Using Associative Classifiers as a Surrogate for Explainability in Text Categorization

by

Md Tanvir Alam Anik

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

Master of Science

Department of Computing Science
University of Alberta

Abstract

Giving reasons for justifying the decisions made by classification models has received less attention in recent artificial intelligence breakthroughs than improving the accuracy of the models. Recently, AI researchers are paying more attention to filling this gap, leading to the introduction of the emerging topic of explainable AI (XAI). XAI is a field of artificial intelligence that aims to create more transparent and understandable AI systems. A form of XAI approach called "model-independent explanations" aims to offer explanations without requiring the internals of a trained model.

This study presents BARBE for text, a technique that can explain the decisions made by any black-box classifier for text datasets with a high degree of precision, without relying on information about the internal architecture of the model. A probability score from the black-box classifier is not necessary in order to use BARBE. In addition, BARBE offers explanations in two distinct formats: firstly, the generation of rules; secondly, the importance score for salient features. Because they more closely match human intuition, rules are seen to be a superior explanation method. Additionally, BARBE makes use of association rules. An association rule is in the form of "if X then Y," which means that if X occurs, then the likelihood of Y occurring increases as well. BARBE not only provides a single rule but also provides a set of rules where each rule may contain a conjunction of features. In this study, we introduce two different versions of BARBE and illustrate their capability to effectively generate explanations for sentences of varying lengths. We propose a data augmentation technique for BARBE that can generate more meaningful rules as the explanation.

Our study demonstrates the effectiveness of BARBE in generating explanations for detecting cyberbullying in the context of a resource-constrained language. The experimental analysis shows that BARBE outperforms other XAI frameworks in generating more convincing explanations for resource-constrained language. This is a significant finding as it demonstrates the potential of BARBE as a tool for improving the explainability of machine learning models trained using formal and informal embedding, even in contexts where data is limited or constrained.

Preface

BARBE for tabular data was proposed by Mohammad Hossein Motallebi Shabestari in his thesis. The author has extended his work to incorporate BARBE for text datasets. A conference paper regarding the implementation of BARBE for text has been submitted to Database and Expert Systems Applications (DEXA). A journal paper with some major enhancements will be submitted to ACM Knowledge and Information Systems (KAIS). The author is one of the co-authors of these papers.

In Chapter 6, the application of BARBE for generating explanations in the detection of cyberbullying is explored in the context of resource-constrained Language. This work started in the context of a course project for the course Data Mining and Knowledge Discovery (CMPUT 690) in Winter 2022 semester with two other co-authors: Md Saiful Islam and Nazmus Sakeef. Later, the project led to the submission of a paper to the ACM Transactions on Asian and Low-Resource Language Information Processing. The author is one of the co-authors of that paper.

The author is also a co-author of a survey paper regarding text data augmentation that will be submitted soon.

“The more you know, the more you realize how much you don’t know.”

-Aristotle

Dedicated to my family for supporting me through every step of my life.

Acknowledgements

I want to begin by expressing my sincere gratitude to Professor Osmar R. Zaiane. Without his support, this work would not have been possible. The interactions I had with Professor Osmar while I was studying at the University of Alberta, and the valuable insights I gained from him, will serve as a lasting source of inspiration throughout my professional career.

Furthermore, I would like to express my thanks to all the other members of the AMII XAI Lab for the valuable conversations we had. Thank you Randy, Housam, Shahin, Rayhan, Sheila.

Finally, I would like to express my heartfelt gratitude to my family. I am especially thankful to my parents, and my wife who have consistently provided me with their unwavering love and support, especially during challenging times. I am grateful to my daughter who cheered me up always. I want to express my appreciation to all of my friends in Edmonton who have made my stay in this lovely city a pleasant and enjoyable experience.

Special thanks to my co-authors Saiful and Maliha for their valuable contributions and hard work. Their dedication and insights were instrumental in bringing this work to completion, and I am deeply grateful for their collaboration.

Table of Contents

List of Tables

List of Figures

Abbreviations

AI Artificial Intelligence.

BARBE Black-box Association Rule-Based Explanations.

BARBE-S BARBE with SigDirect.

BARBE-C BARBE with CFAR.

BOW Bag-of-Words.

CAR Classification Association Rule.

CBA Classification Based on Associations.

CDA Counterfactual Data Augmentation.

CFAR Classification by Frequent Association Rules.

CMAR Classification based on Multiple Association Rules.

CNN Convolutional Neural Network.

CPAR Classification based on Predictive Association Rules.

CRF Conditional Random Field.

DA Data Augmentation.

DARPA Defense Advanced Research Projects Agency.

DNN Deep Neural Networks.

EDA Easy Data Augmentation.

EHR Electronic Health Records.

ES	Expert System.
EU	European Union.
FP-G	Frequent Pattern Growth.
GDPR	General Data Protection Regulation.
GLM	Generalised Linear Model.
HDA	Hierarchical Data Augmentation.
IG	Integrated Gradient.
LHS	Left Hand Side.
LIME	Local Interpretable Model-agnostic Explanations.
LM	Language Model.
LSTM	Long Short-term Memory.
ML	Machine Learning.
NER	Named-Entity Recognition.
NLP	Machine Learning.
NMT	Neural Machine Translation.
PALEX	Pattern Aided Local Explanation.
PSO	Particle Swarm Optimization.
RBF	Radial Basis Functions.
RF	Random Forest.
RFR	Relative Frequency Ratio.
RHS	Right Hand Side.
RIPPER	Repeated Incremental Pruning to Produce Error Reduction.

SHAP	SHapley Additive exPlanations.
SVM	Support Vector Machine.
TFIDF	Term Frequency-Inverse Document Frequency.
VQA	Visual Question-Answering.
XAI	eXplainable Artificial Intelligence.

Chapter 1

Introduction

1.1 Motivation

In recent years, an increasing number of businesses have been leveraging the benefits of Artificial Intelligence (AI) in their products, resulting in a more significant impact of intelligent machines on society than ever before. The uses of AI span from simple activities like route-finding to complex ones like self-driving cars. For example, Google Maps uses AI to provide users with the fastest route, while Tesla Autopilot is replacing human drivers in vehicles. The introduction of Expert Systems (ES) in real-life problems marked the beginning of AI applications decades ago. Since then, more advanced intelligent systems have been developed, differing from ES systems in various ways.

While AI has come a long way since the introduction of ES decades ago, there are growing concerns about the application of machine learning models in industries such as healthcare, law enforcement, energy management, cybersecurity, etc. These models can be difficult to interpret, making it challenging for stakeholders and end-users to understand how the models are making predictions and what factors are driving those predictions. One key motivation for addressing this challenge is to improve the interpretability of machine learning models. By developing a rule-based explainer using an associative classifier, it can be possible to provide more transparent and understandable explanations for model predictions. This, in turn, can help build trust and understanding among stakeholders and end-users, particularly in industries where AI has a significant impact on the daily lives of individuals.

Shabestari introduces Black-box Association Rule-Based Explanations (BARBE) in his thesis [1]. BARBE can explain the decisions of any black-box classifier on tabular datasets with a high level of precision. In this thesis, we aim to extend the BARBE framework for text datasets. Our primary motivation in this thesis is to utilize BARBE to generate rules that can explain the prediction of machine learning models for text. BARBE consists of two primary elements: a data augmentation method that produces synthetic datasets to be labeled by the black-box, and an associative classifier that generates rules from these labeled data. BARBE uses SigDirect [2] as the underlying associative classifier. One limitation of SigDirect is its inability to process high dimensional datasets having a large feature vector space, as it requires high memory and long run time. We are interested in overcoming this limitation of BARBE with SigDirect by replacing the underlying associative classifier SigDirect with an ensemble approach proposed by the authors in [3] that utilizes a set of base learners where each base learner is an associative classifier and finally combining the rules generated from each base learner to formulate the final set of rules as the explanation. We call it BARBE with CFAR which has the ability to efficiently handle high-dimensional datasets with large feature vector spaces by distributing the feature space among base learners, each of which is trained on a subset of the feature vector space. We investigate the effectiveness of BARBE with SigDirect and BARBE with CFAR for different black-box classifiers in binary and multiclass text classification tasks. We also explore a suitable data augmentation technique for BARBE that can effectively augment the dataset without increasing its dimensionality.

One key aspect of developing BARBE is to generate explanations in the form of rules which are more satisfactory and meaningful to explain the decision made by the black-box models when compared with explainers such as LIME [4], Anchor [5], and SHAP [6]. Through a comparative analysis, we aim to demonstrate that our method is capable of providing more accurate and meaningful explanations than these existing explainers. By developing a more effective rule-based explainer, we hope to provide a valuable tool for researchers, practitioners, and end-users in various domains where the interpretability of machine learning models is critical.

The key motivation points of this thesis are:

- To investigate data augmentation techniques to identify a suitable method for BARBE that can effectively augment the training dataset without increasing its dimensionality.
- To extend the BARBE for text so that it can generate rules that can explain the prediction of machine learning models for text datasets.
- To overcome the limitations of SigDirect, the underlying associative classifier used in BARBE, by replacing it with an ensemble approach that utilizes a set of base learners, each of which is trained on a subset of the feature vector space.
- To demonstrate through a comparative analysis that BARBE is capable of providing more meaningful explanations than existing explainers such as LIME, Anchor, and SHAP.
- To apply BARBE for the explanation of cyberbullying detection in the context of resource-constraint language.

1.2 Explaining Black-box Systems

Black-box models are complex machine learning models that are difficult to interpret, and their lack of transparency can be a significant limitation. The term “black-box” originates from the fact that the internal workings of the model cannot be easily understandable in terms of generating the inference. This lack of transparency can be a significant disadvantage in many applications as the reasoning behind its decisions is vague. Because of this, many users might decide to remain with straightforward, transparent models that can offer a clear justification for their decisions. Model complexity and accuracy, nevertheless, sometimes trade off since more complicated models can frequently provide better accuracy at the expense of interpretability. Even if it means compromising some accuracy, being able to explain the thinking behind a model's judgments may be crucial in some fields, such as healthcare, finance, or criminal justice.

In an effort to address this shortcoming, researchers have been looking for ways to explain how these models infer. Humans can have more faith in the model's output and can more readily spot faults or biases in the system by knowing the elements that contributed to a certain decision. This can help to ensure that the model is making fair and accurate decisions and can also help to build trust between the model and its users. "Model-dependent" explainers [7], [8] are a type of explainability method that relies on the model's architecture or internal workings of the machine learning model in order to generate explanations. The major dependency here is to achieve understandability regarding how the model functions and the factors that it considers when making a prediction. If the model architecture needs to be modified, or a different model is chosen for a specific task, the explainer needs to be changed.

Conversely, "Model-agnostic" explanations are techniques that can be used to provide insights into how a machine learning model makes its predictions, without requiring a deep understanding of the model's architecture or inner workings. These methods operate by examining the model's inputs and outputs to determine which features are most crucial for a particular prediction. In general, model-agnostic explainers can work for any model e.g. a DNN model based on Long Short Term Memory (LSTM) [9] or a Support Vector Machine (SVM) that uses a specific kernel [10] to explain the decision. Since they do not need to employ a separate explainer for each unique classifier, these explainers also provide users the opportunity to get familiar with a certain kind of explanation. A technique called "Permutation feature importance", introduced by Breiman [11] involves randomly permuting the values of each feature in the input data and measuring how much the model's output changes as a result. The features that cause the largest changes in the output are deemed the most important. Ribeiro et al. [4] introduce the concept of local surrogate models that involves training a model on a subset of the data, and using this model to explain the predictions made by the original model. Their published method called Local Interpretable Model-agnostic Explanations, known as LIME [4] is one of the state-of-the-art explainers that can explain any model, regardless of its underlying architecture or internal workings. The authors seek to substantiate their assertion in their paper using a few well-known datasets. Another popular model-agnostic explainer is SHAP [6], a method for computing the Shapley value, a concept from cooperative game theory, to assign an importance score to each input feature based on how much it contributes to the model's output. In our thesis, we will thoroughly examine these explainers as we compare the performance of our proposed architecture with them.

1.3 Thesis Statement

Our proposal is to construct a post-hoc explainer with the aid of an associative classifier that is a representation of a true model-agnostic approach using rule-based techniques. Earlier, it was achieved for tabular data [1]. The focal point of our study is to investigate whether this explainer can effectively generate explanations for text datasets. We aim to explore how the associative classifier can be utilized to clarify the operation of a black-box that classifies sentences.

The implementation of Shabestari [1] for tabular data involves taking a tabular record and producing a perturbation in the immediate vicinity of the original instance. However, when it comes to sentences, the method for perturbation is different. The perturbation technique takes a sentence and creates many copies around it, close enough to ask the black-box to label. Data augmentation (DA) technique can be employed to achieve this objective. There are several methods available for carry-ing out DA. Nevertheless, a problem arises when DA produces sentences that include words that were not originally present in the sentence, which is not meaningful for an eventual explanation. As a result, a suitable method can be Easy Data Augmentation (EDA) [12], which drops random words from the sentence without increasing dimensionality. Once the perturbed sentences are ready along with their labels, the associative classifier generates rules to explain the label.

The associative classifier implemented in [1] is not capable of handling high-dimensional datasets with large feature vector space due to its requirement of high memory and long run time. To address this limitation, we propose to replace the associative classifier with an ensemble approach [3] that employs a set of base learners as associative classifiers. The rules generated from each base learner are then combined to form the final set of rules, which serves as the explanation.

We also demonstrate the efficacy of our proposed solution for detecting cyberbullying in the context of resource-constraint language.

1.4 Thesis Contribution

This manuscript contains two major contributions to the field of XAI. First, we introduce Black-box Association Rule-Based Explanations (BARBE) for text. Regardless of the black-box model being used, BARBE uses association rules to provide explanations for text datasets. We demonstrate that the explanation generated by BARBE for text outperforms other explainers such as LIME [4], Anchor [5], and SHAP [6]. Our study introduces two distinct versions of BARBE, which have been developed to generate effective explanations for sentences of varying lengths in both binary and multiclass classification tasks.

