, pp. 105–130, 2018.

B. Wu, F. Jia, W. Liu, B. Ghanem, and S. Lyu, "Multi-Label Learning With Missing Labels Using Mixed Dependency Graphs," *Int J Comput Vis*, pp. 875–896, 2018.

Y. Cong, G. Sun, J. Liu, H. Yu, and J. Luo, "User Attribute Discovery With Missing Labels," *Pattern Recognition*, vol. 73, pp. 33–46, 2018.

M. Dehghani, A. Severyn, S. Rothe, and J. Kamps, "Learning to Learn from Weak Supervision by Full Supervision," *Proc. NIPS Workshop on Meta-Learning, 2017*.

C. Li, L. Jiang, and W. Xu, "Noise Correction To Improve Data And Model Quality For Crowdsourcing," *Engineering Applications of Artificial Intelligence*, 2019.

R. Lu, X. Jin, S. Zhang, M. Qiu, and X. Wu, "A Study on Big Knowledge and Its Engineering Issues," *IEEE Transactions on Knowledge and Data Engineering*, vol. 31, no. 9, pp. 1630–1644, 2019.

X. Chu, I. F. Ilyas, and P. Papotti, “Holistic data cleaning: Putting violations into context,” in *2013 IEEE 29th International Conference on Data Engineering (ICDE)*, Apr. 2013, pp. 458–469.

T. Rekatsinas, X. Chu, I. F. Ilyas, and C. Ré, “HoloClean: holistic data repairs with probabilistic inference,” *Proceedings of the VLDB Endowment*, vol. 10, no. 1, 2017.

A. Reddy, M. Ordway-West, M. Lee, M. Dugan, J. Whitney, R. Kahana, B. Ford, J. Muedsam, A. Henslee, and M. Rao, “Using Gaussian Mixture Models to Detect Outliers in Seasonal Univariate Network Traffic,” in *2017 IEEE Security and Privacy Workshops (SPW)*, May 2017, pp. 229–234.

C. Pit--Claudel, Z. Mariet, R. Harding, and S. Madden, “Outlier Detection in Heterogeneous Datasets using Automatic Tuple Expansion,” 2016.

F. Riahi and O. Schulte, “Model-based exception mining for object-relational data,” *Data Mining and Knowledge Discovery*, vol. 34, no. 3, pp. 681–722, 2020.

Y. Liu, Z. Li, C. Zhou, Y. Jiang, J. Sun, M. Wang, and X. He, “Generative Adversarial Active Learning for Unsupervised Outlier Detection,” *IEEE Transactions on Knowledge and Data Engineering*, 2019.

S. Schelter, D. Lange, P. Schmidt, M. Celikel, F. Biessmann, and A. Grafberger, “Automating large-scale data quality verification,” *Proceedings of the VLDB Endowment*, vol. 11, no. 12, pp. 1781–1794, 2018.

. Dallachiesa, A. Ebaid, A. Eldawy, A, Elmagarmid, I. F. Ilyas, M. Ouzzani, and N. Tang, “NADEEF: a commodity data cleaning system,” in *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, New York, USA, 2013, pp. 541–552.

loannis Koumarelas, T. Papenbrock, and F. Naumann, “MDedup: Duplicate Detection with Matching Dependencies,” *Proceedings of the VLDB Endowment*, vol. 13, no. 5, pp. 712–725, 2020.

E. H. M. Pena, E. C. de Almeida, and F. Naumann, “Discovery of Approximate (and Exact) Denial Constraints,” *Proceedings of the VLDB Endowment*, vol. 13, no. 3, pp. 266–278, 2019.

X. Chu, I. F. Ilyas, S. Krishnan, and J. Wang, “Data Cleaning: Overview and Emerging Challenges,” in *Proceedings of the 2016 International Conference on Management of Data*, New York, USA, 2016, pp. 2201–2206.

Z. Abedjan, X. Chu, D. Deng, R. C. Fernandez, I. F. Ilyas, M. Ouzzani, P. Papotti, M. Stonebraker, and N. Tang, "Detecting Data Errors: Where Are We and What Needs to Be Done?," *Proceedings of the VLDB Endowment*, vol. 9, no. 12, pp. 993–1004, 2016.

J. Yang, S. Rahardja, and P. Fränti, "Outlier Detection: How to Threshold Outlier Scores?," in *Proceedings of the International Conference on Artificial Intelligence, Information Processing and Cloud Computing*, New York, USA, 2019.

A. Heidari, J. McGrath, I. F. Ilyas, and T. Rekatsinas, "HoloDetect: Few-Shot Learning for Error Detection," in *Proc. of the 2019 International Conference on Management of Data*, Netherlands, 2019, pp. 829–846.

F. Neutatz, M. Mahdavi, and Z. Abedjan, "ED2: A Case for Active Learning in Error Detection," in *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, New York, USA, 2019, pp. 2249–2252.

S. Krishnan, J. Wang, E. Wu, M. J. Franklin, and K. Goldberg, "ActiveClean: Interactive Data Cleaning for Statistical Modeling," *Proceedings of the VLDB Endowment*, vol. 9, no. 12, pp. 948–959, 2016.

S. Krishnan, M. J. Franklin, K. Goldberg, and E. Wu, "BoostClean: Automated Error Detection and Repair for Machine Learning," *arXiv:1711.01299 [cs]*, 2017.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is All you Need," in *Advances in Neural Information Processing Systems*, Curran Associates, Inc., 2017, pp. 5998–6008.

J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-Based Models for Speech Recognition," in *Advances in Neural Information Processing Systems*, 2015, pp. 577–585.

J. Krantz and J. Kalita, "Abstractive Summarization Using Attentive Neural Techniques," *arXiv:1810.08838 [cs]*, Oct. 2018.

A. Sternberg, J. Soares, D. Carvalho, and E. Ogasawara, "A Review on Flight Delay Prediction," *arXiv:1703.06118 [cs]*, 2017.

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," *arXiv:1810.04805 [cs]*, 2019.

M. R. A. Rashid, G. Rizzo, M. Torchiano, N. Mihindukulasooriya, O. Corcho, and R. García-Castro, "Completeness and consistency analysis for evolving knowledge bases," *Journal of Web Semantics*, vol. 54, pp. 48–71, 2019.

M. Farid, A. Roatis, I. F. Ilyas, H.-F. Hoffmann, and X. Chu, "CLAMS: Bringing Quality to Data Lakes," in *Proceedings of the 2016 International Conference on Management of Data*, San Francisco, California, USA, 2016, pp. 2089–2092.

H. Saxena, L. Golab, and I. F. Ilyas, "Distributed Discovery of Functional Dependencies," in *2019 IEEE 35th International Conference on Data Engineering*, Macao, 2019, pp. 1590–1593.

E. K. Rezig, M. Ouzzani, W. G. Aref, A. K. Elmagarmid, and A. R. Mahmood, "Pattern-Driven Data Cleaning," *arXiv:1712.09437 [cs]*, 2017.

A. Qahtan, N. Tang, M. Ouzzani, Y. Cao, and M. Stonebraker, "Pattern functional dependencies for data cleaning," *Proceedings of the VLDB Endowment*, vol. 13, no. 5, pp. 684–697, 2020.

Z. Abedjan, L. Golab, and F. Naumann, "Profiling relational data: a survey," *The VLDB Journal*, vol. 24, no. 4, pp. 557–581, 2015.

E. D. Cubuk, B. Zoph, J. Shlens, and Q. V. Le, "RandAugment: Practical automated data augmentation with a reduced search space," *arXiv:1909.13719 [cs]*, Nov. 2019.

E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, and Q. V. Le, "AutoAugment: Learning Augmentation Strategies From Data," presented at the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 2019, pp. 113–123.

B. Zoph and Q. V. Le, "Neural Architecture Search with Reinforcement Learning," *arXiv:1611.01578 [cs]*, 2017.

D. Stoller, S. Ewert, and S. Dixon, "Adversarial Semi-Supervised Audio Source Separation Applied to Singing Voice Extraction," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Calgary, Alberta, Canada, 2018, pp. 2391–2395.

S. Lim, I. Kim, T. Kim, C. Kim, and S. Kim, "Fast AutoAugment," in *Advances in Neural Information Processing Systems*, Curran Associates, Inc., 2019, pp. 6665–6675.

Y. Li, G. Hu, Y. Wang, T. Hospedales, N. M. Robertson, and Y. Yang, "DADA: Differentiable Automatic Data Augmentation," *arXiv:2003.03780 [cs]*, 2020.

T. Luong, H. Pham, and C. D. Manning, “Effective Approaches to Attention-based Neural Machine Translation,” in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, Lisbon, Portugal, 2015, pp. 1412–1421.

J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin, “Convolutional Sequence to Sequence Learning,” in *Proceedings of the 34th International Conference on Machine Learning*, 2017, pp. 1243–1252.

K. Ahmed, N. S. Keskar, and R. Socher, “Weighted Transformer Network for Machine Translation,” *arXiv:1711.02132 [cs]*, Nov. 2017.

Z. Dai, Z. Yang, Y. Yang, J. Carbonell, Q. Le, and R. Salakhutdinov, “Transformer-XL: Attentive Language Models beyond a Fixed-Length Context,” in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy, Jul. 2019, pp. 2978–2988.

Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, “RoBERTa: A Robustly Optimized BERT Pretraining Approach,” *arXiv:1907.11692 [cs]*, Jul. 2019.

Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, “ALBERT: A Lite BERT for Self-supervised Learning of Language Representations,” *arXiv:1909.11942 [cs]*, 2020.

D. Hendrycks and K. Gimpel, “Gaussian Error Linear Units (GELUs),” *arXiv:1606.08415 [cs]*, 2018.

J. Torres, C. Vaca, L. Terán, and C. L. Abad, “Seq2Seq models for recommending short text conversations,” *Expert Systems with Applications*, vol. 150, 2020.

J. T. Hancock and T. M. Khoshgoftaar, “Survey on categorical data for neural networks,” *Journal of Big Data*, vol. 7, no. 1, 2020.

H. Nam and H.-E. Kim, “Batch-Instance Normalization for Adaptively Style-Invariant Neural Networks,” in *Advances in Neural Information Processing Systems*, 2018, pp. 2558–2567.

D. Ulyanov, A. Vedaldi, and V. Lempitsky, “Instance Normalization: The Missing Ingredient for Fast Stylization,” *arXiv:1607.08022 [cs]*, 2017.

W. L. Taylor, “‘Cloze Procedure’: A New Tool for Measuring Readability,” *Journalism Quarterly*, vol. 30, no. 4, pp. 415–433, 1953.

J. Wei and K. Zou, “EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks,” *arXiv:1901.11196 [cs]*, 2019.

N. S. Tawfik and M. R. Spruit, “Evaluating sentence representations for biomedical text: Methods and experimental results,” *Journal of Biomedical Informatics*, vol. 104, Apr. 2020.

Crane, David, “The Cost of Bad Data,” Integrate, Inc, 201AD. [Online]. Available: https://demand.integrate.com/rs/951-JPP-414/images/Integrate_TheCostofBadLeads_Whitepaper.pdf.

D. W. Cearley, “Top 10 Strategic Technology Trends for 2020,” Gartner, 2020. [Online]. Available: https://www.gartner.com/en/publications/top-tech-trends-2020.

D. Dua and C. Graff, *UCI Machine Learning Repository*. University of California, Irvine, School of Information and Computer Sciences, 2017.

P. C. Arocena, B. Glavic, G. Mecca, R. J. Miller, P. Papotti, and D. Santoro, “Messing up with BART: Error Generation for Evaluating Data-Cleaning Algorithms,” *Proceedings of the VLDB Endowment*, vol. 9, no. 2, pp. 36–47, Oct. 2015.

D. P. Kingma and J. Ba, “Adam: A Method for Stochastic Optimization,” *arXiv:1412.6980 [cs]*, Jan. 2017.

A. Estabrooks, T. Jo, and N. Japkowicz, “A Multiple Resampling Method for Learning from Imbalanced Data Sets,” *Computational Intelligence*, vol. 20, no. 1, pp. 18–36, 2004.

S. Akcay, A. Atapour-Abarghouei, and T. P. Breckon, “GANomaly: Semi-supervised Anomaly Detection via Adversarial Training,” in *Computer Vision – ACCV 2018*, 2019, pp. 622–637.

E. Adeli, K-H Thung, L. An, G. Wu, F. Shi, T. Wang, and D. Shen, “Semi-Supervised Discriminative Classification Robust to Sample-Outliers and Feature-Noises,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 2, pp. 515–522, 2019.

S. Eduardo and C. Sutton, “Data Cleaning using Probabilistic Models of Integrity Constraints,” in *Neural Information Processing Systems*, 2016.

G. Zhu, Q. Wang, Q. Tang, R. Gu, C. Yuan, and Y. Huang, “Efficient and Scalable Functional Dependency Discovery on Distributed Data-Parallel Platforms,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, no. 12, pp. 2663–2676, 2019.

J. N. Yan, O. Schulte, M. Zhang, J. Wang, and R. Cheng, “SCODED: Statistical Constraint Oriented Data Error Detection,” presented at the SIGMOD’20, Portland, OR, USA, 2020.

C. De Sa, I. F. Ilyas, B. Kimelfeld, C. Re, and T. Rekatsinas, “A Formal Framework For Probabilistic Unclean Databases,” in *International Conference on Database Theory (ICDT 2019)*, 2019.

K. Chaitanya, N. Karani, C. F. Baumgartner, A. Becker, O. Donati, and E. Konukoglu, “Semi-supervised and Task-Driven Data Augmentation,” in *Information Processing in Medical Imaging*, 2019, pp. 29–41.

S. Liu, J. Zhang, Y. Chen, Y. Liu, Z. Qin, and T. Wan, “Pixel Level Data Augmentation for Semantic Image Segmentation Using Generative Adversarial Networks,” in *2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 1902–1906.

Q. Xie, Z. Dai, E. Hovy, M.-T. Luong, and Q. V. Le, *Unsupervised Data Augmentation for Consistency Training*. 2020.

L. Zhang, G.-J. Qi, L. Wang, and J. Luo, “AET vs. AED: Unsupervised Representation Learning by Auto-Encoding Transformations Rather Than Data,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, USA, 2019, pp. 2547–2555.

M. Freitag, S. Amiriparian, S. Pugachevskiy, N. Cummins, and B. Schuller, “auDeep: unsupervised learning of representations from audio with deep recurrent neural networks,” *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 6340–6344, 2017.

S. O. Arik and T. Pfister, “TabNet: Attentive Interpretable Tabular Learning,” *arXiv:1908.07442 [cs, stat]*, Feb. 2020.

Appendix A. Interview Guidelines and Scripts

The M-Lean framework uses an iterative process of in-depth semi-structured interviews as a method for data collection in the first phase. As mentioned in Section 3.6.2, semi-structured interviews are organized as a set of open-ended questions, and depending on the answers of the interviewee to a given question, the follow-up question can change between different interviews. Nonetheless, once the interviewee set is identified, a set of common questions can be created for all the interviews. The topics of the questions in each layer were derived from the three research questions of this phase, discussed in Section 3.5.1. The questionnaires that were used to guide the interviews in each layer are presented in Table A.1.

Table A.1. Common questionnaire for interviews in Phase 1

Layer	Topic	Questions
Layer 1	Overall view of the stored datasets	What are the datasets managed by your team? Who are the users for these datasets? Can you elaborate on the database schemas for these datasets?
	Using ML techniques with datasets	Have any of these datasets been used in ML models within the organization? What ML algorithms that are mostly applied by your team?
	Quality of the datasets	Are the datasets well structured? How reliable are the sources for these datasets? How consistent are the datasets?
Layer 2	Overall business processes and business challenges	What are the challenges that face your team? In your business unit, do you have any business decisions that can be automated? What data sources do people in your team usually search for? Do you apply any automation technique to your customer interactions?
	Reasons and expectations for ML	What are your expectations for using ML in business processes?

		What are the metrics, if predicted, would have a positive impact on your team? What are the useful and challenging aspects of predictive systems? Do you think ML is the right approach to help the business process to be more efficient?
Layer 3	Overall Process of License Renewals	Can you elaborate on the overall process of the license renewals? What daily activities related to the renewal process that you do as a part of your job?
	Anticipating renewal risks	What software systems do you use in your job? Do you look up the customer's renewal history before contacting him/ her? What kind of information do you use to anticipate renewal risks? Do you use recommendations from predictive modes to anticipate renewal risks?
	Eliciting Requirements	What are the most important business requirements for a license cancellation predictive system? What are the key business data requirements for this system? How will these requirements help to add value to your team?