BARBE explains the decisions of a black-box model in a more human-understandable way. It presents the most important features that contribute to the decision of the black-box model. The features are represented in the form of rules. These rules are easier for humans to understand, as they provide a set of conditions in order for the model to make a particular decision. These rules are called association rules that take into account the associations among different features. As an illustration, suppose a black-box model is assigned the task of determining whether a given sentence conveys positive or negative sentiment. In this case, association rules may consider the words contained within the sentence that are contributing to its positive or negative sentiment.

Furthermore, we propose an explanation of cyberbullying detection using BARBE in a resource-constraint language. It is one of the applications of BARBE for text. The existing research on machine learning models that predict whether a text is a bully or not lacks human-understandable information about the reasons for their predictions. Deep-learning models are often black-box classifiers that may learn irrelevant patterns in the data, making it difficult to understand the underlying reasons for predictions. To address this issue, we have applied BARBE to justify cyberbullying detection in resource-constraint language to better understand the features that contribute to the models' prediction.

1.5 Thesis Outline

Chapter 2 begins by providing how interpretability and explainability are defined and differentiated. We review some of the fundamental concepts in XAI in this Chapter, and do some background study. We discuss how EDA helps generate synthetic datasets.

Chapter 3 explores the rule-based classifiers commonly found in the literature, with a particular focus on the association rule-based classifier. Moving forward, we discuss the utilization of SigDirect, an associative classifier employed in BARBE. We also provide the motivation for using SigDirect. Furthermore, we discuss the limitations of SigDirect and introduce CFAR which utilizes an ensemble of associative classifiers.

Chapter 4 presents BARBE for text, where the methodology is discussed in detail. Experimental analysis of BARBE is conducted in Chapter 5 and compared with

LIME, Anchor, and SHAP. We discuss the shortcomings of the other explainers and demonstrate how BARBE outperforms them.

In Chapter 6, we present the application of BARBE when detecting cyberbullying in a resource-constraint language.

In Chapter 7, we present an evaluation to demonstrate the effectiveness of BARBE in generating an explanation that is highly aligned with human intuition.

Finally, in Chapter 8, we finish this work by presenting our conclusions and suggesting potential research opportunities.

Chapter 2

Background

This chapter provides a basic understanding of explainable AI (XAI), its definitions, why it is needed, its importance, and its applications. We discuss the interpretable and explainable systems after that. Our proposed method called BARBE uses a simple data augmentation strategy to generate augmented texts around the provided instance. As a result, we include some data augmentation-related research works in this chapter. Our spotlighting of data augmentation-related studies will provide a brief comprehensive overview of the techniques in machine learning. This chapter also covers the most recent post-hoc explainers since we compare them in our comparison research for the experimental analysis part. We provide some studies regarding the efficacy of explainers in the field of cyberbullying detection in this chapter as well since our manuscript contains the application of BARBE for generating explanations to detect cyberbullying in the context of resource-constrained Language.

2.1 Explainable Artificial Intelligence (XAI)

The term “explainable artificial intelligence,” or XAI, refers to the use of strategies and methods to make AI systems more transparent and interpretable. Understanding how AI algorithms generate judgments or predictions with the use of XAI approaches helps increase the accountability, transparency, and trustworthiness of AI systems. Especially in delicate or crucial applications like healthcare [13], banking [14], and legal systems [15], the aim of XAI is to give meaningful explanations that can help people comprehend the judgments made by AI systems. XAI techniques are intended to help humans understand the inner workings of AI models, including how they make decisions and predictions, and how they learn from data.

It can be tricky to detect how advanced machine learning algorithms, such as deep learning models, arrive at their predictions or choices since many of these algorithms are super complicated and difficult to interpret. For example, Zhao et al. [16] discuss the importance of interpretability in artificial intelligence systems, particularly for Visual Question-Answering (VQA) models. To process resumes of job applicants, Amazon used an AI-based recruiting system, although it was later shown that the algorithm was biased towards male candidates. The product development team was disbanded as a result of this serious problem [17]. The problem may have been identified and prevented, resulting in the production of a superior product, if an explanation tool had been employed earlier in the development process. Moreover, the company also suffered damage from being labeled as biased towards a particular gender. Such lack of transparency and interpretability can limit the adoption of AI in critical applications such as healthcare [13], banking [14], and legal systems [15], where decisions can have significant impacts on people's lives.

The General Data Protection Regulation (GDPR), which regulates the privacy of user data, was established by the European Union (EU) in 2018 [18]. The primary goal of the law is to guarantee that businesses that interact with the data of EU citizens secure it. In addition to data privacy, the law also includes a provision for the right to explanation, which requires companies to provide justifications to their clients when automated systems make decisions that have legal implications. Moreover, U.S. Equal Credit Opportunity Act [19] mandates that credit companies provide reasons when a person is denied credit. In 2016, the Defense Advanced Research Projects Agency (DARPA) launched a program called DARPA XAI [20], with the aim of creating AI systems that could be explained to humans. The program aimed to develop

AI systems that were transparent and could be understood by people, unlike some existing black-box AI systems that are difficult to interpret.

2.2 Interpretable and Explainable Systems

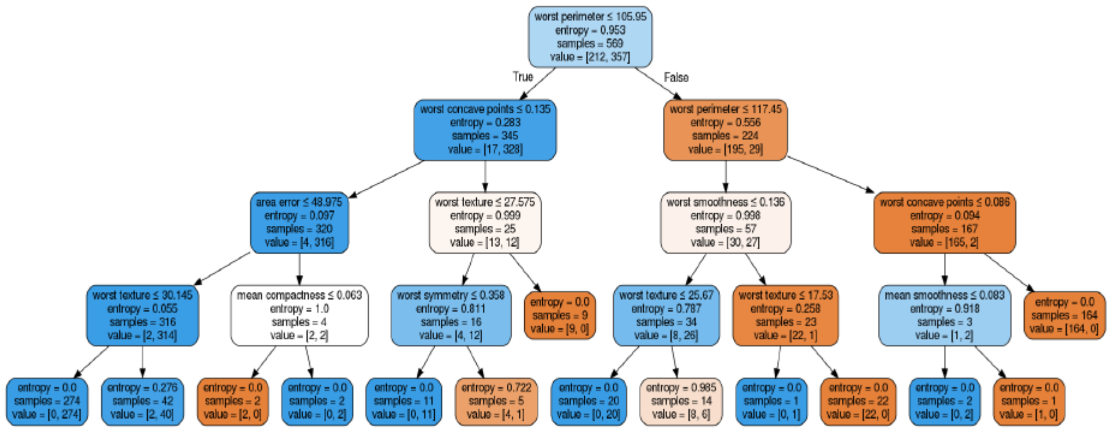
We review some interpretable and explainable systems in this section.

2.2.1 Interpretable Systems

In our investigation of interpretable systems, we are exploring the studies that focus on making systems fully transparent and understandable to humans. Transparent models can be divided into three categories: 1. Generalised Linear Models, 2. Decision Trees, and 3. Rule-based models according to [21]. In addition, it is important to note that even for interpretable classifiers, certain conditions must be satisfied to guarantee their transparency and interpretability. These conditions may include using certain methods or strategies while creating the classifier.

Interpretable models, such as generalized linear models [22], are a popular class of models that are often used in various applications. These models include well-known classifiers like linear regression and logistic regression. Linear regression aims to identify a linear relationship between a dependent variable and one or more independent variables. In logistic regression, the dependent variable is binary, and a logistic function is employed to map the log-odds to a probability value for the dependent variable. As a result, the model is able to produce good predictions depending on the likelihood of a certain occurrence. Both approaches can use the weights of features, either directly or indirectly to interpret a system.

Another type of transparent model is the decision tree. In contrast to linear models, decision trees employ a process of dividing the space into multiple distinct parts. At each node in the tree, the current subspace is further divided into smaller segments, creating smaller clusters of the population. Each path in the tree corresponds to a unique partition in the domain space, based on the value of at least one feature that differs from other partitions. Various algorithms have been proposed over the years to create such trees, with C4.5 [23] being one of the most popular and widely used decision tree algorithms. During training, C4.5 employs the difference of entropy as the criterion to determine which feature to use when expanding a node. At test time, the label of the leaf node of the path that applies to the instance is used to make predictions. The path and the features used along the path can be presented to users as an explanation for why and how the model has arrived at its conclusion. Additionally, the order of features present in internal nodes can provide insights about the system, with features occurring in the upper layers of the tree indicating high global importance, while those in lower layers suggest local importance within a subspace. Figure 2.1 presents the diagram of a decision tree distinguishing benign and malignant breast tumors.



worst perimeter ≤ 105.95 entropy = 0.953 samples = 569 value = [212, 357]
True
False
worst concave points ≤ 0.135 entropy = 0.283 samples = 345 value = [17, 328]
worst perimeter ≤ 117.45 entropy = 0.556 samples = 224 value = [195, 29]
area error ≤ 48.975 entropy = 0.097 samples = 320 value = [4, 316]
worst texture ≤ 27.575 entropy = 0.999 samples = 25 value = [13, 12]
worst smoothness ≤ 0.136 entropy = 0.998 samples = 57 value = [30, 27]
worst concave points ≤ 0.086 entropy = 0.094 samples = 167 value = [165, 2]
worst texture ≤ 30.145 entropy = 0.055 samples = 316 value = [2, 314]
mean compactness ≤ 0.063 entropy = 1.0 samples = 4 value = [2, 2]
worst symmetry ≤ 0.358 entropy = 0.811 samples = 16 value = [4, 12]
entropy = 0.0 samples = 9 value = [9, 0]
worst texture ≤ 25.67 entropy = 0.787 samples = 34 value = [8, 26]
worst texture ≤ 17.53 entropy = 0.258 samples = 23 value = [22, 1]
mean smoothness ≤ 0.083 entropy = 0.918 samples = 3 value = [1, 2]
entropy = 0.0 samples = 164 value = [164, 0]
entropy = 0.0 samples = 274 value = [0, 274]
entropy = 0.276 samples = 42 value = [2, 40]
entropy = 0.0 samples = 2 value = [2, 0]
entropy = 0.0 samples = 2 value = [0, 2]
entropy = 0.0 samples = 11 value = [0, 11]
entropy = 0.722 samples = 5 value = [4, 1]
entropy = 0.0 samples = 20 value = [0, 20]
entropy = 0.985 samples = 14 value = [8, 6]
entropy = 0.0 samples = 1 value = [0, 1]
entropy = 0.0 samples = 22 value = [22, 0]
entropy = 0.0 samples = 2 value = [0, 2]
entropy = 0.0 samples = 1 value = [1, 0]

Figure 2.1: A decision tree taken from [1] distinguishing benign and malignant breast tumors

Researchers have identified rule-based models as another type of transparent model, alongside linear models and decision trees. Rule-based models generate a set of patterns that frequently occur in the training set and apply these patterns to classify new instances. They offer more than one explanation for a prediction. Each rule consists of antecedent items and a consequent class label, with the final class determined by a specific criterion, such as confidence scores. To interpret rule-based models, users can examine the rules that apply to a specific instance, providing multiple explanations as each rule is an explanation. Chapter 3 discusses this type of model in more detail.

2.2.2 Explainable Systems

Explainable systems are machine learning models that are tailored to provide explanations for their decision-making processes. which can be important for ensuring fairness, transparency, and accountability in decision-making processes. These explanations assist users in comprehending the reasoning behind the inference of a machine learning model, which can be important for ensuring fairness, transparency, and accountability in decision-making processes. Model-specific explainers are a class of explainable systems that provide explanations for the decisions made by a specific machine learning model. These explainers are designed to assist users to understand the internal workings of the model and how it arrives at its predictions. They are trained on the same data as the underlying model, making them effective at providing meaningful explanations. The explanations provided by model-specific explainers can vary depending on the type of model being used. For example, for a decision tree model, the explainer might show the path that the model took through the tree to arrive at a particular decision. For a neural network, the explainer might highlight the features that are most influential in the model's decision-making process.

Another type of explainable system is called a model-agnostic explainer where our research focus is on. As black-box models don't have any explainability features, these systems are added to them to generate explanations for them. These two types of explainers are also known as model-dependent and model-independent explainers, respectively. It is important to note that the dependency or specificity referred to is related to the ability to access the internal workings of the machine learning model. Because model-agnostic explainers do not rely on the internal workings of a specific machine learning model, they can be used to provide explanations for a wide range of models. This makes them particularly useful in situations where the exact nature of the model is unknown. Another method of categorizing explanation techniques involves grouping models based on their knowledge of the domain [24]. An application-dependent explainer assumes that the user has knowledge of the domain and uses domain-specific vocabulary to provide explanations, such as using medical terminology in medical applications. In contrast, a generic explainer does not possess any domain-specific knowledge and provides explanations that are domain independent.

Model-specific Explainers

Model-specific explainers, also known as model-dependent explainers are created to work with a specific machine learning model architecture. Although the majority of these methods are created to provide explanations for Deep Neural Networks (DNNs), there are some techniques that have been developed for explaining other types of classifiers as well. Tree ensembles are a collection of classifiers that utilize multiple decision trees to improve accuracy compared to a single tree, making them a preferred choice for achieving high accuracy scores. However, this improvement in accuracy often results in a lack of interpretability in ensemble models. If accuracy is the primary concern in a task, and understanding how the model arrived at its decision is a secondary concern, then using ensemble methods is reasonable. Moore et al. [25] present a technique for generating explanations for individual instances in tree ensemble models like random forests. Their approach involves calculating the expected output change for each node in every tree of the ensemble model and utilizing these changes to generate an explanation for a specific data point. The explanation produced by their system includes an importance score for each feature present in the data point, which is the sum of the prediction output change for all the corresponding nodes along the paths used to determine the class label of the instance.

Over the past few years, there has been a significant research focus on explaining DNNs. This is largely due to the widespread popularity of DNNs and their successful application to various domains, including Natural Language Processing (NLP) for tasks like sentiment analysis [26], as well as image processing for tasks like object localization [27]. As a result, developing explainable systems for DNNs has become increasingly important for promoting transparency and understanding of these models. The attention mechanism [28] is a recent innovation in DNNs that has considerably enhanced their performance, notably in the field of NLP. To address the challenge of vanishing or exploding gradients in sequence-to-sequence tasks, such as Neural Machine Translation, DNN previously relied on the architecture of Long Short-Term Memory (LSTM) [29]. NMT translates the text from one language to another. The model takes a sentence in the source language as input and generates a corresponding sentence in the target language as output. NMT is based on DNN and uses an encoder-decoder architecture to perform the translation. In NMT, the decoder in an LSTM-based model faces the challenge of generating the first word in the target sentence using only the information in the state vector of the last word in the source sentence. This limitation often made the translation task more difficult. However, the attention mechanism has enabled the model to take advantage of the weighted sum of the state vectors of source words, including the information corresponding to the first word in the target language, which usually lies at the beginning of the source sentence. The assigned weights indicate which source words the model should pay more attention to, resulting in better performance in various natural language processing tasks, including NMT [29].

To illustrate an example of using the attention mechanism, Choi et al. [30] proposes a machine learning model called Retain, which utilizes a reverse time attention mechanism for predicting medical outcomes. The authors employ a DNN model to analyze Electronic Health Records (EHR) in reverse chronological order to predict heart failure diagnosis. The EHR data is treated as a time series dataset, and two attention mechanisms are used to process the records. They utilize the weights from the attention layers to offer explanations, such as considering the physician visits that led to a significant impact on the model's decision. For example, the record with the highest result after multiplying its attention weights by the corresponding embedding weights is deemed the most important.

In the context of deep-learned models, there are multiple methods for gaining insight into how a model makes its predictions. One such approach is to use the attention mechanism, as mentioned previously. Researchers have focused on explaining DNN using methods where the internals of the model are exposed to the explanation module [31], [32]. Another method involves computing gradients. This approach entails calculating a gradient with respect to the predicted class and using the back-propagation algorithm to propagate this gradient back to the input. This process allows the input to be combined with the gradient to identify the important pixels that contribute most significantly to the predicted class. These important pixels can be used to generate explanations for the model's decision-making process (e.g., Grad-CAM [33]).

Model-agnostic Explainers

In this section, we review the explainers that assert their capability to explain the decisions independent of the model architecture. Model-agnostic explainers have no dependency on the internal working of the model. Nevertheless, it is important to highlight that obtaining accurate explanations is a significant obstacle because the individual providing the explanation lacks the ability to access the internal workings of the system. In contrast, a model-specific explainer has the capacity to access all of its components. We will review some state-of-the-art model-agnostic explainers in this section.

Local Interpretable Model-agnostic Explanations (LIME) The research community has enthusiastically welcome LIME, introduced by Ribeiro et al. [4] is one of the earliest XAI frameworks claimed to be model-agnostic. LIME is available as a

Python package. Because of its availability and ease to use, researchers and developers have begun implementing it readily to incorporate explainability into their systems. As an illustration, the classification of biologically generated fuel compounds was examined by Whitmore et al. [34] using LIME on 2-D chemical structures. Mishra et al. [35] show the application of LIME for music content analysis. Modarres et al. [36] demonstrate the efficacy of LIME in credit lending. LIME has been employed in NLP endeavors, such as Named-Entity Recognition (NER), which involves sequence-to-sequence tagging. [37] discusses the importance of explainability in sequence tagging models for NER and explores the use of several techniques to achieve it, including saliency maps and layer-wise relevance propagation. The authors of [37] suggest two distinct methodologies for utilizing LIME. In the first approach, they offer explanations for every word in a sentence, whereas, in their second methodology, they attempt to elucidate every named-entity (as a phrase, not an individual term) by utilizing LIME.

As with other post-hoc model-agnostic explainers, LIME must depend on a model's input and output to produce explanations since it assumes that the classifier functions as a black-box. To put it differently, in contrast to transparent models where some portion of the decision tree, if not the entire tree, can be visualized or where certain rules in a rule-based method can be demonstrated, LIME possesses no knowledge regarding the inner workings of the model. The only pieces of information it has access to are the input space and the target predictions. It should be taken into account that the explanations pertain to each individual data point, not the entire model. Our interest is to present the output of LIME for the purpose of explaining text dataset in this manuscript. As an example, for a sentiment classification sentence, the output of LIME would be a set of words or phrases along with their weights, which indicate how important they are for the model's prediction. These words or phrases are usually highlighted in a different color to make them easily visible. If the sentence is "I loved the movie, it was great!", and the model predicts a positive sentiment, LIME may identify the words “loved” and “great” as the most important features that contribute to the model’s decision. The weights associated with these words indicate how much they influence the model’s prediction. In applications that entail classifying natural text, like sentiment analysis, individual tokens are typically transformed into a vector space. LIME method of explanation employs the tokens themselves and produces words as part of the explanation, rather than solely numerical values associated with them. In Figure 2.2, we show the output of LIME for a sentence “I did enjoy as it was an excellent movie”. This sentence has been chosen from IMDB movie review dataset [38] which is widely used for sentiment classification tasks. According to the output of LIME, the words “excellent” and “enjoy” are responsible for making the sentence positive labeled by the block-box. Orange bars represent the positive outcome and the blue bars represent the negative. It is evident from the right side of the Figure 2.2 that the word “excellent” has more weight than the word “enjoy”.

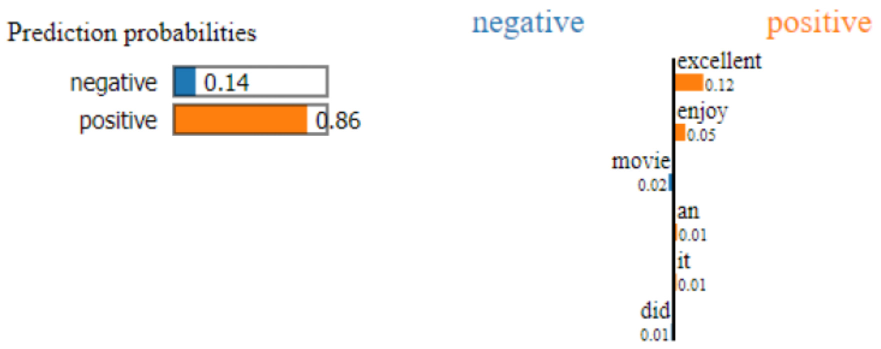


Figure 2.2: Explanation provided by LIME for a sentence “I did enjoy as it was an excellent movie”. This sentence has been picked from IMDB movie review dataset [38]. The probabilities on the left are the prediction probabilities of the underlying black-box model. On the right, the features and their corresponding importance scores generated by LIME are shown in order of their importance. A higher absolute value on the right side bar chart denotes that LIME perceives the feature as having a more substantial impact on the classification outcome.

LIME works by generating a synthetic dataset around the original data point and labeling its instances using a black-box model. This dataset is then used to build an interpretable model for explaining the data point. Our discussion provides a detailed explanation of how LIME generates explanations.

To train a transparent model, LIME requires a new dataset consisting of new samples. These samples are perturbations of the original instance that the system aims to explain to the user. The most commonly used technique in model-agnostic explainers is perturbation, where small changes are made to the original data to create new samples [39]. LIME uses different methods to create samples based on the type of dataset. In the case of text data, LIME generates 5,000 new data points by perturbing the original text, unless the user specifies a different number using a parameter as mentioned in the Python package release by the author. The perturbation technique comprises randomly deleting a word at a time from the original sentence. Essentially, the perturbation process begins by selecting a specific text instance to explain, and then LIME generates several perturbations by randomly removing words from the original text. The number of perturbations can be set by the user or specified as a default value in the LIME package which is set to 5,000 according to the settings in the Python package. LIME then uses the perturbed text samples and their corresponding predicted labels from the black-box model to train an interpretable model. The interpretable model is trained to identify the important words or features that contribute to the prediction of the black-box model. Finally, LIME generates an explanation for the original text instance by highlighting the important words or phrases identified by the interpretable model, thus providing insights into the decision-making process of the black-box model.

After creating the local interpretable model, LIME proceeds to extract and provide explanations to the user based on the features. This is a pretty straight forward step in the LIME algorithm, requiring only the extraction of feature weights from the linear model. The research paper [4] only provides an example with linear models, but LIME can use other interpretable models such as decision trees or rule-based models to extract feature-based explanations. However, the method of extracting feature weights from the model remains the same regardless of the type of interpretable model used.

Although LIME has been widely used in various systems, many researchers, including the authors of LIME, have developed new models to improve certain aspects of LIME. We discuss two models, KLIME [40] and LIME-SUP [41], that use LIME as their core algorithm. We review LORE [42], a method that takes advantage of the Genetic Algorithm to create a neighborhood around the instance for generating explanations. KLIME aims to provide explanations for a limited number of representative points that can be used for the remaining points. KLIME aims to generate explanations for a limited number of representative points that can be used to explain the remaining points in the dataset. Unlike LIME, KLIME distributes the training data into K clusters and for each cluster, trains a Generalised Linear Model (GLM) based on the data points of the cluster. The popular K-means clustering is used to create the clusters, whereas another GLM is trained based on all the training instances of the dataset. KLIME chooses whether to utilize the GLM for the related cluster or the global GLM when explaining a new instance based on the number of training data points in the cluster. If there are 20 or more training data points in the cluster, the GLM for that cluster is used; otherwise, the global GLM is used. It is important to note that the K-means clustering has stability issues and finding the optimal value for the parameter K of K-means clustering impacts the number of clusters generated by KLIME.

Hu et al. [41] propose another method similar to KLIME, called LIME-SUP that employs supervised partitioning to divide the training data, which is different from KLIME's K-means approach. In LIME-SUP, model-based trees are used, where the nodes represent models that apply to specific regions of the problem space, unlike decision trees. As compared to KLIME, this technique produces more accurate local models in the leaves since the space is divided according to the training data. Also, subspaces of LIME-SUP are more stable than K-means subspaces in KLIME. The LIME-SUP technique, on the other hand, requires training labels to create the tree, which may not always be accessible. While LIME-SUP offers an improvement over KLIME, both methods provide faster explanation generation by sacrificing the accuracy of the explanations, which are based on data points in the vicinity of the instance being explained.

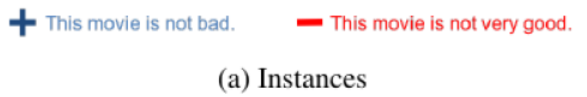
Rules are highly interpretable and an appropriate technique to provide users with explanations as mentioned previously in this chapter. The method called LORE, introduced by Guidotti et al. [42] uses rules as the form of explanation. They use a Genetic Algorithm to create a neighborhood around the instance, with the data point selection algorithm choosing up to half of the data points from the same class as the original data point. While the Genetic Algorithm is used to create data points, class labels are determined by querying the black-box model. They train a decision tree with synthetic data points and utilize it to create two sorts of rules: a single decision rule and a set of counterfactual rules. The decision rule identifies the attributes that contributed the most to the decision made by following the path leading to the original instance in the decision tree. The counterfactual rules provide alternate approaches to changing the decision by amending the input attributes, which are retrieved by traversing all pathways in subtrees starting from nodes along the original instance's path.

Pattern Aided Local Explanation (PALEX) is another method proposed to provide explanations for black-box models. In their method, Jia et al. [43] suggest a set of patterns as the explanation. In their work, these patterns are extracted by taking advantage of contrast sets built in the vicinity of the instance. They rely on the FP-Growth algorithm [44] to build contrast sets. Their method, however, requires defining a few hyper-parameters such as minimum support, and minimum growth ratio thresholds. One noticeable fact we observed in the experiments they report is that different values for the mentioned parameters work best across different datasets. For example, among the five datasets, the best minimum support threshold varies from 0.10 to 0.25 which is significant. Finally, their method requires a probability score provided by the black-box model.

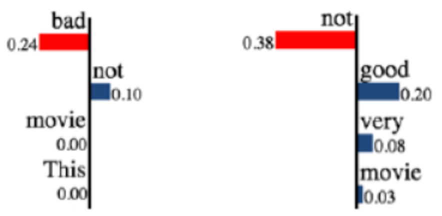
Alternatively, Pastor and Baralis propose LACE [45] as a method that directly learns an associative classifier by exploiting the nearest data points in training data. Their method, however, requires the training data to be available, and this may not always be realistic. Moreover, the percentage of instances in the neighboring data with a different label can significantly change the performance of their method. Additionally, the sparsity of the training data in that neighborhood can have a substantial impact on the performance of their system. Furthermore, this method also can only work on black-box models that provide probability scores. Besides, their associative classifier has many hyper-parameters difficult to set and tune. Finally, they provide no quantitative results on the performance of their system.

Anchor Ribeiro et al. introduce Anchor [5] after LIME by figuring out a weakness of LIME based on the fact that the explanation of LIME is generated from a linear model. The authors demonstrate that assigning an importance score to each feature may not always be an effective method. The text classification example in Figure 2.3 illustrates this situation. For instance, the authors show that the word "not" in a sentence can have both positive and negative influence depending on the other words sitting next to it in the sentence, making it difficult to accurately assign an importance score to this feature alone. An approach like LIME cannot account for such dependencies because it assumes feature independence and produces individual feature importance scores, making it unable to capture the nuanced relationships between features that can affect their impact on the prediction. To address this limitation, Ribeiro et al. [5] propose a set of salient features in the form of "if-then" rules as the explanation since these rules comprehend Anchor to capture complex feature interactions and dependencies that might be ignored by LIME. The authors named the term "Anchors" to describe these rules since they serve as anchor points that guide the prediction toward the target class. This means that modifying the other features would not change the class label, making the rules more reliable and trustworthy for users. In their framework, each anchor is essentially a set of crucial features that contribute significantly to the prediction. For instance, an anchor can be composed of important words in a text such as “not” and “bad.”. They argue that their Anchors approach has high precision, meaning that when an anchor is applied, it is likely to classify most instances that fit the anchor correctly. In other words, the anchor’s definition is precise and specific enough to capture a subset of instances that can be classified with high confidence.