Appendix B. Performance Scores with Inaccurate and Incomplete Supervision

Table B.1: Accuracy values for different techniques with different noise levels (I)

	SVM	KNN	Logit	SVM	KNN	Logit	SVM	KNN	Logit
	activity (low)			activity (moderate)			activity (high)		
Gold	0.89 ± 0.16	0.83 ± 0.03	0.84 ± 0.02	0.89 ± 0.20	0.82 ± 0.18	0.85 ± 0.12	0.89 ± 0.19	0.84 ± 0.15	0.86 ± 0.13
Baseline	0.78 ± 0.01	0.72 ± 0.06	0.64 ± 0.06	0.72 ± 0.08	0.62 ± 0.12	0.33 ± 0.05	0.45 ± 0.06	0.25 ± 0.01	0.17 ± 0.06
S. Mendr	**0.86 ± 0.20**	**0.80 ± 0.20**	**0.84 ± 0.04**	**0.79 ± 0.11**	**0.74 ± 0.12**	**0.72 ± 0.19**	**0.63 ± 0.05**	**0.64 ± 0.09**	**0.62 ± 0.17**
Filtering	0.83 ± 0.17	0.79 ± 0.03	0.65 ± 0.19	0.64 ± 0.13	0.67 ± 0.07	0.62 ± 0.06	0.58 ± 0.17	0.36 ± 0.14	0.5 ± 0.15
Bagging	0.79 ± 0.10	0.73 ± 0.01	0.77 ± 0.13	0.73 ± 0.04	0.61 ± 0.15	0.70 ± 0.20	0.57 ± 0.11	0.61 ± 0.2	0.54 ± 0.12
	APS failure (low)			APS failure (moderate)			APS failure (high)		
Gold	0.92 ± 0.01	0.96 ± 0.09	0.92 ± 0.07	0.89 ± 0.03	0.98 ± 0.11	0.95 ± 0.06	0.85 ± 0.15	0.81 ± 0.19	0.91 ± 0.04
Baseline	0.70 ± 0.16	0.67 ± 0.02	0.60 ± 0.07	0.55 ± 0.10	0.53 ± 0.05	0.5 ± 0.17	0.52 ± 0.16	0.48 ± 0.18	0.43 ± 0.11
S. Mendr	**0.89 ± 0.02**	**0.89 ± 0.19**	**0.86 ± 0.18**	**0.84 ± 0.06**	**0.77 ± 0.18**	**0.81 ± 0.01**	**0.76 ± 0.07**	**0.74 ± 0.13**	**0.76 ± 0.02**
Filtering	0.86 ± 0.20	0.74 ± 0.13	0.74 ± 0.12	0.83 ± 0.13	0.64 ± 0.02	0.71 ± 0.05	0.67 ± 0.06	0.60 ± 0.16	0.62 ± 0.12
Bagging	**0.89 ± 0.07**	0.75 ± 0.01	0.72 ± 0.13	0.81 ± 0.20	0.75 ± 0.20	0.70 ± 0.07	0.72 ± 0.09	0.68 ± 0.01	0.69 ± 0.05
	avila (low)			avila (moderate)			avila (high)		
Gold	0.96 ± 0.06	0.96 ± 0.14	0.95 ± 0.04	0.96 ± 0.02	0.96 ± 0.18	0.95 ± 0.02	0.96 ± 0.14	0.96 ± 0.12	0.95 ± 0.12
Baseline	0.89 ± 0.16	0.79 ± 0.15	0.8 ± 0.13	0.74 ± 0.17	0.69 ± 0.03	0.60 ± 0.04	0.31 ± 0.20	0.40 ± 0.04	0.12 ± 0.11
S. Mendr	**0.93 ± 0.17**	**0.91 ± 0.16**	0.93 ± 0.03	**0.83 ± 0.17**	**0.86 ± 0.16**	0.80 ± 0.11	**0.81 ± 0.17**	**0.74 ± 0.15**	**0.76 ± 0.19**
Filtering	0.89 ± 0.11	0.90 ± 0.11	0.81 ± 0.11	0.76 ± 0.13	0.81 ± 0.17	**0.86 ± 0.03**	0.64 ± 0.22	0.67 ± 0.02	0.65 ± 0.18
Bagging	0.91 ± 0.07	0.85 ± 0.13	**0.94 ± 0.08**	0.79 ± 0.18	0.76 ± 0.01	0.72 ± 0.02	0.69 ± 0.02	0.71 ± 0.07	0.57 ± 0.10
	banana (low)			banana (moderate)			banana (high)		
Gold	0.98 ± 0.02	0.82 ± 0.07	0.91 ± 0.06	0.97 ± 0.02	0.84 ± 0.05	0.93 ± 0.11	0.98 ± 0.17	0.84 ± 0.14	0.93 ± 0.14
Baseline	0.74 ± 0.05	0.72 ± 0.12	0.63 ± 0.11	0.53 ± 0.12	0.45 ± 0.04	0.12 ± 0.18	0.46 ± 0.15	0.33 ± 0.05	0.22 ± 0.17
S. Mendr	0.92 ± 0.11	0.81 ± 0.09	0.9 ± 0.09	0.84 ± 0.20	0.69 ± 0.12	0.66 ± 0.11	0.75 ± 0.19	0.68 ± 0.09	0.61 ± 0.03
Filtering	0.80 ± 0.14	0.74 ± 0.15	0.79 ± 0.20	0.81 ± 0.12	0.55 ± 0.14	0.50 ± 0.19	0.65 ± 0.06	0.50 ± 0.15	0.32 ± 0.06
Bagging	0.83 ± 0.17	0.71 ± 0.19	0.7 ± 0.16	0.78 ± 0.13	0.62 ± 0.17	0.62 ± 0.06	0.66 ± 0.02	0.61 ± 0.07	0.50 ± 0.15
	census (low)			census (moderate)			census (high)		
Gold	0.92 ± 0.03	0.91 ± 0.02	0.89 ± 0.21	0.92 ± 0.05	0.93 ± 0.07	0.89 ± 0.15	0.93 ± 0.01	0.92 ± 0.11	0.89 ± 0.02
Baseline	0.82 ± 0.14	0.82 ± 0.16	0.67 ± 0.02	0.53 ± 0.18	0.79 ± 0.07	0.54 ± 0.08	0.35 ± 0.07	0.62 ± 0.17	0.51 ± 0.19
S. Mendr	**0.89 ± 0.03**	**0.88 ± 0.09**	**0.88 ± 0.06**	**0.81 ± 0.17**	0.83 ± 0.01	**0.84 ± 0.12**	**0.73 ± 0.07**	**0.83 ± 0.06**	**0.82 ± 0.14**
Filtering	0.82 ± 0.17	0.85 ± 0.04	0.71 ± 0.15	0.77 ± 0.16	0.81 ± 0.04	0.67 ± 0.08	0.65 ± 0.06	0.66 ± 0.03	0.56 ± 0.07
Bagging	0.81 ± 0.23	0.86 ± 0.01	0.87 ± 0.03	0.73 ± 0.13	**0.84 ± 0.01**	0.76 ± 0.16	0.67 ± 0.22	0.70 ± 0.06	0.61 ± 0.03
	connect4 (low)			connect4 (moderate)			connect4 (high)		
Gold	0.69 ± 0.06	0.58 ± 0.02	0.64 ± 0.14	0.68 ± 0.08	0.58 ± 0.17	0.63 ± 0.11	0.68 ± 0.11	0.59 ± 0.51	0.62 ± 0.02
Baseline	0.52 ± 0.06	0.45 ± 0.02	0.42 ± 0.08	0.39 ± 0.12	0.21 ± 0.03	0.40 ± 0.07	0.32 ± 0.01	0.17 ± 0.01	0.28 ± 0.14
S. Mendr	**0.66 ± 0.05**	**0.53 ± 0.14**	**0.61 ± 0.05**	**0.58 ± 0.06**	0.38 ± 0.05	**0.59 ± 0.01**	**0.52 ± 0.15**	**0.36 ± 0.13**	**0.56 ± 0.17**
Filtering	0.57 ± 0.11	0.49 ± 0.17	0.59 ± 0.01	0.49 ± 0.12	0.4 0± 0.05	0.45 ± 0.03	0.34 ± 0.08	0.30 ± 0.18	0.30 ± 0.06
Bagging	0.53 ± 0.05	0.51 ± 0.01	0.52 ± 0.18	0.43 ± 0.14	**0.41 ± 0.01**	0.42 ± 0.19	0.32 ± 0.12	**0.36 ± 0.18**	0.33 ± 0.16
	german (low)			german (moderate)			german (high)		
Gold	0.98 ± 0.18	0.94 ± 0.15	0.96 ± 0.15	0.97 ± 0.11	0.96 ± 0.20	0.97 ± 0.12	0.98 ± 0.01	0.95 ± 0.12	0.97 ± 0.12
Baseline	0.80 ± 0.00	0.82 ± 0.08	0.74 ± 0.02	0.69 ± 0.2	0.70 ± 0.13	0.53 ± 0.01	0.57 ± 0.2	0.57 ± 0.02	0.40 ± 0.17
S. Mendr	**0.93 ± 0.03**	**0.84 ± 0.02**	**0.92 ± 0.14**	**0.80 ± 0.17**	**0.83 ± 0.09**	**0.85 ± 0.14**	**0.84 ± 0.19**	**0.75 ± 0.15**	**0.77 ± 0.08**
Filtering	0.84 ± 0.16	0.82 ± 0.12	0.85 ± 0.17	0.73 ± 0.18	0.76 ± 0.16	0.68 ± 0.11	0.68 ± 0.05	0.69 ± 0.11	0.63 ± 0.08
Bagging	0.85 ± 0.18	0.83 ± 0.06	0.81 ± 0.17	0.79 ± 0.15	0.81 ± 0.11	0.77 ± 0.17	0.74 ± 0.18	0.73 ± 0.16	0.68 ± 0.03

Table B.2: Accuracy values for different techniques with different noise levels (II)

	SVM	KNN	Logit	SVM	KNN	Logit	SVM	KNN	Logit
	HTRU2 (low)			HTRU2 (moderate)			HTRU2 (high)		
Gold	0.97 ± 0.17	0.96 ± 0.19	0.96 ± 0.05	0.99 ± 0.01	0.94 ± 0.09	0.96 ± 0.17	0.97 ± 0.08	0.93 ± 0.05	0.95 ± 0.15
Baseline	0.42 ± 0.00	0.28 ± 0.08	0.22 ± 0.02	0.34 ± 0.17	0.20 ± 0.17	0.19 ± 0.04	0.18 ± 0.07	0.07 ± 0.04	0.11 ± 0.11
S. Mendr	**0.88 ± 0.16**	**0.82 ± 0.15**	**0.83 ± 0.18**	**0.83 ± 0.09**	**0.79 ± 0.08**	**0.73 ± 0.16**	**0.74 ± 0.02**	**0.72 ± 0.19**	**0.68 ± 0.05**
Filtering	0.72 ± 0.01	0.78 ± 0.08	0.74 ± 0.08	0.63 ± 0.04	0.72 ± 0.14	0.70 ± 0.18	0.54 ± 0.00	0.63 ± 0.11	0.53 ± 0.17
Bagging	0.74 ± 0.18	0.59 ± 0.05	0.64 ± 0.09	0.67 ± 0.12	0.54 ± 0.16	0.56 ± 0.16	0.58 ± 0.00	0.46 ± 0.10	0.52 ± 0.06
	MoCap (low)			MoCap (moderate)			MoCap (high)		
Gold	0.96 ± 0.04	0.93 ± 0.11	0.98 ± 0.15	0.95 ± 0.03	0.95 ± 0.09	0.96 ± 0.06	0.94 ± 0.11	0.95 ± 0.10	0.96 ± 0.05
Baseline	0.82 ± 0.13	0.77 ± 0.04	0.62 ± 0.09	0.65 ± 0.18	0.75 ± 0.16	0.60 ± 0.17	0.56 ± 0.13	0.63 ± 0.02	0.49 ± 0.15
S. Mendr	**0.94 ± 0.00**	**0.92 ± 0.13**	**0.95 ± 0.01**	0.80 ± 0.04	**0.82 ± 0.02**	**0.79 ± 0.09**	**0.70 ± 0.06**	**0.73 ± 0.02**	**0.71 ± 0.14**
Filtering	0.78 ± 0.08	0.83 ± 0.15	0.77 ± 0.11	0.75 ± 0.15	0.79 ± 0.01	0.71 ± 0.20	0.59 ± 0.04	0.62 ± 0.19	0.59 ± 0.12
Bagging	0.83 ± 0.10	0.85 ± 0.05	0.82 ± 0.11	**0.86 ± 0.06**	0.77 ± 0.05	0.74 ± 0.19	0.57 ± 0.06	0.67 ± 0.06	0.55 ± 0.16
	penbased (low)			penbased (moderate)			penbased (high)		
Gold	0.94 ± 0.06	0.92 ± 0.06	0.98 ± 0.16	0.95 ± 0.1	0.93 ± 0.16	0.91 ± 0.18	0.93 ± 0.08	0.91 ± 0.16	0.98 ± 0.08
Baseline	0.83 ± 0.12	0.80 ± 0.16	0.75 ± 0.12	0.69 ± 0.15	0.65 ± 0.16	0.56 ± 0.11	0.66 ± 0.01	0.38 ± 0.02	0.25 ± 0.14
S. Mendr	**0.91 ± 0.16**	**0.87 ± 0.08**	**0.94 ± 0.13**	**0.88 ± 0.09**	**0.74 ± 0.09**	**0.73 ± 0.03**	**0.81 ± 0.10**	**0.72 ± 0.10**	**0.69 ± 0.01**
Filtering	0.87 ± 0.15	0.83 ± 0.02	0.76 ± 0.07	0.79 ± 0.16	0.72 ± 0.05	0.64 ± 0.02	0.68 ± 0.01	0.52 ± 0.08	0.58 ± 0.04
Bagging	0.89 ± 0.07	0.84 ± 0.11	0.84 ± 0.07	0.80 ± 0.06	0.71 ± 0.07	0.71 ± 0.06	0.67 ± 0.03	0.69 ± 0.06	0.53 ± 0.07
	shoppers intention (low)			shoppers intention (moderate)			shoppers intention (high)		
Gold	0.99 ± 0.16	0.93 ± 0.00	0.92 ± 0.17	0.97 ± 0.08	0.95 ± 0.13	0.94 ± 0.00	0.96 ± 0.04	0.94 ± 0.19	0.92 ± 0.19
Baseline	0.85 ± 0.17	0.75 ± 0.12	0.63 ± 0.08	0.83 ± 0.00	0.64 ± 0.09	0.59 ± 0.00	0.64 ± 0.02	0.53 ± 0.14	0.56 ± 0.19
S. Mendr	**0.92 ± 0.13**	**0.88 ± 0.08**	**0.84 ± 0.06**	**0.87 ± 0.06**	**0.80 ± 0.15**	**0.84 ± 0.14**	**0.74 ± 0.09**	**0.75 ± 0.02**	**0.70 ± 0.03**
Filtering	0.85 ± 0.14	0.78 ± 0.03	0.79 ± 0.01	0.83 ± 0.19	0.72 ± 0.09	0.66 ± 0.13	0.64 ± 0.18	0.65 ± 0.01	0.63 ± 0.05
Bagging	0.87 ± 0.07	0.82 ± 0.17	0.83 ± 0.06	0.82 ± 0.04	0.76 ± 0.08	0.81 ± 0.17	0.67 ± 0.03	0.72 ± 0.18	0.67 ± 0.19
	shuttle (low)			shuttle (moderate)			shuttle (high)		
Gold	0.99 ± 0.16	0.93 ± 0.03	0.94 ± 0.17	0.99 ± 0.03	0.93 ± 0.05	0.94 ± 0.09	0.99 ± 0.15	0.93 ± 0.01	0.94 ± 0.18
Baseline	0.82 ± 0.07	0.85 ± 0.05	0.82 ± 0.16	0.78 ± 0.14	0.67 ± 0.04	0.63 ± 0.15	0.72 ± 0.01	0.55 ± 0.17	0.55 ± 0.08
S. Mendr	**0.91 ± 0.11**	**0.92 ± 0.00**	**0.94 ± 0.01**	**0.85 ± 0.04**	**0.73 ± 0.07**	**0.81 ± 0.06**	**0.81 ± 0.04**	0.73 ± 0.06	**0.72 ± 0.09**
Filtering	0.84 ± 0.04	0.86 ± 0.07	0.88 ± 0.06	0.81 ± 0.11	0.69 ± 0.06	0.66 ± 0.03	0.77 ± 0.00	0.67 ± 0.05	0.60 ± 0.01
Bagging	0.88 ± 0.03	0.84 ± 0.13	0.92 ± 0.17	0.80 ± 0.04	0.71 ± 0.01	0.68 ± 0.02	0.75 ± 0.11	**0.75 ± 0.15**	0.62 ± 0.17
	statlog (low)			statlog (moderate)			statlog (high)		
Gold	0.97 ± 0.18	0.95 ± 0.19	0.89 ± 0.11	0.96 ± 0.13	0.95 ± 0.04	0.89 ± 0.14	0.97 ± 0.09	0.94 ± 0.13	0.89 ± 0.03
Baseline	0.85 ± 0.00	0.79 ± 0.09	0.74 ± 0.08	0.74 ± 0.09	0.65 ± 0.10	0.72 ± 0.02	0.61 ± 0.19	0.44 ± 0.12	0.52 ± 0.01
S. Mendr	**0.94 ± 0.17**	0.81 ± 0.16	**0.93 ± 0.14**	**0.90 ± 0.09**	**0.86 ± 0.12**	**0.85 ± 0.18**	**0.80 ± 0.04**	**0.79 ± 0.13**	**0.77 ± 0.06**
Filtering	0.88 ± 0.16	0.83 ± 0.04	0.88 ± 0.11	0.84 ± 0.11	0.75 ± 0.07	0.78 ± 0.11	0.65 ± 0.20	0.54 ± 0.18	0.67 ± 0.02
Bagging	0.92 ± 0.18	**0.89 ± 0.19**	0.81 ± 0.17	0.81 ± 0.14	0.83 ± 0.12	0.80 ± 0.12	0.78 ± 0.18	0.78 ± 0.13	0.72 ± 0.07
	twonorm (low)			twonorm (moderate)			twonorm (high)		
Gold	0.96 ± 0.04	0.94 ± 0.01	0.93 ± 0.05	0.96 ± 0.08	0.94 ± 0.14	0.93 ± 0.14	0.96 ± 0.15	0.94 ± 0.16	0.93 ± 0.01
Baseline	0.72 ± 0.11	0.78 ± 0.00	0.79 ± 0.00	0.6 ± 0.15	0.71 ± 0.15	0.6 ± 0.13	0.55 ± 0.18	0.63 ± 0.01	0.23 ± 0.02
S. Mendr	**0.87 ± 0.16**	**0.93 ± 0.05**	**0.90 ± 0.04**	**0.86 ± 0.03**	**0.83 ± 0.05**	**0.78 ± 0.01**	**0.73 ± 0.13**	**0.74 ± 0.11**	**0.74 ± 0.19**
Filtering	0.77 ± 0.01	0.84 ± 0.11	0.80 ± 0.01	0.76 ± 0.00	0.76 ± 0.16	0.67 ± 0.00	0.72 ± 0.11	0.66 ± 0.11	0.60 ± 0.13
Bagging	0.84 ± 0.12	0.83 ± 0.00	0.88 ± 0.09	0.77 ± 0.16	0.75 ± 0.19	0.76 ± 0.04	0.69 ± 0.08	0.73 ± 0.03	0.67 ± 0.00
	yeast (low)			yeast (moderate)			yeast (high)		
Gold	0.95 ± 0.00	0.94 ± 0.16	0.97 ± 0.09	0.95 ± 0.18	0.94 ± 0.09	0.97 ± 0.02	0.93 ± 0.18	0.97 ± 0.01	0.97 ± 0.15
Baseline	0.84 ± 0.00	0.83 ± 0.17	0.84 ± 0.00	0.72 ± 0.17	0.69 ± 0.05	0.66 ± 0.02	0.66 ± 0.05	0.40 ± 0.03	0.25 ± 0.12
S. Mendr	**0.90 ± 0.04**	**0.92 ± 0.16**	**0.95 ± 0.06**	**0.85 ± 0.19**	**0.78 ± 0.17**	**0.82 ± 0.19**	**0.70 ± 0.03**	**0.74 ± 0.02**	**0.73 ± 0.17**
Filtering	0.87 ± 0.06	0.85 ± 0.09	0.88 ± 0.00	0.76 ± 0.17	0.77 ± 0.19	0.74 ± 0.18	0.66 ± 0.01	0.66 ± 0.08	0.58 ± 0.02
Bagging	0.83 ± 0.14	0.87 ± 0.16	**0.95 ± 0.00**	0.73 ± 0.12	0.74 ± 0.02	0.77 ± 0.04	0.69 ± 0.09	0.71 ± 0.00	0.64 ± 0.06