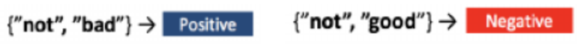
The authors use a greedy algorithm to construct an anchor. A greedy algorithm is a simple heuristic algorithm that follows a specific set of rules to make the best decision at each step. In the case of Anchors, the greedy algorithm aims to construct the most precise and smallest anchor that can capture the most significant features contributing to the prediction. The algorithm considers all possible subsets of features and chooses the one that satisfies the precision and coverage requirements while minimizing the size of the anchor. The greedy algorithm constructs Anchors efficiently while still maintaining a high level of precision. To determine the precision, Anchor uses a synthetic set of instances that are close to the instance being explained. However, unlike LIME, Anchor aims to minimize the number of queries to the black-box model by identifying the smallest set of necessary instances using a multi-armed bandit algorithm. By leveraging this approach, Anchor can adaptively generate the neighborhood around the instance. They stop the greedy approach when the precision exceeds a heuristic criterion, which they set at 0.95 in their study. While Anchors can capture the most important features that contribute to the prediction, it may not provide insight into how these features interact with each other or how their joint effects contribute to the prediction. Thus the association among the features is not expressed by Anchor. As opposed to LIME, the importance score or weight of the features is not exposed by Anchor.


This movie is not bad.
This movie is not very good.

(a) Instances



(b) LIME explanations



(c) Anchor explanations

Figure 2.3: Explanations generated by LIME and Anchor. This figure has been taken from [5]. (a) shows two sentences from the sentiment analysis task. The blue one has a positive sentiment and the red one has a negative sentiment. (b) presents the explanations of LIME, (C) presents the explanations of Anchor

SHAP The SHAP (SHapley Additive exPlanations) [6] framework is built upon the concept of Shapley value [46], a cooperative game theory concept that measures the contribution of each player to the overall outcome of the game. Shapley value provides a fair and efficient way to allocate rewards by considering the marginal contributions of each player in different combinations of players. Shapley value has been widely used in various fields to solve reward distribution problems, and it has also been applied to machine learning models to explain the contributions of each feature to the predicted outcome. In the context of machine learning, the players are the input features, and the outcome is the predicted output. The feature importance can be thought of as the reward a player receives as a participant in the game. The formula used in the SHAP framework to calculate the contribution of each feature to the predicted outcome is as follows:

$$\phi_i(\text{val}) = \sum_{S \subseteq \{x_1,\ldots,x_p\} \setminus \{x_i\}} \frac{|S|!(p-|S|-1)!}{p!} \left(\text{val}\left(S \cup x_i\right) - \text{val}(S)\right) \tag{2.1}$$

This formula computes the feature importance for a given feature i in an instance

x with p features. The formula involves computing a weighted sum over all subsets of x that exclude feature i. For each subset, the formula calculates how adding feature i to that subset affects the final prediction of the model. The weight of each subset is determined by the number of possible ways that subset could have been generated.

The SHAP method, while effective for computing feature importances, can be computationally expensive because it involves computing the weighted sum over all subsets of features. Since the number of subsets grows exponentially with the number of features, this can quickly become infeasible for large datasets. To address this issue, the authors of SHAP propose using a sampling approach to make the computation more efficient. The basic idea behind the sampling approach is to randomly select a subset of subsets to include in the sum. By selecting subsets at random, the hope is that the resulting feature importances will be representative of the true values, while the computation time will be greatly reduced. The authors show that, with appropriate sampling strategies, the resulting feature importances can be very accurate while using a fraction of the computation time required for the full sum. There are many different sampling strategies that can be used with the SHAP method, and the choice of strategy can have a significant impact on the resulting feature importance.

2.3 Data Augmentation

This section provides a brief overview of data augmentation for text, with particular emphasis on simple data augmentation techniques [12] since our proposed method leverages a straightforward data augmentation approach to create additional texts based on the given input. Data augmentation is a set of techniques that aim to enhance the variety of training samples without the need for additional data collection. In recent machine learning research, it has gained considerable interest, with highly acclaimed general-purpose approaches (e.g. UDA [47], AutoAugment [48]). In the context of NLP, data augmentation can be used to increase the size of a corpus of text data, which can help to improve the performance of language models. However, adapting DA techniques for NLP can be more challenging compared to other domains, such as computer vision. This is partly due to the discrete nature of language, where words and sentences are represented by discrete symbols (e.g., individual words or characters) rather than continuous signals. This rules out some techniques commonly used in computer vision, such as adding random noise to images, which are not directly applicable to text data. However, despite the challenges presented by the discrete nature of language, there has been a growing interest in and demand for data augmentation techniques in NLP. This is partly due to the increased availability of large pre-trained language models which have led to significant advances in various NLP tasks, such as text classification, question answering, and language generation. One of the simplest and most efficient data augmentation techniques that have been shown to improve the performance of NLP models on various tasks is EDA proposed by Wei et al. [12]. It is also easy to implement and can be applied to any text dataset without requiring extensive domain knowledge or resources. We describe the core components of EDA in the next section.

2.3.1 Easy Data Augmentation

Famously proposed by Wei et al. [12], Easy Data Augmentation (EDA) uses traditional and very simple data augmentation methods. EDA consists of four simple operations: (synonym replacement, random insertion, random deletion, random swapping) that do a good job of preventing overfitting and helping train more robust models. Inspired by their proposed work, this category also includes techniques that are simple and easy to use. Based on whether the technique requires looking into the context of the sentence, it is again divided into two broad subcategories:

1. Contextual Replacement: One of the easiest and most popular data augmentation techniques is producing a new sentence by replacing n words with similar words while taking into account the context of the sentence. Here similar words refer to synonyms, hyponyms, hypernyms, words with same Part-Of-Speech

(POS) tag etc. Such replacement should be label preserving without changing the meaning of the sentence. This can be done in three ways depending on the task and the level of correctness required.

(a) Thesaurus based: This technique involves finding similar replacement words using thesaurus derived from WordNet, VerbNet etc. Kolomiyets et al. [49] are one of the first ones to implement this technique in order to improve the portability of time expression recognition to non-newswire domains. They replace temporal expression words with potential synonyms from WordNet in order to generate additional training examples. Later, authors like Li et al. [50], Mosolova et al. [51] and Jungiewicz et al. [52] also experiment with word replacements based on thesauri using pre-defined dictionaries for sentiment analysis and toxic comment classification. EDA randomly chooses words from the sentence that do not stop words and replaces them with one of their synonyms chosen at random from WordNet. Zuo et al. [53] use both WordNet and VerbNet to retrieve not only synonyms but also hyperonyms and words of the same category. When it comes to choosing the synonyms, Zhang et al. [54] and Marivate and Sefara [55] look into the geometric distribution by which the insertion of a distant synonym becomes less probable. In order to remove gender bias in datasets, Lu et al. [56] in Counterfactual Data Augmentation (CDA) technique use a bidirectional dictionary of gendered word pairs to replace masculine words with their feminine counterparts and to break the associations between gendered and gender-neutral words. In addition to synonyms, Coulombe et al. [57] also suggest the use of hyponyms and hypernyms to replace the original words in a text. For example: generating a new text 'I have a small animal.' from 'I have a small dog.' In order to ensure that the newly generated texts are syntactically identical, Xiang et al. [58] propose to use

WordNet to find candidates to substitute a word having the same POS tag (Adjective/Adverb, Verb or Noun). Experiments done on eight benchmark datasets used for NLP classification tasks show that the augmented data helps improve the accuracy of deep learning models including state-of-the-art transformer-based models. Jungiewicz et al. [52] have utilized WordNet and Thesaurus.com for finding synonyms for substitution. Their augmented data improve the performance of two of their CNN-based models for text classification. Such thesaurus-based methods, however, suffer from the limitation of the replacement range and word part-of-speech.

(b) Semantic embedding based: To tackle the limitations of thesaurus-based word replacement, a number of works have proposed using embedding replacement of words instead. Such embeddings aim to represent words in a dense vector by making sure that similar words are close to each other in the embedding space. Pre-trained word vectors like as Glove, Word2Vec, FastText, etc. are often used to find words closest to the original word in the vector space. Wang and Yang [59] use this technique to better classify annoying tweets. They replace each original word in the tweet with one of the k-nearest-neighbor words using cosine similarity. Li et al. [60] also replace a word with its top-k nearest neighbors in a context-aware word vector space for generating adversarial examples. Marivate and Sefara [55] and Rizos et al. [61] have used pretrained neural word embeddings for word substitution instead of relying on an external thesaurus [62]. To better classify sentiments of product reviews in Vietnamese language, Huong and Hoang [63] perform word embedding replacement based on the cosine distance for measuring the similarity. Madukwe et al. [64] look into an embedding replacement for improving hate speech detection which often suffers from a lack of diversity and a diminutive class of interest. One of the problems with this however is the fact that antonyms of a keyword could be in the same vicinity as that keyword because they usually co-occur in documents. In order to avoid that, instead of pre-trained Word2Vec, they use counter-fitted (synonym and antonym) word embedding [65] and Levy and Goldberg's [66] skip-gram model (word2vecf2). Also when it comes to choosing the words to be substituted, they use techniques like Particle Swarm Optimization (PSO) [67] and Integrated Gradient (IG) [68]. Their proposed methods of selecting candidate words are superior to the baseline methods on the Founta and Davidson datasets. However, word embedding replacement still suffers from the lack of context when it comes to fetching synonyms, especially for words with multiple meanings and few synonyms.

(c) Language Model (LM) based: To mitigate the limitations of word embeddings and to make use of the context of a sentence, a number of authors have used language models. This is because large pre-trained language models are good at predicting synonyms that are not only similar in meaning but also fit the context in principle. Alzantot et al. [69] in their work utilize the Google 1 billion words language model [70] to choose synonyms that have a high probability of fit. Similarly, Gao et al. [71] compute a weighted average of the embeddings of all possible synonyms predicted by the LMs as the replaced input thinking that the average representations could augment text with richer information. Instead of just relying on synonyms to generate new data, Sosuke Kobayashi [72] makes use of context and replaced words in sentences with other words having paradigmatic relations. As a result, the number of replacement words increases to a great extent which in turn increases the number of augmented sentences. For example: 'the actors are amazing' gets augmented into 'the performances are fantastic', 'the films are fantastic', 'the movies are fantastic', and so on. In order to make sure that the replacement words still preserve the original label and context, they modify a bi-directional language model and make it label-conditional; thus preventing any changes in semantics. Fadaee et al. [73] also propose a similar technique using language models where they slightly modify the source sentence in a way that still preserves the semantics and syntax. As an example, the generated sentence 'My uncle sold his house' from the original sentence 'My uncle sold his motorbike' is valid. However, the generated text 'Alice waters the motorbike' from the original text 'Alice waters the plants' is not semantics preserving and is therefore invalid.

Random Transformation: This category involves simple and easy transformations like random deletion, substitution, swapping, insertion, etc. of characters, words, or sentences in order to produce new data. These techniques usually do not take into account the context of the sentence and are mostly done with the aim of adding perturbations or noises without changing the original label. The newly generated noisy data is usually faint and does not shift the semantics of the text so much that it deviates from the original data or changes human judgement. Most of the time, a combination of these techniques is used at once. Models are often trained on this augmented data in order to increase robustness.

1. Character-level: In order to make NMT models less susceptible to adversarial examples, Belinkov and Bisk [74] add artificial and natural noise to the training data on a character level. This included random switching of single letters (cheese $\rightarrow$ cehese), randomization of the mid part of a word (cheese $\rightarrow$ ceehse), the complete randomization of a word (cheese $\rightarrow$ eseehc) and the replacement of one letter with a neighboring letter on the keyboard (cheese $\rightarrow$ cheeae). Feng et al. [75] followed similar character-level techniques like random deletion, swap, and insertion whilst keeping the first and last letter of every word unchanged in order to fine-tune text generators. Their method was able to perform better than the baseline model in terms of diversity, fluency, semantic context preser-

vation, and sentiment consistency. Karimi et al. [76] propose AEDA which includes random insertion of punctuation marks into the original text as well as changing the position of words in the sentence while keeping the order intact. On five text classification tasks, the model trained on AEDA augmented data outperformed those trained on EDA. Ebrahimi et al. [77] on the other hand generate noisy data by flipping a letter of the input data (one-hot representation) if it increased the loss of an existing model. Upon retraining the model on the additional noisy data, the error rate improves and the success of adversarial attacks is minimized. Li et al. [60] similarly propose 'TextBugger' which generates label-preserving adversarial examples by inserting bugs into texts by swapping two adjacent characters, replacing characters with visually similar characters ('O' with '0'), deleting a random character or inserting a space into a word.

2. Word-level: Wei et al. [12] in their famously proposed EDA technique use a combination of synonym replacement, random deletion, insertion, and swap on a word level and show that in low resource settings which are able to improve model performance in five text classification tasks. Miao et al. [78] have looked into a number of random local modifications including removing words in the sentence with some probability p as well as inserting words in random positions in a sentence for boosting the performance of language models for semi-supervised opinion mining. Rastogi et al. [**rastogi2020can**] also have experimented with random deletion, swapping, and substitution of words and achieved a performance boost in the toxic comments classification task. Others have looked into replacing non-important words with random words (Niu and Bansal [79]) and randomly swapping any two words in a sentence (Artetxe et al. [80], Lample et al. [81]) for a number of tasks like text classification, sequence labeling and so on. Although the semantics of a language depends largely on

the order of the text information, a slight change in order is still detectable by humans [82]. With this intuition, Dai et al. [83] first split token sequences into segments according to labels, then randomly chose some segments to shuffle the order of the tokens inside, with the label order unchanged. They also combine it with other techniques like label-wise token replacement and mention replacement using some binomial distribution and reported improvement in model performance for NER tasks, especially for small datasets. To make models more robust to common spelling mistakes, Coulombe et al. [57] and Regina et al. [84] introduce a list of the most common English misspellings. For example, replacing "across" as "accross" to generate an augmented text containing a misspelling. In order to improve generalization, Xie et al. [85] introduce 'blank noising' which replaces random words with '_'. Wei et al. [86] leverage curriculum learning by first training models on only original examples and then introducing augmented data (using EDA) as training progresses for improving few-shot text classification. On four diverse text classification tasks, they also found that common data augmentation techniques can improve the performance of triplet networks by up to 3.0% on average.