Table B.3: MCC values for different techniques with different noise levels (I)

	SVM	KNN	Logit	SVM	KNN	Logit	SVM	KNN	Logit
	activity (low)			activity (moderate)			activity (high)		
Gold	0.87 ± 0.17	0.82 ± 0.06	0.87 ± 0.1	0.89 ± 0.08	0.85 ± 0.17	0.84 ± 0.03	0.87 ± 0.08	0.82 ± 0.05	0.83 ± 0.02
Baseline	0.78 ± 0.01	0.74 ± 0.07	0.64 ± 0.06	0.72 ± 0.09	0.62 ± 0.11	0.34 ± 0.06	0.45 ± 0.07	0.25 ± 0.04	0.17 ± 0.14
Smart Mendr	**0.87 ± 0.16**	**0.79 ± 0.08**	**0.84 ± 0.16**	**0.79 ± 0.01**	**0.66 ± 0.13**	**0.76 ± 0.09**	**0.63 ± 0.01**	**0.65 ± 0.01**	**0.63 ± 0.11**
Filtering	0.83 ± 0.12	0.75 ± 0.08	0.64 ± 0.11	0.77 ± 0.14	0.65 ± 0.01	0.62 ± 0.13	0.58 ± 0.00	0.36 ± 0.00	0.49 ± 0.11
Bagging	0.81 ± 0.02	0.75 ± 0.03	0.76 ± 0.08	0.71 ± 0.17	0.64 ± 0.04	0.70 ± 0.18	0.57 ± 0.17	0.61 ± 0.00	0.53 ± 0.11
	APS failure (low)			APS failure (moderate)			APS failure (high)		
Gold	0.98 ± 0.19	0.98 ± 0.19	0.97 ± 0.07	1.01 ± 0.19	0.95 ± 0.15	0.95 ± 0.11	1.01 ± 0.11	0.95 ± 0.01	0.94 ± 0.19
Baseline	0.72 ± 0.08	0.70 ± 0.07	0.60 ± 0.15	0.55 ± 0.02	0.54 ± 0.18	0.50 ± 0.08	0.53 ± 0.16	0.49 ± 0.18	0.43 ± 0.05
Smart Mendr	**0.91 ± 0.00**	**0.90 ± 0.06**	**0.86 ± 0.08**	**0.87 ± 0.08**	**0.77 ± 0.06**	**0.83 ± 0.17**	**0.74 ± 0.19**	**0.72 ± 0.17**	**0.73 ± 0.13**
Filtering	0.84 ± 0.03	0.75 ± 0.15	0.75 ± 0.14	0.83 ± 0.18	0.64 ± 0.15	0.69 ± 0.02	0.67 ± 0.01	0.61 ± 0.07	0.62 ± 0.11
Bagging	0.90 ± 0.03	0.74 ± 0.08	0.75 ± 0.19	0.80 ± 0.09	0.75 ± 0.06	0.68 ± 0.02	**0.74 ± 0.17**	0.66 ± 0.13	0.70 ± 0.17
	avila (low)			avila (moderate)			avila (high)		
Gold	0.94 ± 0.19	0.97 ± 0.08	0.93 ± 0.00	0.93 ± 0.06	0.97 ± 0.16	0.93 ± 0.01	0.94 ± 0.19	0.92 ± 0.01	0.95 ± 0.00
Baseline	0.89 ± 0.18	0.79 ± 0.05	0.80 ± 0.11	0.74 ± 0.02	0.69 ± 0.16	0.58 ± 0.03	0.32 ± 0.03	0.39 ± 0.07	0.11 ± 0.02
Smart Mendr	0.90 ± 0.09	**0.88 ± 0.02**	**0.93 ± 0.16**	**0.80 ± 0.00**	**0.80 ± 0.03**	**0.80 ± 0.01**	**0.79 ± 0.01**	**0.73 ± 0.05**	**0.76 ± 0.00**
Filtering	**0.91 ± 0.09**	0.84 ± 0.07	0.81 ± 0.04	0.73 ± 0.03	0.77 ± 0.02	0.69 ± 0.19	0.63 ± 0.05	0.65 ± 0.15	0.64 ± 0.13
Bagging	0.88 ± 0.00	0.84 ± 0.06	**0.93 ± 0.04**	0.77 ± 0.14	0.75 ± 0.00	0.71 ± 0.19	0.67 ± 0.16	0.72 ± 0.14	0.55 ± 0.02
	banana (low)			banana (moderate)			banana (high)		
Gold	0.90 ± 0.00	0.77 ± 0.00	0.87 ± 0.00	0.89 ± 0.12	0.78 ± 0.05	0.88 ± 0.00	0.87 ± 0.04	0.75 ± 0.13	0.87 ± 0.00
Baseline	0.70 ± 0.16	0.68 ± 0.15	0.60 ± 0.13	0.50 ± 0.08	0.42 ± 0.15	0.11 ± 0.07	0.44 ± 0.14	0.29 ± 0.00	0.20 ± 0.20
Smart Mendr	**0.87 ± 0.13**	**0.75 ± 0.06**	**0.83 ± 0.12**	**0.80 ± 0.04**	**0.65 ± 0.15**	**0.59 ± 0.13**	**0.70 ± 0.05**	**0.61 ± 0.16**	**0.59 ± 0.06**
Filtering	0.76 ± 0.00	0.71 ± 0.12	0.71 ± 0.08	0.77 ± 0.06	0.50 ± 0.19	0.47 ± 0.15	0.60 ± 0.14	0.49 ± 0.18	0.23 ± 0.05
Bagging	0.80 ± 0.12	0.67 ± 0.06	0.64 ± 0.11	0.73 ± 0.14	0.59 ± 0.14	0.57 ± 0.00	0.63 ± 0.16	0.58 ± 0.06	0.49 ± 0.07
	census (low)			census (moderate)			census (high)		
Gold	0.95 ± 0.15	0.93 ± 0.08	0.89 ± 0.03	0.95 ± 0.04	0.92 ± 0.14	0.91 ± 0.16	0.93 ± 0.14	0.91 ± 0.05	0.90 ± 0.01
Baseline	0.83 ± 0.18	0.81 ± 0.16	0.65 ± 0.18	0.54 ± 0.18	0.81 ± 0.08	0.55 ± 0.08	0.36 ± 0.06	0.62 ± 0.14	0.50 ± 0.15
Smart Mendr	**0.90 ± 0.17**	**0.87 ± 0.15**	**0.89 ± 0.19**	**0.84 ± 0.13**	**0.83 ± 0.02**	**0.86 ± 0.00**	**0.74 ± 0.09**	**0.81 ± 0.08**	**0.83 ± 0.00**
Filtering	0.83 ± 0.08	0.86 ± 0.00	0.70 ± 0.17	0.75 ± 0.08	**0.83 ± 0.08**	0.67 ± 0.14	0.65 ± 0.13	0.67 ± 0.12	0.56 ± 0.07
Bagging	0.82 ± 0.02	0.85 ± 0.15	0.85 ± 0.00	0.74 ± 0.11	0.79 ± 0.04	0.75 ± 0.20	0.67 ± 0.12	0.68 ± 0.06	0.61 ± 0.19
	connect4 (low)			connect4 (moderate)			connect4 (high)		
Gold	0.68 ± 0.14	0.59 ± 0.06	0.64 ± 0.01	0.68 ± 0.17	0.59 ± 0.19	0.64 ± 0.08	0.71 ± 0.00	0.57 ± 0.12	0.65 ± 0.11
Baseline	0.53 ± 0.17	0.44 ± 0.09	0.41 ± 0.05	0.38 ± 0.14	0.21 ± 0.11	0.39 ± 0.06	0.32 ± 0.16	0.17 ± 0.14	0.28 ± 0.01
Smart Mendr	**0.64 ± 0.11**	**0.54 ± 0.14**	**0.61 ± 0.03**	**0.59 ± 0.00**	**0.55 ± 0.18**	**0.59 ± 0.12**	**0.53 ± 0.00**	**0.56 ± 0.02**	**0.57 ± 0.00**
Filtering	0.59 ± 0.13	0.48 ± 0.08	0.50 ± 0.18	0.49 ± 0.04	0.38 ± 0.02	0.47 ± 0.06	0.35 ± 0.12	0.30 ± 0.00	0.30 ± 0.19
Bagging	0.53 ± 0.15	0.49 ± 0.17	0.53 ± 0.07	0.44 ± 0.00	0.39 ± 0.02	0.42 ± 0.00	0.32 ± 0.16	0.36 ± 0.04	0.33 ± 0.04
	german (low)			german (moderate)			german (high)		
Gold	0.94 ± 0.08	0.86 ± 0.14	0.9 ± 0.14	0.89 ± 0.2	0.91 ± 0.08	0.88 ± 0.14	0.89 ± 0.16	0.89 ± 0.08	0.9 ± 0.12
Baseline	0.77 ± 0.19	0.74 ± 0.13	0.67 ± 0.07	0.67 ± 0.13	0.66 ± 0.14	0.50 ± 0.05	0.53 ± 0.05	0.54 ± 0.00	0.37 ± 0.06
Smart Mendr	**0.86 ± 0.18**	0.77 ± 0.15	**0.85 ± 0.17**	**0.75 ± 0.14**	**0.76 ± 0.04**	**0.79 ± 0.19**	**0.77 ± 0.11**	**0.70 ± 0.11**	**0.71 ± 0.13**
Filtering	0.76 ± 0.06	0.77 ± 0.03	0.78 ± 0.00	0.72 ± 0.06	0.72 ± 0.07	0.62 ± 0.09	0.63 ± 0.06	0.66 ± 0.05	0.56 ± 0.17
Bagging	0.81 ± 0.03	**0.79 ± 0.18**	0.77 ± 0.12	0.71 ± 0.00	0.74 ± 0.05	0.73 ± 0.02	0.67 ± 0.05	0.61 ± 0.19	0.63 ± 0.17

Table B.4: MCC values for different techniques with different noise levels (II)

	SVM	KNN	Logit	SVM	KNN	Logit	SVM	KNN	Logit
	HTRU2 (low)			HTRU2 (moderate)			HTRU2 (high)		
Gold	0.96 ± 0.09	0.93 ± 0.16	0.97 ± 0.2	0.98 ± 0.12	0.96 ± 0.02	0.94 ± 0.15	0.96 ± 0.03	0.95 ± 0.15	0.95 ± 0.03
Baseline	0.42 ± 0.05	0.28 ± 0.14	0.23 ± 0.04	0.34 ± 0.00	0.20 ± 0.00	0.02 ± 0.14	0.19 ± 0.06	0.07 ± 0.14	0.12 ± 0.14
Smart Mendr	**0.87 ± 0.11**	**0.81 ± 0.01**	**0.83 ± 0.18**	**0.84 ± 0.01**	**0.78 ± 0.09**	**0.71 ± 0.19**	**0.72 ± 0.00**	**0.70 ± 0.00**	**0.67 ± 0.14**
Filtering	0.72 ± 0.01	0.79 ± 0.05	0.74 ± 0.09	0.62 ± 0.11	0.71 ± 0.11	0.70 ± 0.01	0.54 ± 0.11	0.63 ± 0.07	0.52 ± 0.06
Bagging	0.76 ± 0.05	0.59 ± 0.14	0.66 ± 0.14	0.69 ± 0.13	0.54 ± 0.01	0.56 ± 0.00	0.58 ± 0.04	0.44 ± 0.03	0.51 ± 0.18
	MoCap (low)			MoCap (moderate)			MoCap (high)		
Gold	0.94 ± 0.19	0.95 ± 0.06	0.99 ± 0.00	0.96 ± 0.03	0.95 ± 0.13	0.99 ± 0.04	0.96 ± 0.18	0.95 ± 0.02	0.99 ± 0.19
Baseline	0.83 ± 0.11	0.81 ± 0.11	0.65 ± 0.08	0.65 ± 0.00	0.77 ± 0.14	0.60 ± 0.02	0.56 ± 0.05	0.63 ± 0.11	0.48 ± 0.00
Smart Mendr	**0.92 ± 0.13**	**0.91 ± 0.06**	0.85 ± 0.04	**0.80 ± 0.07**	**0.84 ± 0.11**	0.72 ± 0.11	**0.70 ± 0.18**	**0.69 ± 0.00**	**0.73 ± 0.00**
Filtering	0.81 ± 0.12	0.84 ± 0.09	0.77 ± 0.13	0.75 ± 0.00	0.78 ± 0.02	0.70 ± 0.09	0.60 ± 0.07	0.67 ± 0.12	0.60 ± 0.01
Bagging	0.85 ± 0.03	0.88 ± 0.11	**0.98 ± 0.07**	0.67 ± 0.07	0.77 ± 0.00	**0.73 ± 0.16**	0.67 ± 0.05	0.65 ± 0.19	0.54 ± 0.17
	penbased (low)			penbased (moderate)			penbased (high)		
Gold	0.90 ± 0.11	0.85 ± 0.09	0.91 ± 0.12	0.95 ± 0.01	0.87 ± 0.05	0.93 ± 0.12	0.90 ± 0.19	0.84 ± 0.07	0.93 ± 0.01
Baseline	0.77 ± 0.05	0.78 ± 0.07	0.69 ± 0.00	0.74 ± 0.02	0.60 ± 0.02	0.53 ± 0.09	0.61 ± 0.03	0.36 ± 0.19	0.24 ± 0.16
Smart Mendr	**0.90 ± 0.05**	**0.81 ± 0.13**	**0.89 ± 0.00**	**0.83 ± 0.19**	**0.72 ± 0.18**	**0.68 ± 0.08**	**0.73 ± 0.01**	**0.68 ± 0.12**	**0.66 ± 0.15**
Filtering	0.81 ± 0.18	0.79 ± 0.19	0.69 ± 0.18	0.76 ± 0.11	0.66 ± 0.12	0.57 ± 0.00	0.64 ± 0.05	0.48 ± 0.01	0.53 ± 0.17
Bagging	0.83 ± 0.13	0.80 ± 0.05	0.79 ± 0.08	0.74 ± 0.09	0.68 ± 0.14	**0.68 ± 0.00**	0.62 ± 0.17	0.66 ± 0.09	0.5 ± 0.00
	shoppers intention (low)			shoppers intention (moderate)			shoppers intention (high)		
Gold	0.90 ± 0.13	0.76 ± 0.18	0.87 ± 0.11	0.90 ± 0.07	0.79 ± 0.13	0.86 ± 0.09	0.90 ± 0.12	0.76 ± 0.18	0.87 ± 0.07
Baseline	0.72 ± 0.00	0.68 ± 0.14	0.60 ± 0.00	0.49 ± 0.03	0.44 ± 0.12	0.10 ± 0.07	0.42 ± 0.18	0.30 ± 0.04	0.20 ± 0.02
Smart Mendr	**0.84 ± 0.09**	**0.74 ± 0.16**	**0.86 ± 0.20**	**0.80 ± 0.07**	**0.65 ± 0.16**	**0.59 ± 0.07**	**0.67 ± 0.07**	**0.62 ± 0.17**	**0.59 ± 0.11**
Filtering	0.75 ± 0.00	0.70 ± 0.18	0.74 ± 0.04	0.74 ± 0.12	0.49 ± 0.08	0.46 ± 0.18	0.60 ± 0.03	0.46 ± 0.14	0.23 ± 0.00
Bagging	0.77 ± 0.19	0.69 ± 0.00	0.63 ± 0.01	0.73 ± 0.00	0.56 ± 0.09	0.57 ± 0.15	0.62 ± 0.09	0.58 ± 0.08	0.48 ± 0.14
	shuttle (low)			shuttle (moderate)			shuttle (high)		
Gold	0.95 ± 0.15	0.89 ± 0.17	0.91 ± 0.11	0.96 ± 0.18	0.90 ± 0.09	0.90 ± 0.00	0.92 ± 0.01	0.86 ± 0.00	0.86 ± 0.12
Baseline	0.79 ± 0.09	0.77 ± 0.07	0.77 ± 0.04	0.73 ± 0.17	0.68 ± 0.04	0.60 ± 0.14	0.65 ± 0.12	0.51 ± 0.07	0.49 ± 0.00
Smart Mendr	**0.91 ± 0.18**	**0.84 ± 0.16**	0.88 ± 0.17	0.75 ± 0.15	0.70 ± 0.15	**0.75 ± 0.08**	**0.77 ± 0.17**	**0.67 ± 0.00**	**0.68 ± 0.11**
Filtering	0.79 ± 0.04	0.78 ± 0.20	0.82 ± 0.02	0.79 ± 0.00	**0.71 ± 0.13**	0.63 ± 0.14	0.71 ± 0.00	0.64 ± 0.00	0.58 ± 0.19
Bagging	0.85 ± 0.12	0.79 ± 0.00	**0.89 ± 0.07**	**0.80 ± 0.19**	0.69 ± 0.11	0.66 ± 0.07	0.68 ± 0.00	0.66 ± 0.09	0.58 ± 0.08
	statlog (low)			statlog (moderate)			statlog (high)		
Gold	0.95 ± 0.02	0.94 ± 0.16	0.88 ± 0.04	0.93 ± 0.13	0.95 ± 0.19	0.88 ± 0.05	0.93 ± 0.00	0.96 ± 0.17	0.90 ± 0.15
Baseline	0.76 ± 0.13	0.76 ± 0.09	0.70 ± 0.01	0.69 ± 0.06	0.60 ± 0.16	0.66 ± 0.18	0.58 ± 0.06	0.42 ± 0.12	0.49 ± 0.19
Smart Mendr	**0.87 ± 0.15**	**0.82 ± 0.14**	**0.86 ± 0.18**	**0.84 ± 0.17**	**0.78 ± 0.08**	**0.79 ± 0.16**	**0.76 ± 0.09**	0.72 ± 0.14	**0.74 ± 0.17**
Filtering	0.84 ± 0.14	0.78 ± 0.03	0.81 ± 0.03	0.80 ± 0.14	0.69 ± 0.06	0.72 ± 0.09	0.63 ± 0.19	0.50 ± 0.05	0.63 ± 0.05
Bagging	0.86 ± 0.00	**0.82 ± 0.14**	0.77 ± 0.09	0.75 ± 0.03	0.77 ± 0.06	0.76 ± 0.04	0.72 ± 0.06	**0.73 ± 0.06**	0.68 ± 0.08
	twonorm (low)			twonorm (moderate)			twonorm (high)		
Gold	0.93 ± 0.18	0.94 ± 0.04	0.91 ± 0.05	0.97 ± 0.06	0.93 ± 0.05	0.94 ± 0.13	0.92 ± 0.02	0.95 ± 0.13	0.91 ± 0.04
Baseline	0.72 ± 0.13	0.77 ± 0.08	0.79 ± 0.04	0.60 ± 0.00	0.68 ± 0.18	0.60 ± 0.01	0.53 ± 0.09	0.61 ± 0.07	0.22 ± 0.16
Smart Mendr	**0.85 ± 0.00**	**0.94 ± 0.16**	**0.87 ± 0.00**	**0.87 ± 0.05**	**0.76 ± 0.02**	**0.78 ± 0.05**	**0.73 ± 0.09**	**0.73 ± 0.09**	**0.74 ± 0.00**
Filtering	0.81 ± 0.04	0.85 ± 0.05	0.78 ± 0.07	0.77 ± 0.16	0.84 ± 0.18	0.67 ± 0.04	0.72 ± 0.05	0.64 ± 0.15	0.59 ± 0.11
Bagging	0.81 ± 0.12	0.81 ± 0.06	0.86 ± 0.04	0.77 ± 0.19	**0.76 ± 0.15**	0.76 ± 0.16	0.67 ± 0.13	0.71 ± 0.02	0.67 ± 0.02
	yeast (low)			yeast (moderate)			yeast (high)		
Gold	0.96 ± 0.09	0.95 ± 0.03	0.98 ± 0.02	0.92 ± 0.18	0.94 ± 0.19	0.95 ± 0.15	0.96 ± 0.11	0.98 ± 0.09	0.97 ± 0.08
Baseline	0.83 ± 0.05	0.82 ± 0.19	0.83 ± 0.07	0.70 ± 0.15	0.68 ± 0.03	0.67 ± 0.02	0.64 ± 0.08	0.4 ± 0.02	0.24 ± 0.07
Smart Mendr	**0.90 ± 0.18**	**0.95 ± 0.00**	**0.93 ± 0.03**	**0.84 ± 0.15**	**0.81 ± 0.05**	**0.80 ± 0.14**	**0.70 ± 0.14**	**0.74 ± 0.08**	**0.71 ± 0.04**
Filtering	0.87 ± 0.08	0.83 ± 0.01	0.86 ± 0.00	0.78 ± 0.14	0.75 ± 0.14	0.74 ± 0.07	0.64 ± 0.09	0.69 ± 0.08	0.58 ± 0.06
Bagging	0.85 ± 0.04	0.87 ± 0.09	0.92 ± 0.05	0.75 ± 0.13	0.71 ± 0.15	0.74 ± 0.01	0.69 ± 0.15	0.71 ± 0.00	0.65 ± 0.14