Instead of manipulating words, some authors have looked into manipulating slot values. Peng et al. [87] augment input dialogue acts by either deleting, replacing, or inserting slot values to obtain more combinations in the task of spoken language understanding. Song et al. [88] similarly augment datasets by copying user utterances and replacing real slot values with randomly generated strings. The technique proves to be effective for copy-mechanism models in dialogue state tracking in three widely used datasets (WoZ 2.0, DSTC2, and Multi-WoZ 2.0).

3. Sentence-level: Yan et al. [89] perform random deletion, insertion, and shuffling of sentences in the legal documents dataset. Random deletion of sentences

from a legal document is done according to a certain probability with the intuition that deleting irrelevant statements will not affect the understanding of the legal case. Moreover, since documents with the same label may have similar sentences, they randomly select sentences from other legal documents with the same label and insert them to get augmented data. They also perform random swapping of sentences between legal documents thinking that since sentences independently contain relatively complete semantics compared to words, the sentence order in the legal document would have little effect on the meaning of the original text. Yu et al. [90] employ an attention mechanism for both word-level and sentence-level random deletion in their proposed hierarchical data augmentation (HDA) technique for text classification.

Chapter 3

Association Rule-Based Classifiers

Rule-based classifiers are an important category of transparent classifiers. During the prediction phase, these classifiers typically utilize a collection of rules developed during the training phase. Association rule classification is a technique that involves applying pattern mining to the classification task. In essence, frequent itemsets that are linked with a particular class label are transformed into rules that characterize that class label. The rules are therefore conjunctions of feature-values implying a class label: f_1, and f_2, and f_3, and f_4, and ..., $f_n \rightarrow class1$.

In this chapter, we begin by briefly examining various types of rule-based classifiers before delving into the notion of association rules and association rule-based classifiers. In Section 3.3, we provide a detailed overview of a particular association rule classifier called SigDirect [2], which we use in our experiments. We evaluate the effectiveness of our SigDirect classifier implementation using text dataset and compare the explanations with other state-of-the-start explainers in the experiment section. Finally, we discuss the limitations of SigDirect and introduce Classification by Frequent Association Rules (CFAR), a method that utilizes an ensemble of associative classifiers.

3.1 Rule-based classifiers

Essentially, rule-based classifiers function similarly to ”if” statements in computer programming languages, where a statement is executed only if a specific condition is met. Each rule in rule-based classifiers consists of two primary components: a set of items on the left-hand side (LHS) and an item on the right-hand side (RHS), which represents the class label in classification tasks. Alternatively, these components can be referred to as antecedent and consequent, respectively. Additionally, some classifiers might include additional information to each rule, such as the frequency of the LHS in the dataset (support score) or the frequency of the LHS co-occurring with the RHS in the dataset (confidence score). One of the popular rule-based methods is OneR [91]. The algorithm generates a single rule for each value of each feature using one feature in the antecedent and the label in the consequent. For every rule, the classifier retains the number of mistakes it committed on the training set, which it uses during the testing phase. At test time, the algorithm chooses the rule with the lowest error score among the relevant rules and returns the label associated with it as the final class label.

In contrast to OneR, which only uses one feature in the antecedent, the First Order Inductive Learner (FOIL) [92] incorporates multiple features in the antecedent. This classifier, which relies on First-Order Logic, generates rules using separate-and-conquer method. During each iteration, the algorithm constructs a new rule by adding new feature-values to it via a top-down greedy approach. The selection of the subsequent feature-value to be added to the rule is determined by the Foil-gain metric, which aims to include as many positive examples as possible under the rule while reducing negative examples. Once the new rule has been established, all instances that the rule applies to are eliminated from the present dataset, and the next iteration of the algorithm commences.

RIPPER is another prominent rule-based classifier that operates on multi-class datasets by utilizing a technique called Repeated Incremental Pruning to Produce Error Reduction [93]. Like FOIL, this classifier takes advantage of the separate-and-conquer approach to generate rules in an iterative manner. However, this rule learner partitions the data into growing and pruning sets. Using FOIL's gain metric, it creates a new rule based on the growing set and then utilizes the pruning set to eliminate some of the feature-value literals that were added during construction, only if the removal contributes to producing a better rule. It is worth mentioning that RIPPER introduces an additional post-processing stage to enhance the quality of the generated rules.

3.2 Association Rule-Based Classifiers

Association rules are specific types of rules that leverage the associations between features. They are mainly utilized in pattern mining, where they help extract frequent itemsets from transactional datasets. The Apriori algorithm [94] and FP-Growth [95] are two of the most notable algorithms employed to extract these frequent patterns from datasets. In this section, we are going to review these algorithms. Association rule mining aims to identify relationships between items in a transactional dataset. For instance, given a dataset that records transactions from a grocery store, the primary objective is to discover frequent itemsets like {"p1', "p2', "p3', and "p4'}. Using the frequency of each item, we can create association rules, such as "p1", "p2" → "p3", "p4" or "p1", "p2" → "p3". There are no restrictions on items in the RHS itemset.

The Apriori algorithm is a classic algorithm for association rule mining. It is primarily used to extract frequent itemsets from large datasets. The technique operates by periodically scanning the dataset for frequent itemsets of increasing size. During each iteration, the algorithm applies a minimum support threshold to prune itemsets that are less frequent than the specified threshold. This approach is known as the "Apriori principle," which states that if an itemset is frequent, then all of its subsets must also be frequent. The Apriori method may prune a huge part of the search area using this approach, making it more efficient than other algorithms that scan the full dataset. It is composed of two main phases: the frequent itemset generation phase and the rule generation phase. During the frequent itemset creation phase, the algorithm scans the dataset for frequent itemsets of various sizes. During the rule generation phase, the algorithm constructs association rules from frequent itemsets that meet a minimum confidence threshold. The confidence threshold specifies the minimum level of certainty required for an association rule to be considered significant.

FP-Growth (Frequent Pattern Growth) is a popular algorithm for mining frequent itemsets in large datasets. It was introduced as an improvement over the Apriori algorithm, which can be inefficient when dealing with large datasets due to its requirement of multiple passes over the data. The FP-Growth algorithm operates by constructing an FP-Tree, which is a tree-like structure that provides a compressed representation of the dataset. This structure enables faster generation of frequent itemsets. To construct the FP-Tree, the algorithm initially scans the dataset to determine the frequency of each item. It then recursively adds frequent items to the tree, commencing with the most frequent item.

CBA (Classification Based on Associations) [96] is a classifier that resembles the Apriori algorithm in the manner it extracts Classification Association Rules (CARs) from data. In contrast to regular association rules, which can contain any number and type of items in the consequent of a rule, the consequent of a CAR is constrained to only one item, which must correspond to the class label. After the rules have been generated by the algorithm, they are initially arranged based on their confidence and support. The algorithm then implements the concept of database coverage and preserves only those rules that cover at least one instance of the dataset if no previous rule has covered it. Unseen instances are classified using the first applicable rule in the aforementioned list.

CMAR (Classification based on Multiple Association Rules) [97] is another approach that utilizes FP-Growth instead of Apriori to generate rules. The algorithm applies a pruning strategy similar to CBA once it has generated all the rules. To classify each instance, the learner applies a particular measure to determine the appropriate class label.

CPAR (Classification based on Predictive Association Rules) [98] is a hybrid approach that combines traditional rule-based methods with association rule mining techniques. This algorithm builds upon the FOIL algorithm mentioned earlier and utilizes the FOIL gain metric to determine the most suitable feature in the algorithm. A significant contrast between CPAR and FOIL is that CPAR assigns a significance score to each instance. Instead of deleting samples covered by a created rule, a weighted score is used to lower their importance.

3.3 SigDirect

In this section, we spotlight on SigDirect, a classifier introduced by Li and Zaiane [2] known as Statistically Significant Dependent Classification Association Rules for Classification. SigDirect is at the core of the BARBE. We discuss the concept of statistical significance in this section along with the reason to select SigDirect to explicate the reasoning behind the decisions made by black-box systems.

SigDirect differs from other techniques based on the fact that it makes use of the concept of Statistical Significance to eliminate untrustworthy rules while other approaches rely on a threshold for support and confidence values to discard such rules. The Statistical Significance test can be used to ensure that the results of an experiment are not the result of errors such as sampling errors. When performing the Statistical Significance test, we start by assuming a null hypothesis, which implies that there is no association between the measured variables, and any observed outcome is purely by chance. Our aim is to demonstrate that this hypothesis is improbable. To be more precise, once we have selected a threshold α, which depends on the particular field of study, we can determine if the outcome of an experiment is statistically significant by checking if the p-value (i.e., the probability of obtaining results as extreme as those observed in a null hypothesis) is lower than alpha. For most scientific research, α is set to 0.05. The formula, known as Fisher's exact test to compute the p-value for a classification association rule presented in the form of $X \to c_k$ is:

$$p\left(X \to c_k\right) = \sum_{i=0}^{\min\{\sigma(X,\neg c_k),\sigma(\neg X,c_k)\}} \frac{\begin{pmatrix} \sigma(X) \\ \sigma\left(X, c_k\right) + i \end{pmatrix} \begin{pmatrix} \sigma(\neg X) \\ \sigma\left(\neg X, \neg c_k\right) + i \end{pmatrix}}{\begin{pmatrix} |D| \\ \sigma\left(c_k\right) \end{pmatrix}} \tag{3.1}$$

Although it is computationally expensive to compute such value, Hamalainen et al. [99] propose a method to efficiently compute the lower bound for p-value when $\sigma\left(c_k\right) \geq \sigma\left(X\right)$:

$$p\left(X \to c_k\right) \geq \frac{\sigma(\neg X)!\sigma\left(c_k\right)!}{|D|!\left(\sigma\left(c_k\right) - \sigma(X)\right)!} \tag{3.2}$$

By employing this equation, we can calculate the lower bound for the rule. If the calculated lower bound exceeds the threshold value α, there is no further need to compute the exact p-value, as the rule is already not statistically significant.

SigDirect employs a strategy similar to Apriori for rule generation. Subsequently, it uses an instance-based approach for pruning to retain only high-quality rules while discarding the rest. Like Apriori, SigDirect leverages the expansion of the kth level to construct the $k+1$th level by utilizing the training set.

During the kth iteration, every node undergoes the following evaluation process: First, the lower bound of the p-value is calculated using Equation 3.2. If the resulting lower bound is less than the statistical significance threshold α set by the user, then Equation 3.1 is employed to derive the exact p-value. By utilizing this method, we can save on the expenses incurred in computing the exact p-value, which can be quite costly, for rules that are not significant. Upon calculating the p-value, the algorithm proceeds to assess the candidate rule's minimality and non-redundancy to determine whether it is a viable selection. It is imperative that the p-value of every parent of the LHS of the candid rule must not be less than the p-value of the candid rule itself. Ultimately, to ascertain whether a candidate rule is minimal, we verify that all occurrences of the LHS itemsets in the training set pertain to the identical class.

An inherent advantage of employing SigDirect as the transparent model in comparison to other classifiers such as CPAR or CMAR is the elimination of the need to calibrate any hyper-parameters in SigDirect. By hyper-parameter, we focus on support and confidence thresholds that play a major role in CBA and CMAR. This benefit proves particularly useful when elucidating a black-box model, where it is crucial to train the explainer, i.e., the transparent model like SigDirect, for each unique decision of the black-box, i.e., varying test instances. Furthermore, typically, there is a lack of ground truth data to measure the accuracy of the explanations generated by the explainer model. Therefore, there exists no simple method for end-users to adjust any hyper-parameters. Therefore, there exists no simple method for end-users to adjust any hyper-parameters. The authors [2] also demonstrate the performance of SigDirect not only in terms of accuracy but also its effectiveness in generating fewer rules. This benefit enables the end-users to get a better understanding of the explanation effortlessly.

SigDirect is available as a Python package[1]. It exploits certain numerical libraries like Numpy and Scipy to perform calculations more rapidly than with pure Python code. We opt to use Python, rather than other programming languages like C++, which are generally faster, because it enables other researchers to enhance or customize SigDirect to suit their specific requirements. The increasing popularity of Python in the field of machine learning cannot be understated. Python has now surpassed other programming languages in ML, both in academics and industry. According to recent research, 57% of machine learning developers and data scientists prefer to use Python for ML tasks [100].

[1]You can access the code at https://github.com/mhmotallebi/sigdirect.

SigDirect is an associative classifier. We are using SigDirect as the core component of our proposed methodology. An issue with SigDirect is that it generates many rules making it difficult to evaluate them. Zaiane and Antonie [101] conduct a study to reduce the number of rules by introducing a two-stage pruning technique without affecting the accuracy of the classifier. Their proposed method called SigD2 improves the interpretability of the learned model by reducing the number of rules using the pruning technique. However, SigDirect and SigD2 both have a limitation in dealing with datasets with a large feature vector, as it requires high memory and long run time. Therefore, in the case of datasets with high dimensions, SigDirect and SigD2 encounter memory and run-time problems due to the large feature vector space. To mitigate these problems in SigDirect and SigD2 when dealing with high-dimensional datasets, an ensemble of associative classifiers can be employed. In the following section, we explore CFAR, a methodology that leverages an ensemble approach to tackle the aforementioned challenges.

3.4 CFAR

Kabir and Zaiane [3] introduce Classification by Frequent Association Rules (CFAR), that can handle high-dimensional datasets. CFAR addresses the limitations of SigDirect and SigD2 by following an ensemble approach where multiple base learners are used to generate association rules. Each base learner is trained with a subset of the feature vector, and the rules generated by all the base learners are aggregated and ranked based on their frequency. The rules having higher frequency are eventually selected to explain the decision of the black-box model.