Table B.5: Accuracy values for different approaches for different levels of incomplete supervision (I)

	SVM	KNN	Logit	SVM	KNN	Logit	SVM	KNN	Logit
	activity (easy)			activity (medium)			activity (hard)		
Gold	0.82 ± 0.16	0.78 ± 0.17	0.80 ± 0.17	0.79 ± 0.00	0.74 ± 0.11	0.81 ± 0.04	0.80 ± 0.05	0.79 ± 0.04	0.76 ± 0.00
Baseline	0.76 ± 0.11	0.64 ± 0.05	0.65 ± 0.14	0.72 ± 0.01	0.47 ± 0.06	0.48 ± 0.00	0.42 ± 0.04	0.14 ± 0.00	0.35 ± 0.17
S. Mendr	**0.75 ± 0.08**	**0.73 ± 0.01**	**0.80 ± 0.19**	**0.75 ± 0.06**	**0.72 ± 0.18**	**0.70 ± 0.19**	**0.68 ± 0.13**	**0.65 ± 0.07**	**0.63 ± 0.03**
SSL	0.74 ± 0.04	0.69 ± 0.00	0.65 ± 0.18	0.70 ± 0.00	0.66 ± 0.16	0.61 ± 0.02	0.52 ± 0.08	0.53 ± 0.00	0.59 ± 0.06
	APS failure (easy)			APS failure (medium)			APS failure (hard)		
Gold	0.91 ± 0.05	0.90 ± 0.04	0.87 ± 0.03	0.94 ± 0.09	0.88 ± 0.00	0.86 ± 0.08	0.93 ± 0.18	0.88 ± 0.13	0.87 ± 0.15
Baseline	0.67 ± 0.00	0.74 ± 0.15	0.63 ± 0.14	0.49 ± 0.07	0.62 ± 0.17	0.60 ± 0.13	0.48 ± 0.11	0.22 ± 0.09	0.46 ± 0.15
S. Mendr	**0.92 ± 0.04**	**0.86 ± 0.19**	**0.86 ± 0.18**	0.81 ± 0.09	**0.82 ± 0.12**	**0.77 ± 0.00**	**0.81 ± 0.15**	**0.78 ± 0.07**	**0.73 ± 0.17**
SSL	0.86 ± 0.07	0.83 ± 0.05	0.84 ± 0.11	**0.84 ± 0.05**	0.77 ± 0.14	0.75 ± 0.11	0.69 ± 0.17	0.61 ± 0.06	0.67 ± 0.00
	avila (easy)			avila (medium)			avila (hard)		
Gold	0.96 ± 0.13	0.92 ± 0.03	0.94 ± 0.12	0.89 ± 0.00	0.95 ± 0.11	0.94 ± 0.19	0.97 ± 0.06	0.92 ± 0.13	0.96 ± 0.01
Baseline	0.88 ± 0.14	0.85 ± 0.09	0.84 ± 0.13	0.84 ± 0.06	0.81 ± 0.02	0.77 ± 0.02	0.73 ± 0.15	0.72 ± 0.00	0.61 ± 0.00
S. Mendr	**0.92 ± 0.14**	**0.89 ± 0.09**	**0.92 ± 0.03**	**0.88 ± 0.00**	**0.87 ± 0.03**	**0.88 ± 0.00**	**0.81 ± 0.01**	**0.84 ± 0.06**	**0.81 ± 0.15**
SSL	0.89 ± 0.19	0.83 ± 0.00	0.91 ± 0.03	0.81 ± 0.01	0.85 ± 0.16	0.81 ± 0.04	0.72 ± 0.16	0.67 ± 0.04	0.72 ± 0.17
	banana (easy)			banana (medium)			banana (hard)		
Gold	0.91 ± 0.16	0.78 ± 0.09	0.86 ± 0.12	0.90 ± 0.09	0.78 ± 0.12	0.86 ± 0.11	0.90 ± 0.13	0.78 ± 0.13	0.85 ± 0.00
Baseline	0.77 ± 0.01	0.68 ± 0.11	0.62 ± 0.02	0.49 ± 0.02	0.67 ± 0.01	0.50 ± 0.13	0.44 ± 0.16	0.53 ± 0.11	0.30 ± 0.13
S. Mendr	**0.88 ± 0.14**	**0.76 ± 0.08**	**0.82 ± 0.14**	**0.84 ± 0.00**	**0.72 ± 0.06**	**0.81 ± 0.02**	**0.76 ± 0.17**	**0.66 ± 0.13**	**0.76 ± 0.00**
SSL	0.84 ± 0.15	0.70 ± 0.14	0.77 ± 0.02	0.72 ± 0.06	0.68 ± 0.16	0.71 ± 0.04	0.63 ± 0.18	0.56 ± 0.01	0.62 ± 0.08
	census (easy)			census (easy)			census (easy)		
Gold	0.82 ± 0.01	0.82 ± 0.03	0.83 ± 0.11	0.82 ± 0.06	0.81 ± 0.11	0.81 ± 0.03	0.86 ± 0.12	0.85 ± 0.1	0.83 ± 0.18
Baseline	0.75 ± 0.13	0.75 ± 0.17	0.76 ± 0.17	0.75 ± 0.15	0.66 ± 0.11	0.71 ± 0.16	0.59 ± 0.12	0.38 ± 0.19	0.60 ± 0.07
S. Mendr	**0.82 ± 0.18**	**0.86 ± 0.06**	**0.84 ± 0.19**	**0.76 ± 0.00**	**0.75 ± 0.17**	0.73 ± 0.02	**0.73 ± 0.10**	**0.82 ± 0.04**	**0.71 ± 0.16**
SSL	0.76 ± 0.12	0.73 ± 0.09	0.82 ± 0.13	0.72 ± 0.11	0.71 ± 0.13	**0.74 ± 0.12**	0.65 ± 0.11	0.69 ± 0.15	0.60 ± 0.17
	connect4 (easy)			connect4 (medium)			connect4 (hard)		
Gold	0.63 ± 0.00	0.63 ± 0.04	0.59 ± 0.11	0.63 ± 0.11	0.62 ± 0.11	0.59 ± 0.03	0.66 ± 0.08	0.65 ± 0.07	0.57 ± 0.18
Baseline	0.53 ± 0.09	0.49 ± 0.01	0.38 ± 0.19	0.49 ± 0.06	0.46 ± 0.18	0.27 ± 0.14	0.22 ± 0.07	0.09 ± 0.19	0.07 ± 0.00
S. Mendr	**0.61 ± 0.08**	**0.63 ± 0.18**	**0.54 ± 0.19**	**0.58 ± 0.02**	**0.57 ± 0.16**	**0.51 ± 0.03**	**0.54 ± 0.04**	**0.40 ± 0.13**	**0.46 ± 0.17**
SSL	0.59 ± 0.08	0.50 ± 0.15	0.53 ± 0.03	0.50 ± 0.02	0.50 ± 0.07	0.43 ± 0.12	0.38 ± 0.19	0.28 ± 0.09	0.34 ± 0.18
	german (easy)			german (medium)			german (hard)		
Gold	0.92 ± 0.00	0.86 ± 0.03	0.93 ± 0.19	0.91 ± 0.11	0.84 ± 0.08	0.90 ± 0.08	0.94 ± 0.13	0.85 ± 0.05	0.86 ± 0.02
Baseline	0.74 ± 0.02	0.77 ± 0.11	0.73 ± 0.17	0.66 ± 0.01	0.69 ± 0.14	0.65 ± 0.08	0.53 ± 0.04	0.42 ± 0.04	0.61 ± 0.12
S. Mendr	**0.86 ± 0.13**	0.79 ± 0.06	**0.89 ± 0.04**	**0.81 ± 0.03**	**0.80 ± 0.11**	**0.79 ± 0.16**	**0.77 ± 0.03**	**0.79 ± 0.08**	**0.76 ± 0.12**
SSL	0.79 ± 0.03	**0.84 ± 0.00**	0.80 ± 0.17	0.72 ± 0.17	0.72 ± 0.01	**0.79 ± 0.02**	0.70 ± 0.12	0.72 ± 0.02	0.67 ± 0.15

Table B.6: Accuracy values for different approaches for different levels of incomplete supervision (II)

	SVM	KNN	Logit	SVM	KNN	Logit	SVM	KNN	Logit
	HTRU2 (easy)			HTRU2 (medium)			HTRU2 (hard)		
Gold	0.92 ± 0.02	0.90 ± 0.06	0.85 ± 000	0.89 ± 0.15	0.86 ± 0.13	0.92 ± 0.01	0.93 ± 0.00	0.84 ± 0.15	0.88 ± 0.11
Baseline	0.74 ± 0.16	0.80 ± 0.05	0.75 ± 0.09	0.70 ± 0.09	0.68 ± 0.04	0.67 ± 0.04	0.39 ± 0.15	0.36 ± 0.13	0.30 ± 0.00
S. Mendr	**0.85 ± 0.12**	**0.88 ± 0.12**	**0.79 ± 0.16**	**0.76 ± 0.13**	**0.82 ± 0.11**	**0.83 ± 0.18**	**0.76 ± 0.00**	**0.75 ± 0.17**	**0.85 ± 0.01**
SSL	0.71 ± 0.00	0.77 ± 0.14	0.76 ± 0.00	0.73 ± 0.13	0.76 ± 0.08	0.81 ± 0.15	0.53 ± 0.11	0.63 ± 0.02	0.69 ± 0.02
	MoCap (easy)			MoCap (medium)			MoCap (hard)		
Gold	0.86 ± 0.07	0.85 ± 0.03	0.89 ± 0.04	0.90 ± 0.18	0.85 ± 0.13	0.87 ± 0.13	0.89 ± 0.11	0.87 ± 0.04	0.90 ± 0.03
Baseline	0.75 ± 0.13	0.73 ± 0.15	0.74 ± 0.12	0.72 ± 0.14	0.65 ± 0.01	0.73 ± 0.06	0.61 ± 0.04	0.21 ± 0.14	0.49 ± 0.16
S. Mendr	**0.86 ± 0.15**	**0.85 ± 0.16**	**0.88 ± 0.09**	**0.81 ± 0.00**	**0.77 ± 0.00**	0.80 ± 0.09	**0.78 ± 0.05**	**0.73 ± 0.00**	**0.78 ± 0.03**
SSL	0.81 ± 0.16	0.76 ± 0.06	0.84 ± 0.00	0.78 ± 0.18	0.74 ± 0.07	**0.81 ± 0.05**	0.64 ± 0.07	0.67 ± 0.08	0.68 ± 0.08
	penbased (easy)			penbased (medium)			penbased (hard)		
Gold	0.92 ± 0.09	0.85 ± 0.17	0.89 ± 0.03	0.93 ± 0.11	0.86 ± 0.02	0.90 ± 0.05	0.93 ± 0.19	0.84 ± 0.06	0.88 ± 0.15
Baseline	0.86 ± 0.13	0.76 ± 0.00	0.67 ± 0.05	0.83 ± 0.14	0.57 ± 0.02	0.45 ± 0.15	0.72 ± 0.13	0.31 ± 0.17	0.33 ± 0.13
S. Mendr	**0.93 ± 0.00**	**0.81 ± 0.16**	**0.87 ± 0.13**	**0.86 ± 0.00**	**0.78 ± 0.00**	**0.84 ± 0.01**	**0.81 ± 0.02**	**0.74 ± 0.12**	**0.78 ± 0.19**
SSL	0.85 ± 0.07	0.77 ± 0.17	0.86 ± 0.13	0.84 ± 0.11	0.76 ± 0.16	0.72 ± 0.16	0.75 ± 0.03	0.60 ± 0.00	0.63 ± 0.08
	shoppers intention (easy)			shoppers intention (medium)			shoppers intention (hard)		
Gold	0.90 ± 0.04	0.86 ± 0.07	0.85 ± 0.20	0.89 ± 0.12	0.85 ± 0.11	0.85 ± 0.06	0.90 ± 0.01	0.86 ± 0.03	0.87 ± 0.17
Baseline	0.76 ± 0.14	0.74 ± 0.19	0.79 ± 0.18	0.76 ± 0.13	0.62 ± 0.19	0.70 ± 0.01	0.57 ± 0.18	0.16 ± 0.08	0.31 ± 0.09
S. Mendr	**0.87 ± 0.01**	**0.84 ± 0.17**	**0.85 ± 0.01**	**0.83 ± 0.11**	**0.77 ± 0.14**	**0.82 ± 0.15**	**0.77 ± 0.04**	**0.75 ± 0.19**	**0.69 ± 0.14**
SSL	0.84 ± 0.05	0.79 ± 0.08	0.80 ± 0.05	0.81 ± 0.19	0.68 ± 0.11	0.75 ± 0.05	0.61 ± 0.01	0.59 ± 0.19	0.57 ± 0.17
	shuttle (easy)			shuttle (medium)			shuttle (hard)		
Gold	0.94 ± 0.18	0.90 ± 0.15	0.91 ± 0.06	0.89 ± 0.01	0.85 ± 0.16	0.84 ± 0.15	0.88 ± 0.01	0.83 ± 0.07	0.90 ± 0.13
Baseline	0.80 ± 0.00	0.79 ± 0.11	0.72 ± 0.00	0.78 ± 0.01	0.76 ± 0.00	0.52 ± 0.11	0.74 ± 0.03	0.37 ± 0.14	0.29 ± 0.15
S. Mendr	**0.87 ± 0.16**	**0.86 ± 0.15**	0.80 ± 0.19	**0.82 ± 0.17**	**0.77 ± 0.09**	**0.84 ± 0.06**	**0.78 ± 0.17**	**0.70 ± 0.07**	**0.77 ± 0.00**
SSL	0.83 ± 0.07	0.80 ± 0.16	**0.82 ± 0.03**	0.80 ± 0.07	0.76 ± 0.01	0.75 ± 0.08	0.77 ± 0.12	0.68 ± 0.17	0.60 ± 0.02
	statlog (easy)			statlog (medium)			statlog (hard)		
Gold	0.92 ± 0.13	0.91 ± 0.09	0.86 ± 0.03	0.97 ± 0.00	0.96 ± 0.17	0.86 ± 0.02	0.96 ± 0.05	0.91 ± 0.02	0.85 ± 0.07
Baseline	0.79 ± 0.12	0.77 ± 0.07	0.60 ± 0.18	0.64 ± 0.06	0.79 ± 0.18	0.46 ± 0.05	0.56 ± 0.01	0.52 ± 0.00	0.16 ± 0.19
S. Mendr	**0.88 ± 0.03**	**0.86 ± 0.17**	**0.88 ± 0.08**	0.83 ± 0.17	**0.82 ± 0.05**	**0.80 ± 0.12**	**0.82 ± 0.11**	**0.80 ± 0.18**	**0.78 ± 0.13**
SSL	0.86 ± 0.08	0.84 ± 0.12	0.76 ± 0.02	**0.86 ± 0.00**	0.78 ± 0.11	0.71 ± 0.07	0.65 ± 0.12	0.67 ± 0.03	0.62 ± 0.04
	twonorm (easy)			twonorm (medium)			twonorm (hard)		
Gold	0.97 ± 0.02	0.87 ± 0.14	0.89 ± 0.16	0.92 ± 0.11	0.94 ± 0.18	0.92 ± 0.02	0.95 ± 0.06	0.95 ± 0.05	0.93 ± 0.02
Baseline	0.83 ± 0.02	0.75 ± 0.17	0.73 ± 0.11	0.65 ± 0.06	0.78 ± 0.11	0.53 ± 0.09	0.5 ± 0.07	0.73 ± 0.17	0.35 ± 0.05
S. Mendr	**0.93 ± 0.01**	**0.91 ± 0.06**	**0.88 ± 0.01**	**0.84 ± 0.07**	**0.84 ± 0.04**	**0.86 ± 0.03**	**0.86 ± 0.18**	**0.73 ± 0.11**	**0.74 ± 0.05**
SSL	0.91 ± 0.07	0.80 ± 0.05	0.82 ± 0.04	0.78 ± 0.02	0.74 ± 0.09	0.76 ± 0.08	0.75 ± 0.05	0.71 ± 0.2	0.71 ± 0.19
	yeast (easy)			yeast (medium)			yeast (hard)		
Gold	0.90 ± 0.02	0.84 ± 0.14	0.89 ± 0.15	0.86 ± 0.03	0.91 ± 0.18	0.92 ± 0.06	0.85 ± 0.06	0.87 ± 0.02	0.93 ± 0.06
Baseline	0.79 ± 0.08	0.77 ± 0.06	0.76 ± 0.19	0.70 ± 0.07	0.59 ± 0.00	0.64 ± 0.06	0.48 ± 0.00	0.46 ± 0.14	0.61 ± 0.16
S. Mendr	**0.87 ± 0.09**	**0.82 ± 0.12**	**0.90 ± 0.14**	**0.77 ± 0.05**	**0.77 ± 0.04**	**0.79 ± 0.09**	**0.79 ± 0.03**	**0.72 ± 0.06**	**0.78 ± 0.02**
SSL	0.81 ± 0.19	0.80 ± 0.03	0.81 ± 0.01	0.73 ± 0.00	0.67 ± 0.17	0.76 ± 0.16	0.60 ± 0.15	0.68 ± 0.12	0.63 ± 0.11