Ensembling associative classifiers allows collecting all the rules learned from training data for labeling test data to a class label. But the inclusion of noisy rules can affect the model's performance. To overcome this, CFAR uses Relative Frequency Ratio (RFR) to select rules that enhance the model's performance. Once the rules are generated by the base learners, the frequency of each rule is calculated and the maximum frequency is determined as $max\_frequency$. For a rule R, the RFR can be calculated as $RFR(R) = \frac{\text{Frequency (R)}}{\text{max\_frequency}}$.

CFAR assumes a threshold T to select the rules having $RFR(R) \geq T$. Finding the optimal value of T is crucial to maximize the accuracy. Hence, CFAR implements a 3 step strategy:

Rule generation : During the rule generation phase, CFAR adopts a random subsampling approach to train 100 SigD2 base learners. The dataset is first divided into training and validation data with an 80:20 ratio. Then, the training data is further split into training and test data in an 80:20 ratio. To train each base learner, CFAR uses a subset of the feature vector of size 30. After training, all the generated rules by the base learners are gathered, and RFR of each rule is calculated.

Find optimum value for T : The optimum value for T is found by trying different values in a linear search fashion. It is assumed that the frequency of a rule among the base learners defines its importance. The generated rules are obtained and the class label is predicted with the selected rules. The value of T is initially set to 1 and is decreased by 0.1 in each step to perform rule selection and class prediction. The accuracy is calculated in each step and the value of T is decreased until the best improvement in accuracy is achieved. If there is no improvement for a certain value of T, the search is stopped and the T value from the previous step is fixed as the optimum T.

Prediction : In the final step, the value of T that provides the best result is obtained from the previous step. Using this value of T, rules are selected and the class label of the validation data is predicted. The performance of the model is then calculated based on the predicted class label. Figure 3.1 shows the architecture of CFAR.

We are introducing CFAR in this section since we want to use it as the core of our proposed methodology called BARBE. Chapter 4 presents BARBE in detail. We are interested to compare the performance of BARBE with SigDirect against BARBE with CFAR to demonstrate the capability of BARBE with CFAR to overcome the limitations that BARBE with SigDirect has while handling high-dimensional datasets. CFAR is nothing but the ensemble of SigD2 where rules generated by a set of base learners are eventually merged together to generate the final explanation.

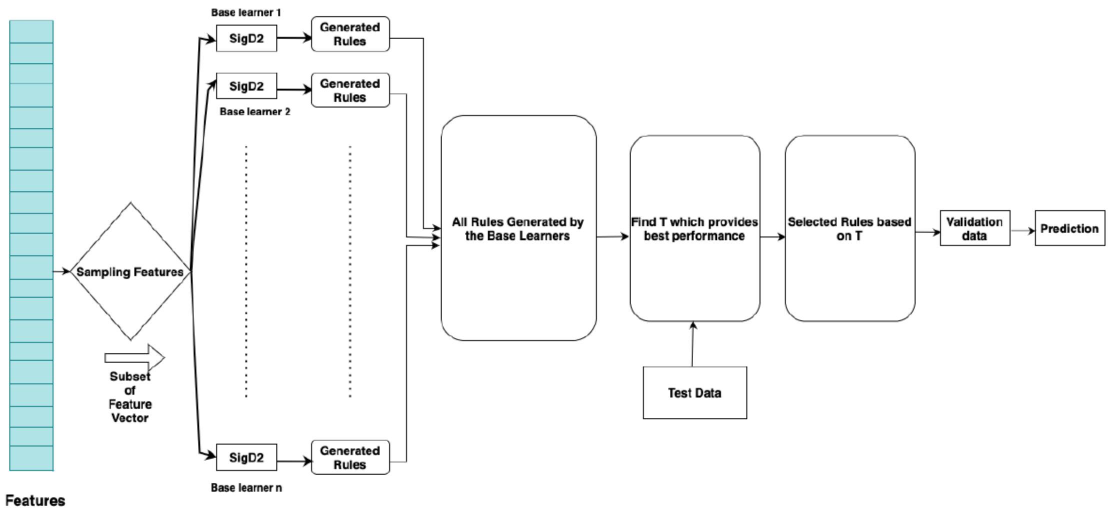


Figure 3.1: The architectural diagram of CFAR taken from [3]. Rules are generated by base learners using training data. The generated rules are then applied to test data to determine the T value that yields the best performance. Final rules are selected using the T value and applied to validation data to evaluate the model's performance.

Chapter 4

BARBE: Black-box Association Rule-Based Explanations

We introduce Black-box Association Rule-Based Explanations (BARBE) in this chapter, our proposed method, which takes advantage of associative classification rules to generate explanations that are not only better but also more easily understood by humans than those produced by other methods, such as LIME, Anchor, and SHAP. Section 4.1 of this chapter highlights challenges that are present in LIME, as well as other model-agnostic explainers. We introduce BARBE and elucidate how it overcomes the aforementioned shortcoming in Section 4.2. We present a demonstration of how BARBE's explanation looks like and provide an overview of its architecture in this Section.

4.1 Shortcomings of Other Methods

Take what LIME generates for the text "I did enjoy as it was an excellent movie" as shown in Figure 2.2. What do the numbers in front of each feature mean? These "importance scores" are simply used for ranking. For example, 0.12 for *excellent* and 0.05 for *enjoy* highlight that *excellent* has higher importance than *enjoy* in making the sentence labeled by the black-box as positive. This leads us to the conclusion that only the order among features matters to the users and not the numbers generated in the explanations. Since LIME uses a weighted loss function for its linear model

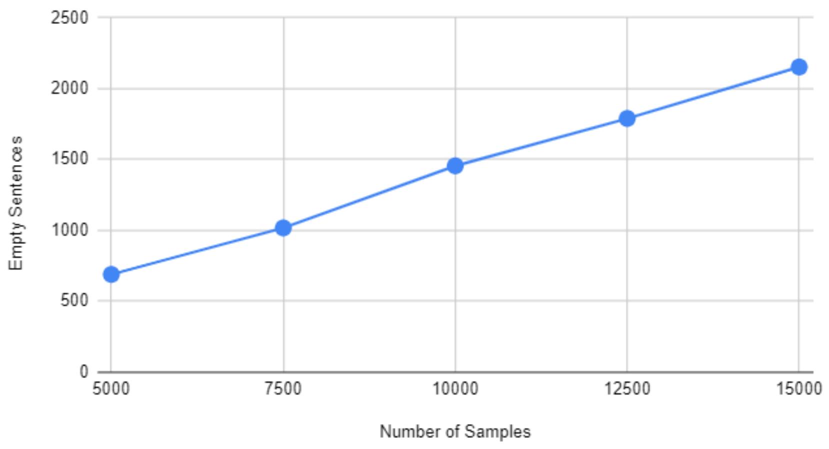


Figure 4.1: Number of empty sentences generated by LIME across different sample sizes

that also benefits from regularisation, it is likely that the instances which are not in the very close proximity of the original instance would be misclassified by this linear model, thus providing wrong explanations to the user.

During the process of generating a neighborhood around the input text for LIME, random words are removed from the input sentence to create a synthetic dataset. As observed during our analysis, this method sometimes leads to the creation of empty sentences with no words in them. Figure 4.1 illustrates the number of empty sentences generated by LIME while creating the synthetic dataset for different sample sizes. By default, LIME produces a synthetic dataset of 5,000 [1] sentences around the provided text. We examine the occurrence of empty sentences in the synthetic dataset produced by LIME across various sample sizes: 5,000, 7,500, 10,000, 12,500, and 15,000. Our investigation finds that the synthetic dataset generated by LIME consistently contains around 13%-15% empty sentences. There is a potential for improving the neighborhood generation process by ensuring that any sentence that contains at least one word is included instead of having empty sentences.

Ribeiro et al. [5], the same authors of LIME, also point out another shortcoming of methods like LIME, the fact that features are taken independently (see example

[1] https://github.com/marcotcr/lime/blob/fd7eb2e6f760619c29fca0187c07b82157601b32/lime/lime_text.py#L374

in Figure 2.2). They introduce Anchor to overcome this issue. In their new method, an explanation is a set of features that whenever they co-occur, the class label is determined with a 95% confidence. This Anchor essentially resembles a rule (with a high confidence threshold of 95%).

The authors of LORE [42] benefit from the idea of using a rule as the explanation in their method as well. In their method, however, Guidotti et al. [42] also provide a set of counter-factual rules helping the user find ways to have a new data point that is labeled differently than the original one while having the least different features compared to it.

Despite the fact that these methods, to some degree, overcome the problem mentioned above, one issue remains: is there always only one set of correlative features (and hence one reason) behind the final outcome of the model? What if there are multiple sets of correlative features that independently derive the final conclusion of the system [102]. Therefore an explanation should not solely focus on independent features or one unique set of associated features but on possibly a set of causes. Hence the interest in an associative classifier that can provide a set of rules as an explanation.

4.2 Proposed Resolution for the Challenge: BARBE

To overcome the above shortcomings, we introduce Black-box Association Rule-Based Explanations or BARBE.

Unlike LIME, BARBE provides a set of rules as the explanation, where not only do rules provide users with important features (what LIME does), but also take care of the associations among them (what LORE and Anchor do). In addition, since we provide multiple rules as an explanation, we can hint at multiple causes that have led to that decision by the system, something that the aforementioned methods are unable to provide. Note that using a decision tree (in systems like LORE) the path in the tree leading to the predicted label results in a single applicable rule which constitutes only one unique cause.

4.2.1 Explanations by BARBE

BARBE generates a descriptive model learned on data labeled by the black-box and provides as the explanation a subset of rules from the model that apply to the instance for which the explanation is expected. From this set of rules and their individual measure of confidence and significance, BARBE can provide an ordered set of important features as an alternative way of providing explanations. This allows the users to have the choice to look at these two types and get a better understanding of the underlying causes. Moreover, as mentioned earlier, each rule in addition to the items in its antecedent and the class label, comes with added information such as its confidence, support value, and p-value.

Figure 4.2 and Table 4.1 demonstrate an example of what BARBE produces for a text labeled as negative by the black-box model. In this example, BARBE produces five rules according to Table 4.1 by which they not only provide important features to the users but also provide the associations among the features. The rules are sorted in this table based on their confidence values. Note that the first four out of five rules provided in this explanation infer class labels negative with higher confidence whereas only one of them infers class labels positive with very low confidence. These five rules are "applicable rules" since these rules include the same features as the original instance. The first four out of five rules are "applied rules" which are a subset of "applicable rules" whose labels match the instance label as predicted by the black-box model.

4.2.2 How does BARBE work?

A high-level representation of the activity diagram of BARBE is shown in Figure 4.3. BARBE generates neighborhood instances around the instance to be explained by creating synthetic instances around the original instance. The synthetic instances

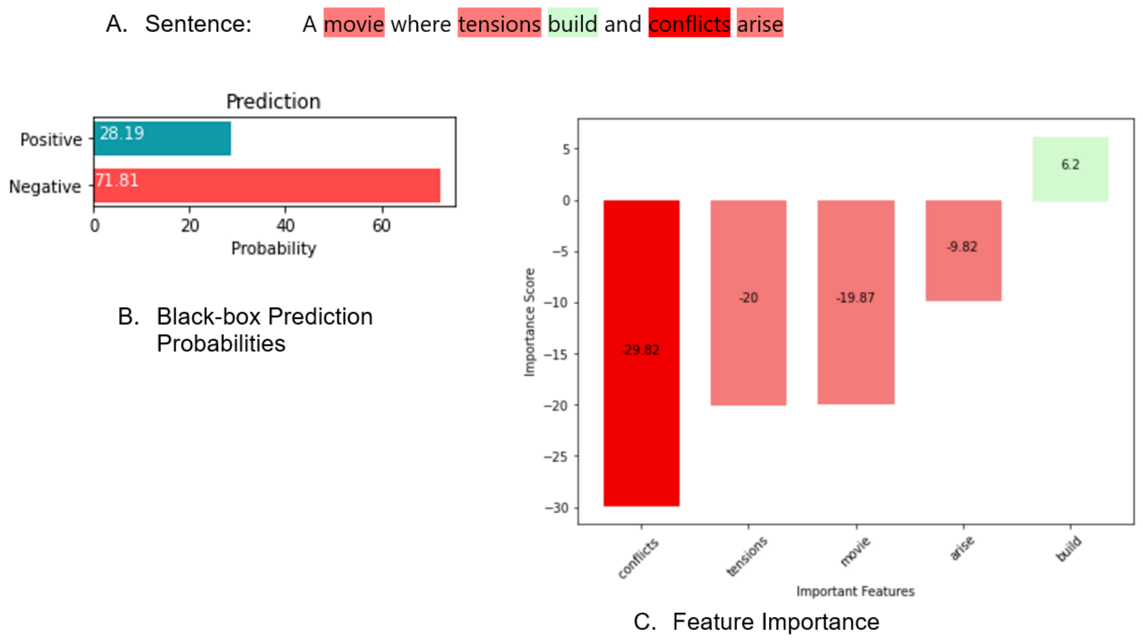


Figure 4.2: The explanation provided by BARBE for an instance of the IMDB movie review dataset labeled as negative by the black-box model. (A) shows the sentence with features highlighted. The red heatmap presents the negative words and the green heatmap presents the positive ones. Table 4.1 presents the set of important rules with their support, confidence, and logarithm of statistical significance values. (B) presents the prediction probability of the black-box, and (C) presents the histogram of important features ranked based on their importance.

Table 4.1: Set of rules generated by BARBE with their support, confidence, and logarithm of statistical significance values along with the sentence with features highlighted. Rules are sorted in this table based on their confidence values. “N” denotes “Negative” label and “P” denotes “Positive” label in the Rule column.