Table B.7: MCC values for different approaches for different levels of incomplete supervision (I)

	SVM	KNN	Logit	SVM	KNN	Logit	SVM	KNN	Logit
	activity (easy)			activity (medium)			activity (hard)		
Gold	0.84 ± 0.15	0.74 ± 0.11	0.79 ± 0.11	0.83 ± 0.14	0.79 ± 0.17	0.8 ± 0.12	0.81 ± 0.02	0.75 ± 0.12	0.78 ± 0.02
Baseline	0.77 ± 0.03	0.63 ± 0.08	0.64 ± 0.00	0.72 ± 0.07	0.50 ± 0.00	0.46 ± 0.00	0.41 ± 0.01	0.15 ± 0.03	0.35 ± 0.19
S. Mendr	**0.79 ± 0.13**	**0.77 ± 0.15**	**0.79 ± 0.15**	0.72 ± 0.00	**0.68 ± 0.12**	**0.65 ± 0.07**	**0.65 ± 0.18**	**0.68 ± 0.11**	**0.61 ± 0.16**
SSL	0.78 ± 0.14	0.73 ± 0.01	0.70 ± 0.07	**0.73 ± 0.02**	0.62 ± 0.00	0.60 ± 0.06	0.52 ± 0.03	0.52 ± 0.19	0.56 ± 0.00
	APS failure (easy)			APS failure (medium)			APS failure (hard)		
Gold	0.93 ± 0.19	0.91 ± 0.17	0.90 ± 0.11	0.96 ± 0.18	0.86 ± 0.11	0.91 ± 0.16	0.95 ± 0.08	0.86 ± 0.11	0.90 ± 0.12
Baseline	0.66 ± 0.16	0.72 ± 0.06	0.60 ± 0.02	0.49 ± 0.18	0.62 ± 0.12	0.64 ± 0.13	0.50 ± 0.18	0.23 ± 0.14	0.46 ± 0.07
S. Mendr	**0.91 ± 0.16**	**0.88 ± 0.14**	**0.84 ± 0.14**	**0.82 ± 0.07**	**0.80 ± 0.17**	**0.83 ± 0.07**	**0.72 ± 0.00**	**0.78 ± 0.08**	**0.77 ± 0.11**
SSL	0.88 ± 0.14	0.84 ± 0.05	0.79 ± 0.07	0.79 ± 0.15	0.72 ± 0.07	0.76 ± 0.04	0.67 ± 0.02	0.62 ± 0.00	0.65 ± 0.18
	avila (easy)			avila (medium)			avila (hard)		
Gold	0.92 ± 0.03	0.93 ± 0.04	0.94 ± 0.17	0.94 ± 0.00	0.95 ± 0.02	0.96 ± 0.09	0.89 ± 0.05	0.96 ± 0.11	0.9 ± 0.11
Baseline	0.87 ± 0.15	0.85 ± 0.14	0.79 ± 0.11	0.83 ± 0.02	0.79 ± 0.15	0.76 ± 0.05	0.73 ± 0.15	0.69 ± 0.04	0.61 ± 0.12
S. Mendr	**0.92 ± 0.14**	**0.90 ± 0.08**	0.89 ± 0.14	**0.86 ± 0.19**	0.83 ± 0.18	**0.84 ± 0.19**	**0.81 ± 0.09**	**0.82 ± 0.03**	**0.83 ± 0.15**
SSL	0.86 ± 0.18	0.87 ± 0.06	**0.93 ± 0.09**	0.80 ± 0.01	**0.84 ± 0.15**	0.79 ± 0.07	0.76 ± 0.08	0.69 ± 0.02	0.74 ± 0.07
	banana (easy)			banana (medium)			banana (hard)		
Gold	0.85 ± 0.07	0.75 ± 0.05	0.84 ± 0.06	0.88 ± 0.07	0.78 ± 0.11	0.87 ± 0.05	0.88 ± 0.01	0.78 ± 0.12	0.88 ± 0.14
Baseline	0.76 ± 0.18	0.65 ± 0.04	0.64 ± 0.09	0.46 ± 0.00	0.63 ± 0.03	0.51 ± 0.17	0.41 ± 0.14	0.52 ± 0.00	0.30 ± 0.14
S. Mendr	**0.89 ± 0.08**	**0.73 ± 0.06**	**0.82 ± 0.12**	**0.86 ± 0.01**	**0.72 ± 0.02**	**0.76 ± 0.02**	**0.73 ± 0.01**	**0.67 ± 0.16**	**0.73 ± 0.12**
SSL	0.87 ± 0.07	0.67 ± 0.11	0.78 ± 0.06	0.73 ± 0.17	0.65 ± 0.13	0.69 ± 0.19	0.65 ± 0.12	0.56 ± 0.01	0.59 ± 0.08
	census (easy)			census (easy)			census (easy)		
Gold	0.88 ± 0.17	0.83 ± 0.18	0.78 ± 0.06	0.85 ± 0.03	0.86 ± 0.15	0.81 ± 0.06	0.84 ± 0.10	0.85 ± 0.13	0.83 ± 0.19
Baseline	0.78 ± 0.08	0.75 ± 0.07	0.73 ± 0.11	0.72 ± 0.17	0.70 ± 0.06	0.70 ± 0.04	0.60 ± 0.02	0.42 ± 0.16	0.59 ± 0.10
S. Mendr	**0.84 ± 0.15**	**0.81 ± 0.18**	**0.76 ± 0.09**	**0.74 ± 0.14**	**0.76 ± 0.00**	**0.77 ± 0.12**	**0.73 ± 0.02**	**0.74 ± 0.15**	**0.71 ± 0.11**
SSL	0.81 ± 0.08	0.73 ± 0.16	0.73 ± 0.11	0.70 ± 0.19	0.73 ± 0.02	0.73 ± 0.13	0.68 ± 0.11	0.65 ± 0.01	0.63 ± 0.11
	connect4 (easy)			connect4 (medium)			connect4 (hard)		
Gold	0.64 ± 0.12	0.61 ± 0.09	0.60 ± 0.01	0.64 ± 0.14	0.65 ± 0.06	0.6 ± 0.06	0.64 ± 0.00	0.64 ± 0.18	0.59 ± 0.12
Baseline	0.51 ± 0.17	0.50 ± 0.12	0.39 ± 0.16	0.47 ± 0.12	0.46 ± 0.12	0.26 ± 0.14	0.22 ± 0.05	0.09 ± 0.08	0.07 ± 0.06
S. Mendr	**0.59 ± 0.18**	**0.60 ± 0.09**	**0.54 ± 0.02**	**0.59 ± 0.00**	**0.58 ± 0.00**	**0.59 ± 0.09**	**0.55 ± 0.19**	**0.61 ± 0.01**	**0.52 ± 0.04**
SSL	0.57 ± 0.17	0.52 ± 0.13	0.53 ± 0.19	0.47 ± 0.04	0.48 ± 0.01	0.45 ± 0.00	0.36 ± 0.00	0.28 ± 0.07	0.34 ± 0.08
	german (easy)			german (medium)			german (hard)		
Gold	0.91 ± 0.11	0.86 ± 0.02	0.91 ± 0.16	0.93 ± 0.11	0.87 ± 0.1	0.88 ± 0.11	0.93 ± 0.12	0.87 ± 0.00	0.86 ± 0.11
Baseline	0.73 ± 0.12	0.77 ± 0.00	0.68 ± 0.13	0.66 ± 0.13	0.68 ± 0.15	0.67 ± 0.19	0.51 ± 0.06	0.45 ± 0.14	0.59 ± 0.15
S. Mendr	**0.89 ± 0.14**	0.80 ± 0.02	**0.89 ± 0.18**	**0.76 ± 0.00**	**0.78 ± 0.18**	**0.80 ± 0.09**	**0.77 ± 0.07**	**0.74 ± 0.20**	**0.78 ± 0.01**
SSL	0.84 ± 0.12	**0.82 ± 0.09**	0.84 ± 0.04	0.71 ± 0.05	0.70 ± 0.00	0.75 ± 0.05	0.69 ± 0.15	0.62 ± 0.15	0.70 ± 0.03

Table B.8: MCC values for different approaches for different levels of incomplete supervision (II)

	SVM	KNN	Logit	SVM	KNN	Logit	SVM	KNN	Logit
	HTRU2 (easy)			HTRU2 (medium)			HTRU2 (hard)		
Gold	0.92 ± 0.09	0.83 ± 0.12	0.91 ± 0.15	0.86 ± 0.14	0.86 ± 0.08	0.89 ± 0.19	0.92 ± 0.06	0.88 ± 0.02	0.90 ± 0.01
Baseline	0.79 ± 0.12	0.83 ± 0.00	0.79 ± 0.06	0.67 ± 0.16	0.71 ± 0.15	0.73 ± 0.02	0.37 ± 0.06	0.37 ± 0.12	0.32 ± 0.09
S. Mendr	**0.87 ± 0.08**	**0.96 ± 0.04**	**0.80 ± 0.04**	**0.81 ± 0.16**	**0.86 ± 0.2**	**0.84 ± 0.01**	**0.79 ± 0.16**	**0.72 ± 0.11**	**0.84 ± 0.01**
SSL	0.70 ± 0.15	0.78 ± 0.12	0.79 ± 0.08	0.74 ± 0.18	0.77 ± 0.2	0.77 ± 0.17	0.55 ± 0.18	0.63 ± 0.12	0.71 ± 0.05
	MoCap (easy)			MoCap (medium)			MoCap (hard)		
Gold	0.85 ± 0.02	0.87 ± 0.16	0.85 ± 0.19	0.84 ± 0.09	0.86 ± 0.12	0.92 ± 0.06	0.87 ± 0.13	0.86 ± 0.15	0.9 ± 0.17
Baseline	0.78 ± 0.05	0.71 ± 0.16	0.72 ± 0.09	0.74 ± 0.01	0.66 ± 0.08	0.71 ± 0.02	0.61 ± 0.18	0.20 ± 0.04	0.47 ± 0.18
S. Mendr	**0.90 ± 0.16**	**0.80 ± 0.00**	**0.86 ± 0.11**	**0.79 ± 0.03**	**0.74 ± 0.19**	0.77 ± 0.19	**0.77 ± 0.08**	**0.71 ± 0.14**	**0.81 ± 0.02**
SSL	0.81 ± 0.08	0.79 ± 0.09	0.81 ± 0.15	0.75 ± 0.18	0.72 ± 0.04	**0.81 ± 0.04**	0.64 ± 0.05	0.69 ± 0.14	0.66 ± 0.18
	penbased (easy)			penbased (medium)			penbased (hard)		
Gold	0.91 ± 0.05	0.84 ± 0.14	0.87 ± 0.09	0.89 ± 0.13	0.86 ± 0.05	0.87 ± 0.13	0.92 ± 0.03	0.87 ± 0.2	0.86 ± 0.06
Baseline	0.84 ± 0.02	0.78 ± 0.14	0.65 ± 0.11	0.84 ± 0.19	0.54 ± 0.11	0.45 ± 0.00	0.72 ± 0.06	0.29 ± 0.16	0.33 ± 0.04
S. Mendr	**0.92 ± 0.19**	**0.79 ± 0.12**	**0.90 ± 0.19**	**0.86 ± 0.03**	**0.78 ± 0.11**	**0.83 ± 0.16**	**0.80 ± 0.12**	**0.75 ± 0.15**	**0.79 ± 0.01**
SSL	0.87 ± 0.16	0.75 ± 0.16	0.86 ± 0.02	0.84 ± 0.03	0.77 ± 0.08	0.70 ± 0.05	0.75 ± 0.12	0.62 ± 0.08	0.64 ± 0.02
	shoppers intention (easy)			shoppers intention (medium)			shoppers intention (hard)		
Gold	0.87 ± 0.18	0.85 ± 0.00	0.82 ± 0.14	0.88 ± 0.12	0.83 ± 0.02	0.83 ± 0.19	0.86 ± 0.18	0.89 ± 0.08	0.86 ± 0.05
Baseline	0.75 ± 0.01	0.76 ± 0.02	0.77 ± 0.18	0.74 ± 0.05	0.63 ± 0.08	0.70 ± 0.12	0.60 ± 0.13	0.17 ± 0.01	0.29 ± 0.18
S. Mendr	**0.88 ± 0.17**	**0.84 ± 0.12**	**0.85 ± 0.12**	**0.88 ± 0.04**	**0.80 ± 0.03**	**0.82 ± 0.14**	**0.78 ± 0.14**	**0.73 ± 0.00**	**0.68 ± 0.16**
SSL	0.85 ± 0.11	0.79 ± 0.03	0.84 ± 0.16	0.83 ± 0.02	0.69 ± 0.16	0.74 ± 0.06	0.59 ± 0.12	0.59 ± 0.08	0.56 ± 0.16
	shuttle (easy)			shuttle (medium)			shuttle (hard)		
Gold	0.93 ± 0.04	0.87 ± 0.04	0.85 ± 0.02	0.96 ± 0.09	0.87 ± 0.12	0.87 ± 0.19	0.91 ± 0.09	0.88 ± 0.16	0.87 ± 0.14
Baseline	0.84 ± 0.01	0.79 ± 0.01	0.66 ± 0.07	0.75 ± 0.04	0.78 ± 0.04	0.55 ± 0.18	0.73 ± 0.17	0.37 ± 0.01	0.29 ± 0.02
S. Mendr	**0.93 ± 0.07**	**0.80 ± 0.18**	0.84 ± 0.08	**0.82 ± 0.00**	**0.80 ± 0.03**	**0.84 ± 0.02**	**0.82 ± 0.01**	**0.73 ± 0.12**	**0.79 ± 0.03**
SSL	0.86 ± 0.19	0.77 ± 0.11	**0.86 ± 0.15**	0.81 ± 0.05	0.77 ± 0.03	0.76 ± 0.14	0.73 ± 0.04	0.66 ± 0.04	0.60 ± 0.13
	statlog (easy)			statlog (medium)			statlog (hard)		
Gold	0.97 ± 0.14	0.89 ± 0.19	0.85 ± 0.00	0.96 ± 0.03	0.89 ± 0.07	0.89 ± 0.04	0.91 ± 0.14	0.96 ± 0.1	0.86 ± 0.18
Baseline	0.79 ± 0.14	0.78 ± 0.02	0.61 ± 0.09	0.64 ± 0.17	0.74 ± 0.09	0.43 ± 0.2	0.57 ± 0.11	0.55 ± 0.14	0.15 ± 0.09
S. Mendr	**0.88 ± 0.02**	**0.92 ± 0.16**	**0.86 ± 0.17**	0.85 ± 0.06	**0.86 ± 0.19**	0.75 ± 0.05	**0.82 ± 0.04**	**0.76 ± 0.03**	**0.74 ± 0.02**
SSL	0.87 ± 0.03	0.87 ± 0.08	0.83 ± 0.17	**0.88 ± 0.18**	0.75 ± 0.05	**0.77 ± 0.07**	0.68 ± 0.16	0.66 ± 0.02	0.59 ± 0.12
	twonorm (easy)			twonorm (medium)			twonorm (hard)		
Gold	0.94 ± 0.12	0.91 ± 0.2	0.91 ± 0.02	0.92 ± 0.13	0.90 ± 0.02	0.87 ± 0.08	0.93 ± 0.12	0.92 ± 0.14	0.89 ± 0.05
Baseline	0.78 ± 0.06	0.77 ± 0.01	0.75 ± 0.11	0.68 ± 0.18	0.73 ± 0.00	0.52 ± 0.13	0.50 ± 0.01	0.71 ± 0.13	0.34 ± 0.07
S. Mendr	0.86 ± 0.01	**0.86 ± 0.19**	**0.88 ± 0.15**	**0.80 ± 0.11**	**0.81 ± 0.05**	**0.80 ± 0.01**	**0.79 ± 0.12**	**0.89 ± 0.19**	**0.87 ± 0.15**
SSL	**0.93 ± 0.17**	0.79 ± 0.05	0.81 ± 0.01	0.73 ± 0.09	0.76 ± 0.08	0.78 ± 0.00	0.75 ± 0.04	0.72 ± 0.12	0.72 ± 0.16
	yeast (easy)			yeast (medium)			yeast (hard)		
Gold	0.87 ± 0.07	0.89 ± 0.06	0.94 ± 0.17	0.85 ± 0.08	0.90 ± 0.17	0.92 ± 0.06	0.83 ± 0.14	0.86 ± 0.09	0.9 ± 0.15
Baseline	0.76 ± 0.05	0.74 ± 0.17	0.72 ± 0.06	0.65 ± 0.00	0.60 ± 0.00	0.68 ± 0.03	0.52 ± 0.11	0.47 ± 0.06	0.61 ± 0.19
S. Mendr	**0.82 ± 0.16**	**0.84 ± 0.17**	**0.90 ± 0.12**	**0.79 ± 0.07**	**0.79 ± 0.07**	**0.81 ± 0.15**	**0.68 ± 0.09**	**0.77 ± 0.08**	**0.78 ± 0.04**
SSL	0.81 ± 0.04	0.83 ± 0.18	0.81 ± 0.03	0.73 ± 0.17	0.67 ± 0.04	**0.81 ± 0.15**	0.60 ± 0.14	0.68 ± 0.00	0.64 ± 0.12

Appendix C. List of Contributions

List of Publications

[1] M. Nashaat, A. Ghosh, J. Miller, and S. Quader, "TabReformer: Unsupervised Representation Learning for Erroneous Data Detection," *ACM Transactions on Data Science,* 2020 (Submitted on June 2020).