Sentence	**Rule**	**Support**	**Confidence**	**Log(SS)**
A movie where tensions build and conflicts arise	$tensions \rightarrow N$	48.91	100.00%	-53.82
A movie where tensions build and conflicts arise	$conflicts \rightarrow N$	43.56	100.00%	-56.92
A movie where tensions build and conflicts arise	$movie \rightarrow N$	50.81	99.35%	-52.87
A movie where tensions build and conflicts arise	$conflicts\ arise \rightarrow N$	25.36	98.19%	-56.45
A movie where tensions build and conflicts arise	$build \rightarrow P$	2.00	31.01%	-55.92

are labeled by the black-box which produces a training set for the SigDirect classifier. The outcome of the training is a set of rules. Rules from the trained model relevant to the original instance are extracted. Lastly, BARBE derives important features from these rules and presents them along with the rules as the explanation. We briefly describe the core components of BARBE in this section.

Neighbourhood Generation For text data, BARBE uses a simple strategy to generate a synthetic dataset around the original instance. The algorithm to generate synthetic text data for BARBE has been presented in Algorithm 1. In order to generate synthetic text data, BARBE makes use of random word removal from the input sentence. This process involves specifying a number n that determines how many times the algorithm should iterate to produce n new sentences. The algorithm then chooses a set of random positions within the input sentence and removes the words at those positions. The resulting sentence, with random words removed, is returned as a synthetic sentence. This process is repeated n times, resulting in n

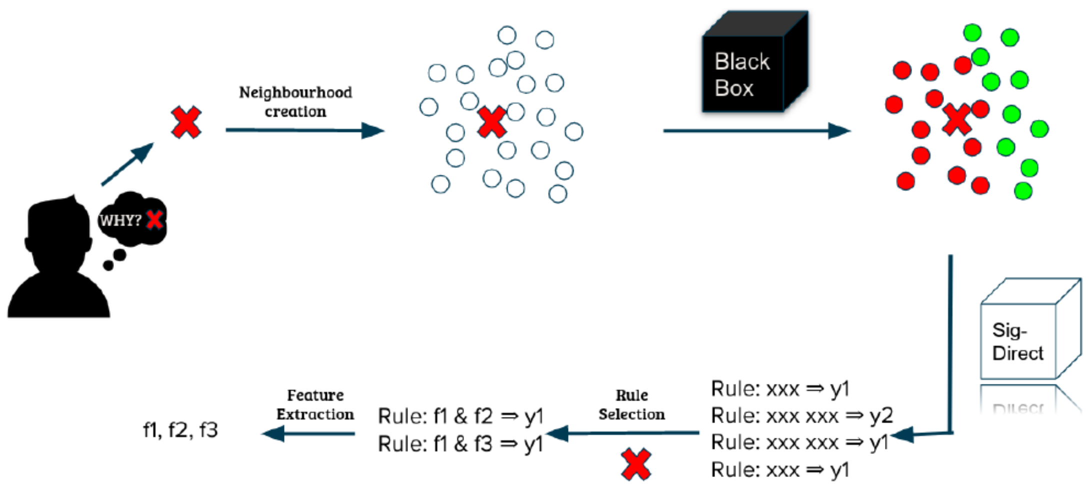


Figure 4.3: A high-level overview of how BARBE generates explanations. BARBE first creates a neighborhood around the provided instance, which comprises perturbed instances created from the provided instance. Next, the black-box is queried to label this perturbed dataset. The perturbed dataset and their corresponding labels are then used to train a supervised model using SigDirect classifier. This results in a set of rules. Relevant rules are extracted from the trained model using the original data point. Finally, BARBE reports the relevant rules along with the important features extracted from them as the explanation to the user.

new sentences. These sentences form the neighborhood dataset around the original text, providing additional instances for the model to learn from. By creating multiple synthetic sentences, BARBE generates the set of perturbed instances which are sent to the black-box for labeling. Unlike LIME, Algorithm 1 ensures no empty sentence is generated for BARBE.

Algorithm 1 BARBE Neighborhood Generation Algorithm

```
Input: $inputSentence, n$
**for** $i \leftarrow 1$ to $n$ **do**
    $sentence \leftarrow inputSentence$
    $nbWordsToDelete \leftarrow random[0, numberOfWords(inputSentence) - 1]$
    **for** $j \leftarrow 1$ to $nbWordsToDelete$ **do**
        $t$ = Select a word randomly and remove it from $inputSentence$
        $outputSentence = outputSentence \cup t$
    **end for**
**end for**
**return** $outputSentence$
```

It is important to note the issues with random word removal in the context of data augmentation. Random word removal may not account for the syntactic or semantic relationships between the words in the original sentence, which may impact the analysis performed on the augmented data. Consider a sentence "I thought this was a wonderful way to spend time on a too hot summer weekend, sitting in the air conditioned theater and watching a comedy" for which we want to create a synthetic dataset from it using random word removal technique. After random word removal, one sentence for example may become: "I thought this was a wonderful way to spend time on a hot summer weekend, sitting in the air theater and watching a comedy". The random word removal technique has removed the words "too" and "conditioned". Such removals may result in the loss of important semantic context. We will delve into this topic further in Section 5.4.

Interpretable model (SigDirect) To create a new training dataset for BARBE, the methodology described above combines the original instance with the synthetic dataset generated through the perturbation process. This combined dataset along with their labels obtained from the black-box is then transformed using one-hot encoding, which converts categorical data into transactions, a convenient data format for SigDirect. Once the data has been transformed, BARBE trains its associative classifier using the new dataset, which produces a set of rules based on the relationships between different features. It is important to note the difference in what LIME does here. LIME creates a synthetic dataset by perturbing the original data, then uses a linear regression model to fit the synthetic data and generate explanations. Linear regression is a type of linear model that is commonly used to identify relationships between input features and output labels. Linear regression is useful for identifying linear relationships between input features and output labels but is not effective for identifying complex relationships and ignores relationships between input features. In addition, it is worth noting that LIME uses and depends on the probability scores generated by the black-box model, whereas BARBE only requires the labels or output values produced by the black-box model.

Rule Extraction In this step, BARBE extracts pertinent rules from the set of rules generated by SigDirect during training, using an approach that involves identifying "applied" and "applicable" rules. "Applicable" rules are those that contain the same feature values as the original instance, while "applied" rules are a subset of the applicable rules that have labels that agree with the label assigned to the instance by the black-box model. Although the association rules generated by the associative classifier are generally relevant to the instance being analyzed, BARBE further refines these rules to identify the most important ones. This process involves narrowing down the rules based on their relevance to the specific instance being explained, which is possible because the training data used to generate the rules is made up of instances that are similar to the original data point.

Feature Extraction Finally, BARBE leverages important rules in order to identify important features. In this step, it selects the most relevant features from the important rules to explain the behavior of the model. To do this, BARBE utilizes a one-hot decoding process to transform the features and rules selected into the feature space of the black-box model. This allows the selected features to be used to explain how the model is making predictions. As an option, the user may specify a value for k when using BARBE, and the system will then return up to k important features. Otherwise, BARBE can provide all of the important features it has identified within important rules. To obtain these features, BARBE examines the rules extracted in the Rule Extraction step and ranks them based on information provided by the SigDirect, including confidence, support, and p-value scores.

It is important to note the basic difference between BARBE with SigDirect and BARBE with CFAR in terms of architecture. The only difference in Figure 4.3 for BARBE with CFAR is the "SigDirect" block to be replaced by CFAR. CFAR is nothing but an ensemble of SigD2 classifiers. BARBE with SigDirect uses a single SigDirect classifier as the underlying associative classifier to generate association rules whereas BARBE with CFAR uses an ensemble of SigD2 classifiers to achieve that purpose.

Chapter 5

Experiments on BARBE

This Chapter outlines the settings used for conducting experiments to evaluate the performance of BARBE for text dataset. We demonstrate both the performance of BARBE with SigDirect and BARBE with CFAR in this Chapter. To analyze the effectiveness of BARBE with SigDirect and BARBE with CFAR in handling text datasets, we train black-box models on both binary and multiclass datasets. By doing so, we aim to showcase how well BARBE can perform across a range of classification tasks, thereby highlighting its potential as a general-purpose tool for generating explanations. Since SigDirect has limitations in processing datasets with a large feature vector space (as discussed in Section 3.3), we choose short sentences for BARBE with SigDirect. On the other hand, for BARBE with CFAR, which does not have such limitations, we select long sentences to generate explanations. We begin by introducing the datasets we used in Section 5.1, followed by a description of our approach to evaluate the performance of BARBE with SigDirect and BARBE with CFAR in Section 5.2. We compare the explanation generated by BARBE with SigDirect and BARBE with CFAR against LIME, Anchor, and SHAP in Section 5.3. In the end, we discuss a parse tree-based approach in Section 5.4 to modify the neighborhood generation technique incorporated by BARBE with CFAR to generate more meaningful rules. We use the shorthand notations BARBE-S and BARBE-C to refer to BARBE with SigDirect and BARBE with CFAR, respectively in this Chapter.

5.1 Datasets

Since we aim to develop a framework that can be applied to any text dataset, it is necessary to adapt BARBE to different datasets to ensure its robust performance on unseen datasets. In this regard, we conduct our experimental analysis on binary and multiclass classification tasks. For binary classification, we select IMDB movie review dataset [38]. The IMDB movie review dataset is a well-known dataset in the field of NLP and machine learning. It consists of 50,000 movie reviews that were scraped from the Internet Movie Database (IMDB), a popular website that provides information about movies and television shows. Each review in the dataset is labeled with a binary sentiment classification (positive or negative), indicating whether the reviewer had a positive or negative opinion about the movie. The dataset is evenly split between positive and negative reviews, with 25,000 reviews in each category. The dataset was created by researchers at Stanford University and has since become widely used in the research community as a benchmark dataset for text classification and sentiment analysis tasks. Because of its large size and high quality, it has been used in a variety of research applications, including the development and evaluation of machine learning models and techniques for NLP. It is available for free download from various online sources and can be easily incorporated into research projects.

For multiclass classification, we select the AG's News Topic Classification Dataset [103] which is widely used for multiclass text classification. It consists of news articles collected from more than 2000 news sources and has been preprocessed to remove any noise and redundant information. The dataset is composed of four different topics: World, Sports, Business, and Science/Technology, and is often used as a benchmark dataset for evaluating the performance of various multiclass text classification models. The dataset has a balanced distribution of classes, making it suitable for multiclass classification tasks.

5.2 Experiments Setup

Evaluation is a big challenge in the XAI literature since figuring out the underlying causes of a decision made by a black-box model is challenging. Moreover, the significance of those causes to a human observer is not a good measure to evaluate the accuracy of the explainer framework. For example, during the training of a deep learning model, let us assume we have mistakenly exposed the ground truth labels of development instances (directly or indirectly) to the model. As a result, our model is performing very well on development and thus we stop the training early. This is while in reality it has not learned the hidden patterns present in the provided data. At this point, we want to evaluate the performance of a well-built model-independent explanation technique by taking advantage of this deep model (which is a black-box model) on one of the development instances. In this scenario, the explanation will include a feature or features that were pointing to the label (the wrongly exposed features). If the produced explanation is evaluated by a human observer, it will be marked as a wrong one because it is not what the human observer was expecting (it is different than the real pattern hidden in the data). In reality, however, the model is indeed explaining the black-box model rightfully, the task it was supposed to do. One way to address this issue is to compare the explanation generated by BARBE-S and BARBE-C with the explanation generated by other state-of-the-art explainers such as LIME, Anchor, and SHAP. By doing so, we can determine the effectiveness of BARBE in generating accurate and meaningful explanations, even in cases where the underlying causes of a decision made by a black-box model are difficult to understand. Therefore, we compare the explanation generated by BARBE-S and BARBE-C against the explanation generated by LIME, Anchor, and SHAP.

The black-box model we leverage in our experiments for binary classification tasks is the SVM. SVM is a popular machine learning algorithm that is often used for text classification tasks. There are several reasons why SVMs are efficient for text classification:

- High-dimensional feature spaces: The purpose of text classification is to categorize documents or fragments of text depending on their content. Text data is commonly represented as a bag of words, which entails counting the occurrences of each word in the document and constructing a vector of word frequencies. Yet, because documents can contain a significant number of unique words, the resulting vector can be extremely high-dimensional, with hundreds or thousands, or even millions of features. SVMs are particularly effective at handling high-dimensional feature spaces because they use a kernel trick to implicitly represent the data in a higher-dimensional space. This allows SVMs to capture more complex relationships between the features and the labels, even when the feature space is very large.

- Robust to noise: Text data is often noisy, which means that it contains errors or inconsistencies that can affect the accuracy of a classification model. For example, a document may contain misspelled words, grammatical errors, or other inconsistencies that make it more difficult to classify. SVM is robust to noise in the data because they are designed to identify patterns in the data that are robust to noise (e.g. outliers). This means that even if a document contains some errors or inconsistencies, the SVM model can still identify the underlying patterns in the data that are relevant for classification. One way that SVM is able to handle noise in the data is by using a margin-based approach. In a binary classification task, SVM tries to identify a hyperplane that separates the positive and negative examples in the feature space, and this hyperplane is chosen to maximize the margin between the two classes. By maximizing the margin, the SVM model is better able to tolerate noise and outliers in the data. Another way that SVM is able to handle noise in the data is by using regularization. Regularization helps to prevent overfitting to the training data, which can make the model more sensitive to noise and outliers.

- Binary classification: Although popular for different multiclass and binary classification tasks, SVM is particularly well-suited for binary classification tasks, which are common in text classification. For example, SVMs can be used to classify documents into "positive" or "negative" sentiment categories, "spam" or "not spam" categories.

The performance of SVM can be influenced by several hyperparameters, such as the kernel type, regularization parameter, and the degree of the polynomial kernel. In brief, we discuss the hyperparameters for SVM in our sentiment classification task.

Kernel type : The performance of the SVM can be considerably impacted by the selected kernel type. Common kernel types for binary sentiment classification include linear, polynomial, and Radial Basis Functions (RBF). While polynomial and RBF kernels can handle more complex decision boundaries, linear kernels are often useful for datasets that can be separated along linear axes. The ideal kernel type for a particular dataset can be found by doing a grid search or randomized search across the kernel parameter space. We select RBF as the kernel type during our experiment because of its capability in capturing complex non-linear relationships between the input features and the target variable.