[2] M. Nashaat, A. Ghosh, J. Miller, S. Quader, and J-F. Puget, "Dealing with Inaccurate and Incomplete Supervision," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 2020. (Submitted on February 2020).

[3] M. Nashaat, A. Ghosh, and J. Miller, "Using Intelligent Active Supervision to Predict Popularity of Mobile News," *Journal of Mobile Human-Computer Interaction*, 2020 (Submitted on March 2020, Attached in Appendix D).

[4] M. Nashaat, A. Ghosh, J. Miller, and S. Quader, "WeSAL: Applying Active Supervision to Find High-quality Labels at Industrial Scale," in *Proc. Hawaii International Conference on System Sciences 2020 (HICSS)*, Maui, Hawaii, USA., 2020, pp. 219-228.

[5] A. Ghosh, M. Nashaat, J. Miller, and S. Quader, "VisExPreS: A Visual Interactive Framework for User-driven Evaluations of Embeddings," *IEEE Transactions on Visualization and Computer Graphics*, 2020. (Under review)

[6] A. Ghosh, M. Nashaat, J. Miller, and S. Quader, "Context-Based Evaluation of Dimensionality Reduction Algorithms – Experiments and Statistical Significance Analysis," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 2020.

[7] A. Ghosh, M. Nashaat, J. Miller, and S. Quader, "Interpretation of Structural Preservation in Low-dimensional Embeddings," *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 2020.

[8] M. Nashaat, A. Ghosh, J. Miller, and S. Quader, "Asterisk: Generating Large Training Datasets with Automatic Active Supervision," *ACM Transactions on Data Science (TDS)*, vol. 1, no. 2, May 2020, doi: 10.1145/3385188.

[9] M. Nashaat, A. Ghosh, J. Miller, and S. Quader, “M-Lean: An End-to-end Development Framework for Predictive Models in B2B Scenarios,” *Information and Software Technology*, vol. 113, 2019, Pages 131-145, doi: 10.1016/j.infsof.2019.05.009.

[10] A. Ghosh, M. Nashaat, and J. Miller, “The Current State of Software License Renewals in The I.T. Industry,” *Information and Software Technology*, vol. 108, pp. 139–152, 2019.

[11] M. Nashaat, A. Ghosh, J. Miller, S. Quader, C. Marston, and J-F. Puget, “Hybridization of Active Learning and Data Programming for Labeling Large Industrial Datasets,” *in Proc. IEEE International Conference on Big Data (Big Data)*, Seattle, WA, USA, 2018, pp. 46-55. doi: 10.1109/BigData.2018.8622459. (Acceptance rate: 12%).

[12] A. Ghosh, M. Nashaat, J. Miller, S. Quader, and C. Marston, “A Comprehensive Review of Tools for Exploratory Analysis of Tabular Industrial Datasets,” *Visual Informatics*, vol. 2, no. 4, pp. 235–253, 2018.

[13] M. Nashaat, K. Ali, and J. Miller, “Detecting Security Vulnerabilities in Object-Oriented PHP Programs,” in *Proc. IEEE 17th International Working Conference on Source Code Analysis and Manipulation (SCAM)*, Shanghai, China, 2017, pp. 159-164. doi: 10.1109/SCAM.2017.20.

List of Patents

[1] M. Nashaat, S. Quader, P. Mierzejewski, "TabReformer: Bidirectional Representation Model for Erroneous Data Detection," Invention Reference P202005526, 2020.

[2] M. Nashaat, S. Quader, D. Reimer, "Semi-Supervised Ensemble Learning for Dealing with Inaccurate and Incomplete Supervision," Invention Reference 96046620, 2020.

[3] M. Nashaat, S. Quader, J-F. Puget, "Labeling Data using Automated Weak Supervision," United States Patent P201910742US01, Invention Reference P201910742, 2019.

[4] M. Nashaat, A. Basak, S. Quader, J. Miller, “Hybridization of Active Learning and Data Programming for Labelling Large Industrial Datasets", ‘PUBLISHED’, IBM Corporation, 2018.

Appendix D. Using Intelligent Active Supervision to Predict Popularity of Mobile News

Abstract

Browsing online content using mobile devices is gaining popularity and winning the battle against desktop web browsing. Therefore, estimating the popularity of online news articles can have significant impact through different applications like network traffic optimization. Previous studies proposed solutions that are tailored for specific conditions such as the availability of accurate ground-truth. In this paper, an improved prediction scheme is proposed to predict the long time popularity of online news articles without the need for ground-truth observations. The proposed framework applies a smart active learning selection policy to obtain the optimal amount of observations and achieve better predictive performance. To evaluate the proposed framework, an extensive set of experiments is conducted to compare it with state-of-the-art techniques. The experimental results indicate that the proposed solution can provide better prediction performance by up to 28.17% when compared to other methods while reducing the amount of required ground truth by 32% on average.

Keywords: Online Content Popularity, Classification Algorithms, Data Mining, Social Media, Machine Learning, Online News, Data Analysis, Predictive Models

Introduction

Online news portals have turned out to be an essential source of information. News is increasingly consumed on the go. The 24/7 news cycle is an ideal match for mobile presentation and consumption. Since they permit simple access to the latest news alongside with easy integration of social media platforms, the amount at which new content is published has reached extraordinary rates (Ye *et al.*, 2019). However, the popularity of news articles tends to show an unbalanced distribution. Previous studies (Rezaeenour *et al.*, 2018) show that while the majority of online content is barely noticed, only a small percentage of the published materials gain high popularity inferred with an increased number of votes (Rezaeenour *et al.*, 2018), comments (Tatar *et al.*,

2011), or shares on social media (Rezaeenour *et al.*, 2018). Hence, in a fundamental way, the value of the mobile-consumer interface is defined as the popularity and reach of content.

As a result, accurate estimation of the degree to which news articles will spread on the web can have valuable implications for many stakeholders such as advertising agencies, online marketing companies, online content providers, and news reporters. For instance, a predictive system that estimates news popularity can recommend how news articles should be organized in online portals to enhance the user browsing experience. Also, such systems can optimize data traffic within wireless networks. Since exchanging data, such as sharing news articles, forms an increasingly essential part in network traffic, predicting the popularity of news articles can substantially optimize network traffic by pre-caching popular content to mobile devices in idle hours and avoiding peak traffic time. Fundamentally, producing an article on-demand service.

For these reasons, several studies (Abbar *et al.*, 2018; Ahmed *et al.*, 2013; Bandari *et al.*, 2012; Deshpande, 2017; Rezaeenour *et al.*, 2018; Shreyas *et al.*, 2016; Tatar *et al.*, 2011; Yu-Jen Lin *et al.*, 2016) proposed techniques to predict the popularity of online content. Some approaches (Abbar *et al.*, 2018; Rezaeenour *et al.*, 2018; Yu-Jen Lin *et al.*, 2016) have focused on attribute selection to investigate the effect of different features. For example, considering specific attributes such as article topicality (Abbar *et al.*, 2018) and user posting behaviors (Yu-Jen Lin *et al.*, 2016) can have a substantial impact on the performance of the final model. Alternatively, other studies (Ahmed *et al.*, 2013; Bandari *et al.*, 2012; Tatar *et al.*, 2011) proposed different approaches for evaluating content popularity, like examining the popularity of offline content (Bandari *et al.*, 2012) or evolution patterns (Ahmed *et al.*, 2013). Finally, some research (Deshpande, 2017; Shreyas *et al.*, 2016) has experimented with different models to recommend a generic model for popularity predictions.

A closer look at these labeling techniques, however, reveals several gaps and challenges. One challenge is to determine which metrics should be used to express popularity (Abbar *et al.*, 2018). For example, different types of user feedback can define popularity, such as the number of user comments, the rating values, or the number of shares through social media. In many real-world applications, these metrics can be combined or even used interchangeably. Moreover, linking popularity metrics with the correct set of predictive features is an essential part of feature engineering (A. Ratner *et al.*, 2017). Since feature engineering is considered as one of the most

important tasks of any machine learning project (A. Ratner *et al.*, 2017), adopting different features according to each metric can be both expensive and time-consuming. Furthermore, several popularity factors, such as the quality of the written content or the importance of article topics to end-users, are difficult to quantify, which could further complicate the process of feature engineering.

However, the advent of new techniques of deep neural learning can alleviate most of the challenges associated with feature engineering by learning the task-specific representation of data. Nevertheless, this comes with another major upfront cost as these data-greedy techniques need massive training examples to achieve top predictive performance. Obtaining hand-labeled datasets is considered as another expensive task in the machine learning pipeline. Moreover, developing predictive systems for online content popularity depends on many varying factors, such as the structure of the news portal or the type of datasets. Therefore, different models may be required for each situation.

Moreover, changing the settings of any of these factors may result in rebuilding the model (Tatar *et al.*, 2011). Additionally, most of the existing models are developed using publicly available datasets, which may not always be accurate or even complete. Therefore, acquiring labeled datasets for such diverse settings had turned out to be an expensive yet indispensable task in the task of predicting the popularity of news articles.

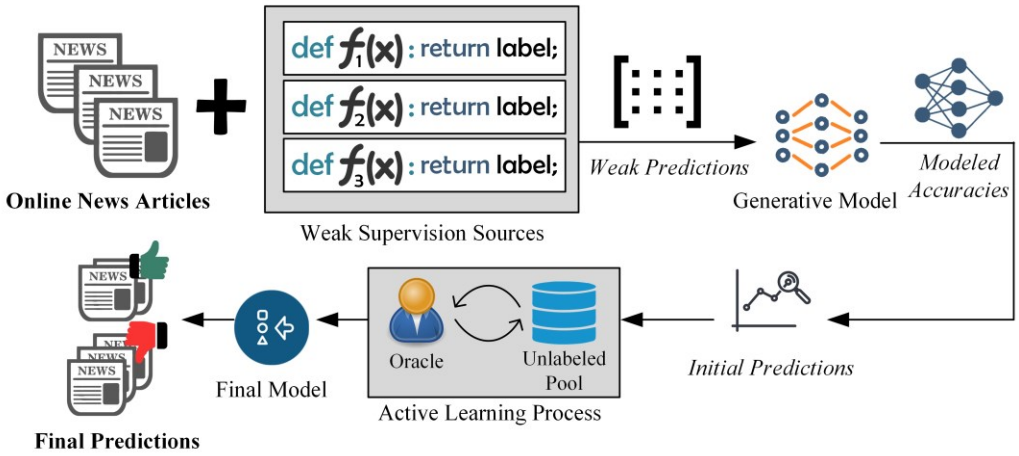


Figure 1. Overview of the proposed method

Therefore, motivated by the shortcomings of these approaches, in this article, an improved prediction scheme is presented to predict the long-time popularity of news articles without the need for ground-truth observations.

The contributions of this article are summarized as follows:

- A new prediction scheme for popularity prediction is offered. The scheme extends weakly generated labels (A. Ratner *et al.*, 2017) and includes humans-in-the-loop in a novel selection policy to rectify the inaccurate data points. Figure 1 illustrates an overview of the proposed model; the approach starts by collecting online news articles. Then the proposed method utilized a set of weak supervision sources to generate initial popularity predictions for the input articles. The proposed method is implemented, so it works with any weak supervision sources. However, the experiments focus on user-defined heuristics in the form of labeling functions (A. Ratner *et al.*, 2017; Varma *et al.*, 2017) since they are the most common mechanism to define weak labels (A. Ratner *et al.*, 2017). After that, the proposed system applies a meta-active learning process to query the user to provide labels for the most useful observations. The output of the proposed system is a trained model for popularity prediction, along with the final predictions generated by the learned selection process.
- An extensive set of experiments are performed to compare the proposed solution with three state-of-the-art techniques (Deshpande, 2017; Li *et al.*, 2018; Uddin *et al.*, 2016) along with traditional active learning strategies to predict online content popularity (Fu *et al.*, 2013). The experimental evaluation aims to estimate the effectiveness of the proposed model in popularity predictions with different classification models.
- To assess the impact of the experimental parameters, sensitivity analysis is conducted in which the labeling budget of the proposed methods is adjusted according to the number of annotations consumed by traditional active learning.

The paper is structured as follows: Section 2 discusses the related background. Section 3 presents the proposed method. The experimental results are discussed in Section 4. While Section 5 concludes the paper.

Related Work

The proposed method utilizes weak predictions along with meta-active learning (Fu *et al.*, 2013) to predict popularity for online content. Therefore, the related work spans across many areas, such as applying machine learning to predict content popularity, active learning as predictive models, and dealing with weakly supervised datasets.

Previous studies (Bao *et al.*, 2019; Garroppo *et al.*, 2018; Liu *et al.*, 2019) have emphasized feature engineering as one of the challenges that face popularity estimation. For example, authors in (Garroppo *et al.*, 2018) applied vocabulary clustering to online content to identify similar patterns of popular topics. Then, the model is used to estimate long-term popularity. Another research (Liu *et al.*, 2019) presents a preliminary analysis of content popularity before developing a regression model that employs the analysis results to predict popular trends in the future. Moreover, Bao *et al.* (Bao *et al.*, 2019) proposed a method that observes online content to decide on the most effective attributes to build the final feature-driven model. However, most of these approaches are content-specific. For example, they focus on certain types of content, such as videos (Garroppo *et al.*, 2018) and tweets (Bao *et al.*, 2019). Therefore, the final models are restricted to analyze content history within a single observed domain. Moreover, unlike the proposed method, none of these techniques have tried to include any domain experience in the learning process.

Alternatively, active learning (Fu *et al.*, 2013) is a special kind of semi-supervised learning in which a learner algorithm gets to choose which examples are added to the training set. This paradigm is proven to generate highly accurate models with minimum labeling effort. Active learning performs efficiently in situations where a large portion of the data is unlabeled, which is usually the case with online content. Most training data of online content are crawled from news portals that do not provide labels along with the data. Hence, active learning can be significantly useful in these settings. Active learning engages the users into the loop by asking them to label information to enhance the training performance of the underlying classifier. In pool-based active learning, the process is initialized with a small number of labeled instances (the seed) and a pool of unlabeled observations X_{train}. Then the learning algorithms iteratively ask the user to provide true labels for specific points from the pool. These points are then moved to the labeled set and used to retrain the classification model. The model is then evaluated using a held-out test set D_{test}, and the process is repeated. The iterative process terminates when either a performance threshold

is reached or a predefined annotation budget is exceeded. In active learning, the algorithm that decides which data instances the users should provide true labels is called the query strategy. There are many traditional query strategies (Fu *et al.*, 2013), such as uncertainty sampling that queries the user to provide labels for the samples about which the learner is most uncertain. Another selection policy is query-by-committee, which also queries the most uncertain samples. However, it measures the uncertainty differently, as it uses a committee of classifiers and queries the instance about which the committee members disagree.

Previous studies (S. Das Bhattacharjee *et al.*, 2017; Sreyasee Das Bhattacharjee *et al.*, 2019; Reis *et al.*, 2019) have applied active learning to different applications. For example, authors in (S. Das Bhattacharjee *et al.*, 2017) presented a human-machine collaborative model to detect misleading information in online content. The system applies active learning to cope with the problem of limited annotated samples. The system combines neural networks with active learning to reduce the labeling cost while attaining an acceptable performance. Another study (Sreyasee Das Bhattacharjee *et al.*, 2019) utilized active learning to identify malicious content in social media. The proposed model (Sreyasee Das Bhattacharjee *et al.*, 2019) initially creates a view-dependent classifier from a small labeled data and then applies active learning to enhance the model performance with additional annotated examples.

Moreover, another system is presented in (Reis *et al.*, 2019) to classify fake news by randomly selecting different sets of features to create a huge number of unbiased models; then, these models are ranked to define the best outcomes. However, although active learning has been applied to a wide range of applications, none of these approaches has tried to examine the problem of predicting the popularity of online news. Although, since most of the publicly available datasets are known to be inaccurate, active learning can provide suboptimal solutions due to the high level of noise in input data (Fu *et al.*, 2013).

Finally, weakly supervised datasets (Zamani & Croft, 2018) have been gaining popularity in machine learning tasks. Since obtaining hand-labeled large datasets has turned to be an impractical in many applications (Zamani & Croft, 2018), inexpensive weakly supervised labels can be utilized to create accurate predictive models. In weak supervision, subject-matter experts provide some form of higher-level, low-quality supervision sources like user-defined labeling function and knowledge bases (Zamani & Croft, 2018) to create training labels which are expected to be noisy.

Since weakly supervised datasets are mostly applied to applications where obtaining accurately labeled datasets can be expensive, previous research (Meng *et al.*, 2018; Shu *et al.*, 2020) has focused on text understanding, document categorization, and intent classification. For example, Meng *et al.* (Meng *et al.*, 2018) have proposed a weakly-supervised method for text classifications. The model first generates a pseudo-document to pre-train the model and then fine-tune it using real unlabeled data. The proposed model applies different types of weak supervision to obtain enough training data for deep learning models. Alternatively, another recent study (Shu *et al.*, 2020) utilizes weak supervision sources from social media to detect fake news articles with limited labeled data. The research (Shu *et al.*, 2020) proposes a framework in which data is first collected from multiple weak sources to train a model. Then, the model runs an inference module to use the learned feature representation to predict labels for unseen data.

However, a closer look at these efforts reveals several shortcomings. First, applying weak sources usually results in imperfect data with conflicting and noisy data points, which affects the performance of the final model. Although most of these approaches (Meng *et al.*, 2018; Shu *et al.*, 2020) have tried to automatically de-noise the data, the complex structure of these models makes it challenging for users to trust their outcomes. Secondly, none of these approaches (Meng *et al.*, 2018; Shu *et al.*, 2020) have tried to engage the users in the process of training the model or assessing its performance to increase user trust. Therefore, the effectiveness of engaging the user to debug these weakly supervised sources in the domain of predicting news popularity is yet to be tested, which is what this research tries to accomplish.

```
def LF_popularity(X):
    links = x.links.count();
    images = x.images.count();
    if images > images_avg and links > links_avg:
        return 1;
    if images <= images_avg and links < links_avg:
        return -1;
    else:
        return 0;
```

Figure 2. Example of a user-defined labeling function the predicts popularity based on the count of image and links in an article

The Proposed Method

The input to the proposed system is a collection of news articles D_N characterized as $\{\mathbf{x}_i, y_i\}_{i=1}^{N}$ where $\mathbf{x}_i$ is a set of features depicting the i^{th} article in the dataset, and $\mathbf{y}_i$ denotes the unknown popularity flag associated with this point. As for the input $\mathbf{x}_i \in R^F$ is described as a set of A attributes to represent each article. For example, the attributes for a given article can include the number of links and images the article contains and its title subjectivity (Rezaeenour *et al.*, 2018). Since these attributes are a set of quantifiable features of the observed article, the set of attributes describing the i^{th} article can be represented by a feature vector $\mathbf{x}_i$. The proposed method also requires a small labeled set of articles of size M as $D_M = \{\mathbf{x}_i^*, y_i^*\}_{i=1}^{M}$ with known popularity y_i^* where M << N. A for the output, the final model predicts popularity flags for the articles in D_N as a boolean label where $y_i^* \in \{-1, 1\}$.

As Figure 1 shows, the proposed model starts by letting the users provide a group of F labeling functions of size L described as $\{f_j\}_{j=1}^{L}$, where f_j: $\mathbf{X} \rightarrow \{-1, 0, 1\}$. In other words, each labeling function outputs a weak prediction for each article in D_N to denote its anticipated popularity based on some user-defined heuristics. An example of a labeling function in Figure 2. As the figure shows, the function can either output a weak prediction {-1, 1}, or abstain {0}. Consequently, the result of applying all the labeling functions F to X is a sparse matrix S where:

$$S_{i,j} = f_j(\mathbf{x}_i) \text{ where } 1 \leq i \leq N \text{ and } 1 \leq j \leq L \quad (1)$$

Afterward, the proposed method applies a generative model M_G (A. J. Ratner *et al.*, 2016) to model the accuracies of these labeling functions. The generative model models S a factor graph by encoding three factors: labeling propensity, labeling accuracy, and the function correlation for each pair of functions. These factors can be formally defined respectively as:

$$\emptyset^{lab}{}_{i,j}(F, Y) = \mathbf{1}\{f_{i,j} \neq 0\} \quad (2)$$

$$\emptyset^{Acc}{}_{i,j}(F, Y) = \mathbf{1}\{f_{i,j} = y_i\} \quad (3)$$

$$\emptyset^{Corr}{}_{i,j,k}(F, Y) = \mathbf{1}\{f_{i,j} = f_{i,k}\} \text{ where } j, k \in P \quad (4)$$

Where P is a set of functions pairs (Bach *et al.*, 2017). As mentioned earlier, these labeling functions depend on imperfect user-defined heuristics. Therefore, their outputs conflict and

disagree on certain points or even abstain, which results in incomplete data. Hence, the proposed method formally describes the pairwise disagreements as:

$$\emptyset^{dis}{}_{i,j,k}(F,Y) = \mathbf{1}\{f_{i,j} \neq f_{i,k}\} \text{ where } j,k \in P, i \in N \quad (5)$$

Furthermore, the method denotes the abstaining conditions, as

$$\emptyset^{abstain}{}_{i,j}(F,Y) = \mathbf{1}\{f_{i,j} = 0\} \quad (6)$$

At this point, the proposed method tries to enhance the accuracy of the labeling function by applying a meta-active learning process. The process designs the selection policy by framing the problem as a regression problem. The active learning stage aims at training a selection policy so that, when applied to a dataset, it selects the data points that would result in the maximum reduction to the generalization error. A detailed process view of the meta-active learning process is illustrated in Figure 3. As the figure shows, the process consists of two main steps, namely, *designing the selection policy* and *applying the policy as a meta-active learning process*.

Designing the selection policy. As for designing the selection policy, the step is outlined as a regression problem. To initialze the regression process, the method first collects a set of labeled observation $D_S = \{\gamma_i, \nabla_i\}_{i=1}^{Q}$ to train the selection policy where γ_i describes a set of attributes for the i^{th} example in D_S. To only include the attributes that are related to data distribution, the model

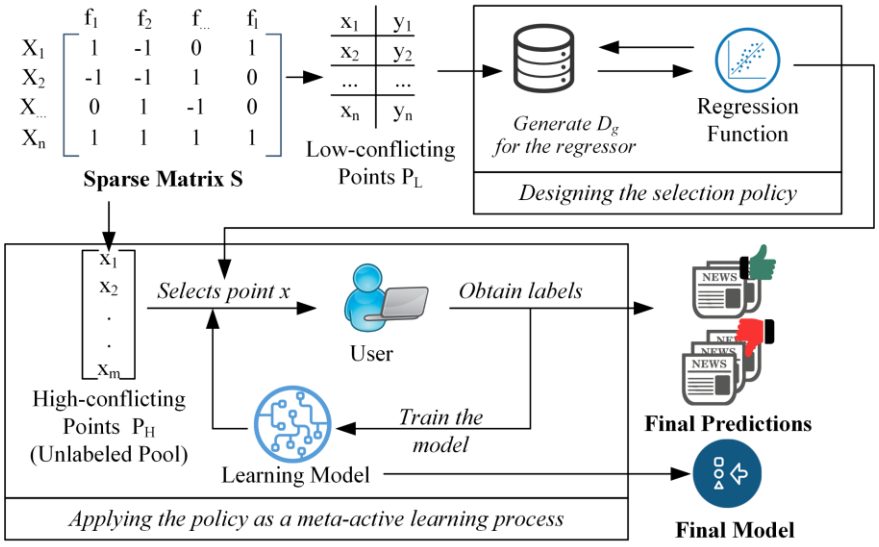


Figure 3. A process view of the meta-active learning process

considers the values of the factors in Equations (2)-(6). On the other hand, ∇_i describes the prospective reduction to the generalization error after adding the i^{th} point to the labeled pool. To gather these labeled examples, the model first classifies the points in S into high-conflicting points P_H and low-conflicting points P_H. The high-conflicting dataset contains the points about which the labeling functions are disagreeing or abstaining. It can be defined as:

$$P_H \subseteq \mathbf{X}, \forall x_i \in D_N \{x_i | \emptyset^{dis}{}_{i,j,k}(F,Y) = \mathbf{1}\{f_{i,j} \neq f_{i,k}\} \cup \emptyset^{abstain}{}_{i,j}(F,Y) = \mathbf{1}\{f_{i,j} = 0\} \quad (7)$$

While the low-conflicting points are denoted as $P_L = D_M \cup (D_N - P_H)$. Then, the low-conflicting points P_L is used to train and evaluate a model M_S. The model is first trained and evaluated on a subset of P_L so the initial generalization error L_g is recorded. Then, the proposed model iteratively adds a new point x from P_L to the training dataset. After that, the model is evaluated again to record the generalization error related to this point L_x. Finally, the reduction in the classification loss is computed and recorded as $\nabla_x = L_g - L_x$. Consequently, the result of this process is the new training dataset D_s that is used later to train the regressor.

Applying the policy. Accordingly, D_s is then used to train a random forest regressor g (Shreyas *et al.*, 2016) as the final selection policy that is built while considering the distribution of the underline space matrix S. The selection policy is then applied to P_H to greedily choose the points with the highest potential error reduction by taking the maximum of the value predicted by the regressor g as:

$$x^* = \arg\max_{x \in D_{Test}} g(\gamma_x) \quad (8)$$

The model then applies the regressor function g to rank the data points in P_H. The time complexity of the ranking step is highly decreased as the size of P_H is much smaller the number of articles in D_N. Therefore, in each iteration of the active learning process, the regressor function ranks the points in P_H using (9). Then, the points denoting the articles with the highest reduction in the generalization error are selected. Next, the user is queried to provide true labels these points, which are then added to the set of final predictions. Finally, this set of predictions is used to retrain a classifier *f* for news popularity. As the iterations of active learning progress, the proposed method gradually builds a set of predictions D_{AL} which represents the data points that received true labels from the user during this stage. The process also outputs a predictive model *f* which is trained using

D_{AL} and can estimate popularity for unseen articles. A complete algorithm of the proposed method is shown in Algorithm 1.

Algorithm 1 The Proposed Method

Input: an unlabeled dataset of news articles D_N, small labeled dataset D_M, a set of labeling functions F, predefined labeling cost.

Output: Final classifier *f* for popularity predictions.

1: Apply F to D_N to generate a sparse matrix S of weak labels.

2: Compute disagreements factor $Ø^{dis}$(F, Y) (Equation 5)

3: Compute abstaining labels factor $Ø^{abstain}$(F, Y) (Equation 6)

4: Classify S into P_H and P_L (Equation 7)

5: Split P_L into training and testing sets and inialize an empty trainng set $D_S = \{\gamma_i, \nabla_i\}_{i=1}^{Q}$

6: Train a classification model M_S using a subset of P_L

7: Calculate the test loss L_g

8: **Loop** for each point in the training set

9: Add a point x to the training set

10: Calculate the new test loss L_x

11: Compute the reduction in the classification loss as $\nabla_x = L_g - L_x$

12: Collect the data point parameters γ_x as in Equations (2)-(6).

13: Add $\{\gamma_x, \nabla_x\}$ to D_S

14: **End**

15: Train a random forest regressor g using D_S

16: Initialize the unlabeled pool as P_H

17: **Loop** until labeling cost is exceeded

18: Apply g to select a point x_i from P_H (Equation 8)

19: Ask the user to provide a correct label for x_i

20: Add the labeled point x_i to the set of final predictions D_{AL}

21: Train classifier *f* using D_{AL}

22: **End**

23: return f as the final model for popularity prediction

Experimental Evaluation

The experiments seek to estimate the effectiveness of the proposed method in popularity predictions for online news articles and compare it to the state-of-the-art techniques. To accomplish such a goal, the experimental evaluation considers different metrics of classification performance along with the number of training examples needed to train each of the methods engaged in the evaluation.

Description of Datasets

The experiments include several datasets with different sizes and dimensionality. A description of datasets is presented here and summarized in Table 1. The table shows, for each dataset, the size of the data (Size), the number of attributes (Dim.), the popularity measure that is used in the experiments (Popularity Measure) and the ratio of the positive class (popular articles) to the dataset size (+/Size).

- **Online News Popularity (Online News)**: This is a real-world dataset that is offered by the University of California at Irvine (UCI) Machine Learning Repository. It contains news articles published on Mashable media platforms, which are retrieved from 2013 to 2015. The dataset contains more than 39k articles with 61 attributes. The popularity term is measured by the number the article URL is shared on twitter.

Table 1. Overview of the datasets

Dataset	Size	Dim.	Popularity Measure	+/ Size
News	39,797	61	# shares	49.34
Reddit Engagement	89,314	12	# commetns (Reddit)	13.12
Webhose News	170,882	84	# comments (Facebook)	33.19

Table 2. Experimental settings

Dataset	# Labeling Functions	Labeling Functions Performance				Active Learning Settings		
		Acc	P	R	F1	Seed	X_{train}	D_{test}
Online News	6	0.74	0.82	0.78	0.80	1,989	24,675	13,133
Reddit Engagement	7	0.83	0.68	0.72	0.70	4,287	58,054	26,973
Webhose News	9	0.66	0.71	0.77	0.74	8,544	111,073	51,265

- **Reddit Community Engagement Dataset (Reddit Engagement):** This is a dataset of Reddit news articles crawled for three months from June to August 2017. The dataset contains 89,314 news posts with 12 attributes. The experiments consider predicting popularity for each post in terms of engagement stats and the number of comments.
- **Webhose's Popular News Article (Webhose News)**: This is another real-world dataset that is provided by Webhose. The dataset has more than 170,000 news articles with 84 attributes. The dataset considers topics from 7 categories and 12 languages where the popularity is measured by the number of comments the article received on Facebook.

Experiments Settings

Baseline methods. The experiments compare the proposed method with three baseline strategies:

- **Gradient boosting learning approach (GBM)** presented in (Uddin *et al.*, 2016). The technique extends gradient boosting models to predict the number of shares using an ensemble of learning algorithms.
- **Vector space model (VSM)** proposed in (Li *et al.*, 2018), which applies a two-stage selection approach to predict news popularity. The method first selects global features related to column information and then chooses local features associated with news popularity. Then the model reconstructs the final model with all the selected features.
- **Ensemble models (Ensemble)** presented in (Deshpande, 2017), which applies a group of predictive models to achieve better performance. The approach convenes decision trees along with boosting and bagging to achieve higher classification accuracy.

Table 3. Experimental results of comparison with baseline techniques

Model	Online News				Reddit Engagement				Webhose News			
	P	**R**	**MCC**	**F1**	**P**	**R**	**MCC**	**F1**	**P**	**R**	**MCC**	**F1**
Proposed Method	**0.88**	**0.97**	**0.96**	**0.92**	**0.93**	**0.88**	**0.92**	**0.90**	**0.91**	**0.95**	**0.85**	**0.93**
GBM	0.81	0.84	0.83	0.82	0.83	0.82	0.84	0.82	0.89	0.83	0.81	0.86
VSM	0.86	0.89	0.91	0.87	0.91	0.80	0.88	0.85	0.71	0.91	0.72	0.80
Ensemble	0.74	0.92	0.84	0.82	0.83	0.79	0.85	0.81	0.86	0.82	0.71	0.84

Writing the labeling functions. Since the proposed method requires providing a set of user-defined heuristics, the experiments consider threshold-based labeling function. In this type, the function assigns a popularity prediction to a given article based on certain attributes (e.g., number of images in the article). The experiments rely on pattern matching methods to create the labeling function used in the experiments. Since these methods are considered as the best practice found in the literature (A. Ratner *et al.*, 2017; Varma *et al.*, 2017; Varma & Ré, 2018). Furthermore, to develop high accuracy labeling functions, the experiments used the set of labeled articles D_M to develop and evaluate the empirical accuracy of the generated functions. The proposed method only accommodates the labeling functions with accuracy more than a predefined threshold of 60% (A. Ratner *et al.*, 2017). The experimental settings for the proposed method are summarized in Table 2. The table shows the number of labeling functions generated for each dataset and the evaluation metrics for the generated labeling functions.

Active Learning settings. Since the proposed method applies a process of meta-active learning, the experiments have to set a stopping condition for the iterative active learning process. To select the stopping condition, another set of experiments are conducted with different sampling techniques of active learning. The experiments applied uncertainty sampling (UNC), query-by committee (QBC), and random sampling (RAND) (Fu *et al.*, 2013) with each dataset and examined the learning curves in each situation. The experiments are averaged over ten runs and stopped the active learning process when the learning curve shows no enhancements with additional points (Bloodgood & Vijay-Shanker, 2009). Then, to maintain fairness throughout the experiments, the same number of iterations is adopted for the proposed method. The experimental settings for active

learning are also depicted in Table 2. For each dataset, the table shows the seed, the initial size of X_{train}, and the size of the test set D_{test} used to evaluate the classifier.

Experiments Results

The following subsections present the results of comparing the proposed method with other predictive methods.

Comparison with Baseline methods

First, the proposed method is compared with a set of predictive models for online popularity. Table 3 shows the experimental results. As the experiments aim to avoid measurement bias, several performance metrics are reported, which include Precision (P) and Recall (R), and F1 measure (F1). Moreover, to report prediction accuracy, the experiments consider the Matthews correlation coefficient (MCC) (Powers, 2011) to describe the confusion matrix instead of accuracy since classification accuracy can be misleading with imbalanced datasets.

As the table depicts, the proposed method achieved higher precision and recall in all the problems. Since the proposed method applies a meta-active learning process to enhance the accuracy of the generated predictions, it managed to achieve better results than the comparing tools. In the online news dataset, the proposed method improves precision by up to 18.92% when compared to the ensemble model. The reason behind this enhancement is due to the good quality of the labeling function in this dataset (Table2). Therefore, the active learning process could rectify a higher number of predictions with the assigned labeling budget. As for the prediction accuracy, the proposed method also outperforms other methods by achieving higher MCC values in all the tasks. On average, the proposed model improved the accuracy of the generated predictions by 3.75%, 5.79%, and 3.90% when compared to GBM, VSM, and ensemble models, respectively. Overall, the results show that the proposed method can maintain a comparative prediction performance for online news popularity when compared to the state-of-the-art techniques.

Comparison with Active learning

The experiments compare the proposed method with active learning for the task of popularity prediction. The main goal of these experiments is to determine the labeling budget for the proposed

Table 4. Experimental results of comparison with active learning (uncertainty sampling)

Dataset	Proposed Method					Active Learning (UNC)				
	P	R	MCC	Acc	# queried examples	P	R	MCC	Acc	# queried examples
Online News	0.93	0.95	0.85	0.92	5,374	0.89	0.9	0.8	0.9	7,764
Reddit Engagement	0.95	0.91	0.91	0.93	13,613	0.91	0.89	0.81	0.93	21,638
Webhose News	0.81	0.92	0.86	0.95	34,381	0.79	0.74	0.62	0.94	48,298

method and how it is compared to the traditional active learning process. The authors applied three query strategies to the three datasets, namely UNC, QBC, and RAND . The learning curves of the three query strategies are shown in Figure 4. The learning curves demonstrate the relationship between accuracy and the number of labeled articles consumed to achieve the corresponding accuracy value. The curves in the figure show that UNC attained the highest efficiency for the three datasets.

Therefore, the experiments report the performance metrics achieved by UNC and the proposed method in Table 4. The table shows the evaluation metrics attained by the proposed method and UNC, along with the number of labeled articles needed to obtain the reported accuracy values. The table illustrates that the proposed method achieved better MCC values than UNC in the three datasets with an overage improvement of 19.10%. The maximum improvement is achieved in the Webhose News dataset with 38.71%. The table also shows that the proposed method maintains less labeling budget than traditional active learning, which proves that the learned selection policy in the proposed method managed to reduce the cost of manual labeling. As mentioned before, the active learning process in the proposed method starts with an unlabeled pool with a much smaller size than the unlabeled pool of traditional active learning. Thus, the budget for manual labeling is highly reduced. For example, in the online news dataset, traditional active learning needed to label 31.47% of the training pool, while the size of the unlabeled pool in the proposed method only represents 21.78% of the training set size, which results in 30.78% decrease in labeling cost. Likewise, the proposed method reduced the labeling budget in the Reddit Engagement dataset and the Webhose News by 37.09% and 28.81% when compared to UNC, respectively.

Moreover, the results indicate that the proposed method achieved better precision and recall values than traditional active learning in the three datasets. For Webhose News, the proposed method surpassed the recall values of active learning by 24.32%. Similarly, it improved the precision value in the same dataset by 2.53%. Generally, the results empirically demonstrate that the models generated by the proposed method achieve remarkable results in real-world situations in popularity predictions for online news.

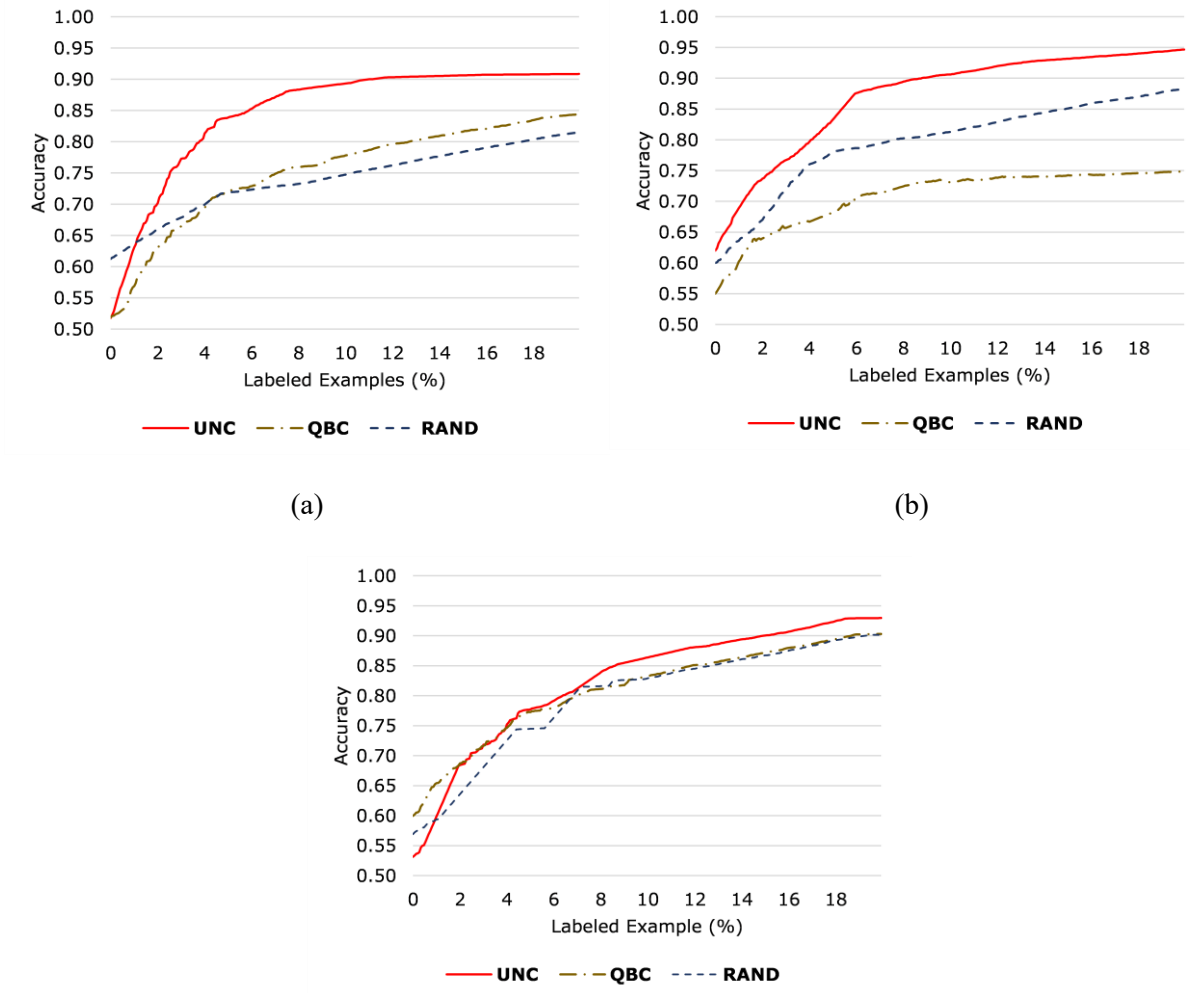


Figure 4. Learning curves of active learning for (a) Online news dataset (b) Reddit Engagement dataset (d) Webhose News dataset

Table 5. Experimental results with different values of λ

Dataset	λ	Active Learning		WeSAL	
		Size of X_{train}	AL Cost %	P_H %	$B_{Labeling}$
News	0.001	24,675	12.20%	18.22%	3,010
	0.0001		31.47%		7,764
	0.00001		37.60%		9,278
Reddit Engagement	0.001	58,054	6.20%	17.19%	3,599
	0.0001		14.00%		8,128
	0.00001		16.60%		9,637
Webhose News	0.001	111,073	8.01%	21,81%	8,886
	0.0001		19.11%		20,660
	0.00001		24.91%		26,658

Sensitivity analysis of the experimental parameters

To test the proposed method with different configurations, another set of experiments are performed to assess the impact of the stopping condition of the active learning process in the proposed method. As mentioned before, the experiments terminated the traditional active learning process when the improvements of classification accuracy do not exceed a threshold λ=0.0001 for a successive number of iterations (Bloodgood & Vijay-Shanker, 2009). The experiments set the same number of labeled articles consumed by traditional active learning as the labeling cost for

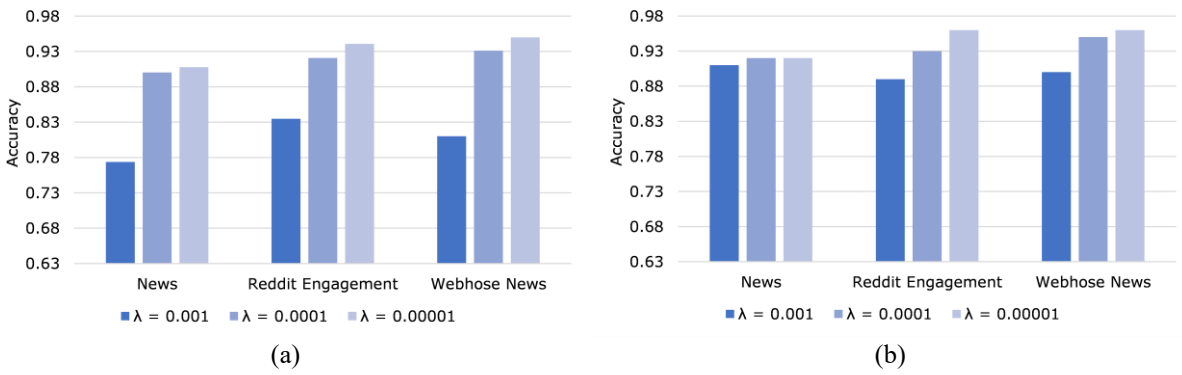


Figure 5. Accuracy values for (a) active learning (UNC) (b) the proposed method values of λ = 0.001, 0.0001, 0.00001

the proposed method. Thus, to test the sensitivity of the stopping criterion, the experiments are repeated with different values of λ. First, the experiments with traditional active learning are repeated with values of λ = 0.001, 0.0001, 0.00001. Figure 5.a shows the accuracy values achieved by the underline classifiers with UNC using the three datasets.

Furthermore, the labeling budget of the proposed method is customized according to the number of annotations consumed by UNC in each dataset. Table 5 shows the size of the unlabeled pool (X_{train}), the annotation budget used by UNC for each value of λ as a percentage of the total size of the unlabeled pool (Labeling cost%), and the size of the unlabeled pool in the proposed method P_H as a percentage of X_{train}. As the table shows, the size of P_H is much smaller than the X_{train} in all the datasets since it only contains the high conflicting predictions generated from the labeling functions. Also, the accuracy values achieved by the proposed method are reported in Figure 5.b.

As figure 5 shows, when the value of λ increases, this can terminate active learning too early, which results in missing useful generalizations (Bloodgood & Vijay-Shanker, 2009). For example, setting λ = 0.001 decreased the accuracy of UNC in the Online News dataset by 14.06 % when compared to the accuracy achieved when λ = 0.0001(Figure 5.a).

Moreover, the results also attest that the labeling budget tends to increase when λ is set to a small value (λ=0.00001). However, the additional cost of manual labeling does not result in a significant enhancement in classification performance. For example, with λ=0.00001, UNC increased its labeling budget in the Webhose News by 29.03%, but with only 2.03% enactment achieved in accuracy values when compared to the performance achieved with λ = 0.0001. Generally, the experimental results show that the choice of λ=0.0001is optimum since it managed to select the elbow values in the learning curves (Bloodgood & Vijay-Shanker, 2009).

Additionally, the results show that the proposed method maintained better results than active learning with different values of λ. Since the size of the unlabeled pool P_H is much less than the size of X_{train}, in some cases, the total size of P_H is less than the number of annotations consumed by active learning. Therefore, changing the value of λ did not affect the performance of the proposed method. Overall, the results illustrated in Figure 5 show that the proposed method managed to achieve better performance than active learning with all variation of λ in all the datasets.

Conclusions

In this paper, a new prediction scheme is proposed to predict the popularity of online news. Online news is consumed on the bus, the train, the car … essentially everywhere given the ubiquitous nature of modern mobile technology. This human – mobile interaction is predicated on getting the correct news article in front of the consumer at the right time. The proposed method does not require ground truth examples to generate the final predictions. Instead, it relies on initial noisy labels from high-level user-defined heuristics. Then, it rectifies these weakly supervised labels by applying a novel meta-active learning selection policy. The experimental results conducted with three real-world datasets show that the proposed method outperforms the state-of-the-art techniques by up to 19.72% in classification performance (MCC). The results also empirically prove that the proposed method could attain better results than traditional active learning while cutting the labeling budget by up to 37.09%.

References

Abbar, S., Castillo, C., & Sanfilippo, A. (2018). To Post or Not to Post: Using Online Trends to Predict Popularity of Offline Content. *Proceedings of the 29th on Hypertext and Social Media*, 215–219.

Ahmed, M., Spagna, S., Huici, F., & Niccolini, S. (2013). A Peek into the Future: Predicting the Evolution of Popularity in User Generated Content. *Proceedings of the Sixth ACM International Conference on Web Search and Data Mining*, 607–616.

Bach, S. H., He, B., Ratner, A., & Ré, C. (2017). Learning the Structure of Generative Models without Labeled Data. *ArXiv:1703.00854 [Cs, Stat]*.

Bandari, R., Asur, S., & Huberman, B. A. (2012). The Pulse of News in Social Media: Forecasting Popularity. *26–33.*

Bao, Z., Liu, Y., Zhang, Z., Liu, H., & Cheng, J. (2019). Predicting popularity via a generative model with adaptive peeking window. *Physica A: Statistical Mechanics and Its Applications*, *522*, 54–68.

Bhattacharjee, S. Das, Talukder, A., & Balantrapu, B. V. (2017). Active learning based news veracity detection with feature weighting and deep-shallow fusion. *2017 IEEE International Conference on Big Data*, 556–565.

Bhattacharjee, Sreyasee Das, Tolone, W. J., & Paranjape, V. S. (2019). Identifying malicious social media contents using multi-view Context-Aware active learning. *Future Generation Computer Systems*, 365–379.

Bloodgood, M., & Vijay-Shanker, K. (2009). A method for stopping active learning based on stabilizing predictions and the need for user-adjustable stopping. *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, 39–47.

Deshpande, D. (2017). Prediction Evaluation of Online News Popularity Using Machine Intelligence. *2017 International Conference on Computing, Communication, Control and Automation (ICCUBEA)*, 1–6.

Fu, Y., Zhu, X., & Li, B. (2013). A survey on instance selection for active learning. *Knowledge and Information Systems*, *35*(2), 249–283.

Garroppo, R. G., Ahmed, M., Niccolini, S., & Dusi, M. (2018). A Vocabulary for Growth: Topic Modeling of Content Popularity Evolution. *IEEE Transactions on Multimedia*, *20*(10), 2683–2692.

Li, Y., Peng, Q., Sun, Z., Fu, L., & Khokhar, S. (2018). A Two-stage Prediction Method of News Popularity only using Content Features. *2018 13th World Congress on Intelligent Control and Automation (WCICA)*, 767–772.

Liu, Y., Zhi, T., Xi, H., Duan, X., & Zhang, H. (2019). A Novel Content Popularity Prediction Algorithm Based on Auto Regressive Model in Information-Centric IoT. *IEEE Access*, *7*, 27555–27564.

Meng, Y., Shen, J., Zhang, C., & Han, J. (2018). Weakly-Supervised Neural Text Classification. *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, 983–992.

Powers, D. M. (2011). *Evaluation: From precision, recall and F-measure to ROC, informedness, markedness and correlation*.

Ratner, A., Bach, S. H., Ehrenberg, H., Fries, J., Wu, S., & Ré, C. (2017). Snorkel: Rapid training data creation with weak supervision. *Proceedings of the VLDB Endowment*, *11*(3), 269–282.

Ratner, A. J., De Sa, C. M., Wu, S., Selsam, D., & Ré, C. (2016). Data Programming: Creating Large Training Sets, Quickly. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, & R. Garnett (Eds.), *Advances in Neural Information Processing Systems* (pp. 3567–3575). Curran Associates, Inc.

Reis, J. C. S., Correia, A., Murai, F., Veloso, A., & Benevenuto, F. (2019). Explainable Machine Learning for Fake News Detection. *Proceedings of the 10th ACM Conference on Web Science*, 17–26.

Rezaeenour, J., Eili, M. Y., Hadavandi, E., & Roozbahani, M. H. (2018). Developing a New Hybrid Intelligent Approach for Prediction Online News Popularity. *International Journal of Information Science and Management (IJISM)*, *16*(1).

Shreyas, R., Akshata, D. M., Mahanand, B. S., Shagun, B., & Abhishek, C. M. (2016). Predicting popularity of online articles using Random Forest regression. *2016 Second International Conference on Cognitive Computing and Information Processing (CCIP)*, 1–5.

Shu, K., Wang, S., Lee, D., & Liu, H. (2020). Mining Disinformation and Fake News: Concepts, Methods, and Recent Advancements. *ArXiv:2001.00623 [Cs]*.

Tatar, A., Leguay, J., Antoniadis, P., Limbourg, A., de Amorim, M. D., & Fdida, S. (2011). Predicting the Popularity of Online Articles Based on User Comments. *Proceedings of the International Conference on Web Intelligence, Mining and Semantics*.

Uddin, M. T., Patwary, M. J. A., Ahsan, T., & Alam, M. S. (2016). Predicting the popularity of online news from content metadata. *2016 International Conference on Innovations in Science, Engineering and Technology (ICISET)*, 1–5.

Varma, P., Iter, D., De Sa, C., & Ré, C. (2017). Flipper: A Systematic Approach to Debugging Training Sets. *Proceedings of the 2nd Workshop on Human-In-the-Loop Data Analytics*.

Varma, P., & Ré, C. (2018). Snuba: Automating weak supervision to label training data. *Proceedings of the VLDB Endowment*, *12*(3), 223–236.

Ye, Q., Luo, Y., Chen, G., Guo, X., Wei, Q., & Tan, S. (2019). Users Intention for Continuous Usage of Mobile News Apps: The Roles of Quality, Switching Costs, and Personalization. *Journal of Systems Science and Systems Engineering*, *28*(1), 91–109.

Yu-Jen Lin, Mi-Yen Yeh, Fang-Yi Chiu, Ya-Hui Chan, & Chia-Chi Wu. (2016). Predicting popularity of articles on bulletin board system. *2016 International Conference on Big Data and Smart Computing*, 169–176.

Zamani, H., & Croft, W. B. (2018). On the theory of weak supervision for information retrieval. *Proceedings of the 2018 ACM SIGIR International Conference on Theory of Information Retrieval*, 147–154.