Regularization parameter (C) : The regularization parameter (C) controls the trade-off between maximizing the margin and minimizing the training error. A high value of C leads to a narrow margin and potentially overfitting, while a low value of C leads to a wider margin and potentially underfitting. A grid search or randomized search can be performed over the range of C values to identify the optimal value for the given dataset. We have set C=1.0 in our experiment which is the default setting of the Python package [1].

[1]https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html

For multiclass text classification tasks, we leverage Convolutional Neural Network (CNN) as the black-box model for our experiments. CNNs are good for text classification for several reasons:

- The Convolutional layers in CNN can learn local patterns and relationships between words in a sentence. This is particularly useful for text classification tasks because the meaning of a sentence is often determined by the combination of its constituent words.
- To extract the most essential elements from the learned patterns, pooling layers might be used. This helps in reducing the dimensionality of the data and capturing the most salient information.
- In multiclass classification, CNNs can learn to differentiate between multiple classes by identifying the unique features that distinguish them. This is particularly useful when the classes are not well-separated or when there is overlap between them.

The performance of CNN can be enhanced by appropriately tuning the hyperparameters. In a CNN, hyperparameters determine the architecture of the network and how the model will be trained. Choosing the right hyperparameters can significantly impact the performance of the model, including accuracy, training time, and resource usage. Some of the important hyperparameters in a CNN include the number and size of convolutional layers, pooling layers, activation functions, learning rate, batch size, and dropout rate. The number and size of layers determine the complexity of the model and its ability to extract meaningful features from the input data. The learning rate determines the step size of the optimizer during training. The batch size determines the number of samples used in each training iteration. The dropout rate determines the amount of regularization applied during training. Here are the values of the hyperparameters we have used to train a CNN with the AG's News Topic Classification Dataset:

- Number of filters: 128
- Kernel size: 3
- Pooling size: 2
- Dropout rate: 0.3
- Learning rate: 0.001
- Optimizer: Adam
- Batch size: 64
- Number of epochs: 30

We have used 3 convolutional layers for the CNN in our experiments. We are not going to discuss why we have picked these values for the hyperparameters setting since our main focus is on analyzing the explanation generated by BARBE-S and BARBE-C. We implement the CNN using the Python package of Keras [2] for our multiclass classification task.

Prior to training our model, it is necessary to transform the text data into a numeric form. The most widely used techniques for achieving this are Bag-of-Words (BOW) [104] and Term Frequency-Inverse Document Frequency (TFIDF) [105], which can both generate numeric representations of input text. The bag-of-words model transforms text into fixed-length vectors by counting the frequency of each word. On the other hand, TFIDF increases the weight of words that appear frequently within a document, while reducing the weight of words that appear frequently across multiple documents. Therefore, common words such as 'this' or 'are' are assigned a lower weight, while words that occur frequently in a few documents are assigned a higher weight. We have used TFIDF to convert the text into features in BARBE-S and BARBE-C. We train SVM as the black-box model with the IMDB movie review

[2]https://keras.io/

dataset and CNN with the AG's news topic classification dataset. We split the dataset into 80% training, 10% validation, and 10% testing sets using random sampling to ensure that the distribution of labels is consistent across all three sets. We achieve 90.10% accuracy using SVM for the binary classification task and 93.25% accuracy using CNN for the multiclass classification task. Once the black-box models are trained, we utilize the neighborhood generation process discussed in Section 4.2.2 to create a synthetic dataset. This dataset is then labeled by the black-box models and used by BARBE-S and BARBE-C to generate explanations in the form of rules.

5.3 Comparison with Other Explainers

We evaluate our method for explaining text against three other competitive explainers as discussed in Section 2.2.2. The explainers are LIME, Anchor, and SHAP. LIME has been widely used as a model-independent method in various domains and is currently the most cited. Additionally, there have been several extensions to LIME introduced over time. Another method that utilizes rules to provide explanations for the decisions of a black-box model is Anchor. Additionally, SHAP aims to provide unified and consistent explanations for a wide range of models and domains. To facilitate our analysis, we divide our experiments into two parts: one for binary classification task using SVM as the black-box model and another for multiclass classification task using CNN as the black-box model. For conducting experiments with binary classification, we select two short sentences from the IMDB movie review dataset for BARBE-S, one labeled as positive and another labeled as negative by the SVM black-box. We generate the explanation using BARBE-S for these two sentences and compare it with the explanation generated by LIME, Anchor, and SHAP. Following that, we select two long sentences from the IMDB movie review dataset for BARBE-C (since BARBE-S has limitations in processing long sentences), one labeled as positive and another labeled as negative by the SVM black-box. We again compare the explanation produced by BARBE-C with the explanation produced by LIME, Anchor, and SHAP.

For conducting experiments with multiclass classification using CNN as the black-box, we select a short sentence from the AG's news topic classification dataset and provide it to the black-box for predicting the news class. Then we ask BARBE-S to generate an explanation for it. We compare this explanation with LIME, Anchor, and SHAP. Furthermore, we select a long sentence for the same dataset to generate an explanation by BARBE-C. We again compare this explanation with LIME, Anchor, and SHAP.

5.3.1 Experiments on Binary Classification Task

BARBE typically uses the data labeled by the black-box model to train a descriptive model that generates a set of rules as the explanation. Each rule has a support, confidence, and statistical significance value associated with it. The rules correspond to the features in the data which constitute the set of important features as the form of explanation. Figure 5.1 demonstrates the result generated by BARBE-S for a sentence with positive sentiment. Figure 5.1A shows the sentence for which BARBE-S generates an explanation. The sentence "Get this wonderful story about success in life" has a positive sentiment as labeled by the black-box model. BARBE-S employs a green gradient to indicate the importance of positive words within a sentence, with stronger shades of green representing the most important words for making the sentence positive as labeled by the black-box model, and lighter shades indicating less important words. The colors are provided to the words based on their frequency within the rules. Since "wonderful" appears in 2 out of 5 rules, it has got a stronger green whereas the word "story" receives a lighter green since it appears in 1 out of 5 rules. A sentence may also contain words that convey a negative impression, and to highlight their importance, BARBE-S utilizes a red gradient. Stronger shades of red represent the words with more negative, while lighter shades of red indicate words that are less negative. By incorporating both green and red gradients/highlights, BARBE-S provides a comprehensive and intuitive visualization of the sentiment expressed in

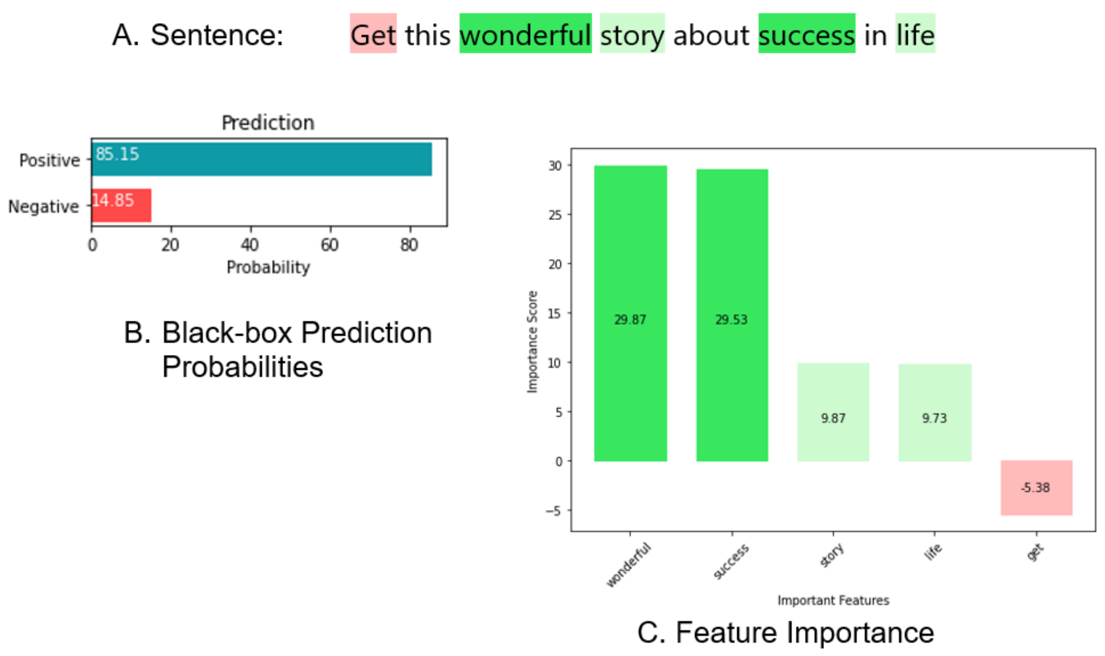


Figure 5.1: The explanation provided by BARBE-S for an instance of the IMDB movie review dataset labeled as positive by the black-box model. (A) shows the sentence with features highlighted. The green heatmap presents the positive words and the red heatmap presents the negative ones. Table 5.1 presents the set of important rules with their support, confidence, and logarithm of statistical significance values. (B) presents the prediction probability of the black-box, and (C) presents the histogram of important features ranked based on their importance.

a sentence, aiding users to understand the underlying reasoning behind the model's prediction. As demonstrated in Figure 5.1A, the words "wonderful" and "success" are deemed highly important in conveying a positive sentiment, as determined by the black-box model. We will explain shortly why these words have got higher importance. The words "story" and "life" are relatively less impactful in conveying a positive sentiment within the sentence. On the other hand, the word "get" has a negative impact on the black-box decision.

The rules generated by BARBE-S are highlighted in Table 5.1. BARBE-S generates a total of 5 rules for this sentence to identify the most important features. These rules are sorted in Table 5.1 based on their confidence values. It is noteworthy that BARBE-S generates a set of rules, not a single rule, and each rule contains a set of words. For the sentence depicted in Figure 5.1A, BARBE-S generates the association

Table 5.1: Set of rules generated by BARBE-S with their support, confidence, and logarithm of statistical significance values along with the sentence with features highlighted. Rules are sorted in this table based on their confidence values. "P" denotes "Positive" label and "N" denotes "Negative" label in the Rule column.

Sentence	**Rule**	**Support**	**Confidence**	**Log(SS)**
Get this wonderful story about success in life	$wonderful \rightarrow P$	46.18	100.00%	-382.93
Get this wonderful story about success in life	$success \rightarrow P$	45.84	99.00%	-269.74
Get this wonderful story about success in life	$wonderful\ \ story \rightarrow P$	25.50	98.72%	-382.02
Get this wonderful story about success in life	$success\ \ life \rightarrow P$	26.86	97.29%	-269.09
Get this wonderful story about success in life	$Get \rightarrow N$	13.75	26.92%	-7.80

rules "wonderful" and "wonderful story" to identify the semantic context required to classify the sentence as positive. These rules capture the key features of the sentence that contribute to its positive sentiment, highlighting the importance of specific words and word combinations in determining the sentiment of a given text. Each rule has its support, confidence, and statistical significance values. BARBE-S provides the logarithm of the p-value reported by SigDirect. This logarithmic transformation can help to highlight smaller p-values, which may be easily overlooked when presented in their original scale. By presenting the logarithm of the p-values, users can more readily identify statistically significant rules. For example, the rule $wonderful \rightarrow Positive$ depicts that the presence of the word "wonderful" is good to treat this sentence as positive. The rule $wonderful\ \ story \rightarrow Positive$ depicts that the presence of the conjunction of the words "wonderful" and "story" can also make the sentence positive. The rule $wonderful \rightarrow Positive$ has support of 46.18, confidence of 100%, and statistical significance -382.93. It is evident that these two rules have a higher confidence value of 100% and 98.72% respectively. Similarly, the two other rules that are responsible for making the sentence positive are $success \rightarrow Positive$ and $success \quad life \rightarrow Positive$. Their confidence values are 99% and 97.29% respectively. Thus, BARBE-S is able to generate a set of explanations to clarify the reasoning behind the decision of the black-box model. Besides, the conjunction of words within the rules asserts that the presence of a conjunction of words can also give justification for the decision made by the black-box model. A conjunction of multiple words: "wonderful story", and "success life" have a better semantic context within the sentence. Such rules emphasize that the two words may stay together in the explanation generated by BARBE-S. It is worth noting that BARBE-S has the capability to identify the rules i.e. features, that contribute to the opposite labeling of a sentence by the black-box model. The rule $get \rightarrow Negative$ although tells us that the feature "Get" has a reverse impact on the black-box decision, this rule has a poor confidence value.

BARBE-S presents the prediction probabilities of the black-box model as presented in Figure 5.1B. The vertical bar chart in Figure 5.1C showcases the most important features based on their frequency within the rules, along with the corresponding weighted confidence value. It is evident from this figure that the words "wonderful" and "success" are the most important features whereas "story", and "life" are the least important ones. The color in this vertical bar chart comes from the color of the words according to Figure 5.1A.

We use LIME as the underlying explainer with the same black-box model and hyperparameter settings to generate an explanation for the sentence "Get this wonderful story about success in life". Figure 5.2 shows the explanation generated by LIME. LIME assigns numerical values only to the features of the input data that are deemed to be important in the prediction process. These values represent the degree of importance assigned to each feature by the LIME model. Larger values indicate that the corresponding feature has a greater impact on the model's prediction, while smaller values indicate that the feature has a lesser impact. From the numbers shown in the top-right bar chat in Figure 5.2, a user can only understand that the words "life" and "wonderful" are the most significant words contributing to the positive label assigned by the black-box model. "success" and "story" have got lower importance score. LIME assigns a higher important score to "life" and a lower important score to "success". But logically, "success" and "wonderful" should be in the front line to carry positive sentiment as expected. LIME is not able to generate any rule here. A rule is more human-readable and understandable. It appears that LIME has only been able to detect single features rather than combinations of multiple features. There has been no demonstration of LIME's ability to handle conjunctions of features in this scenario.

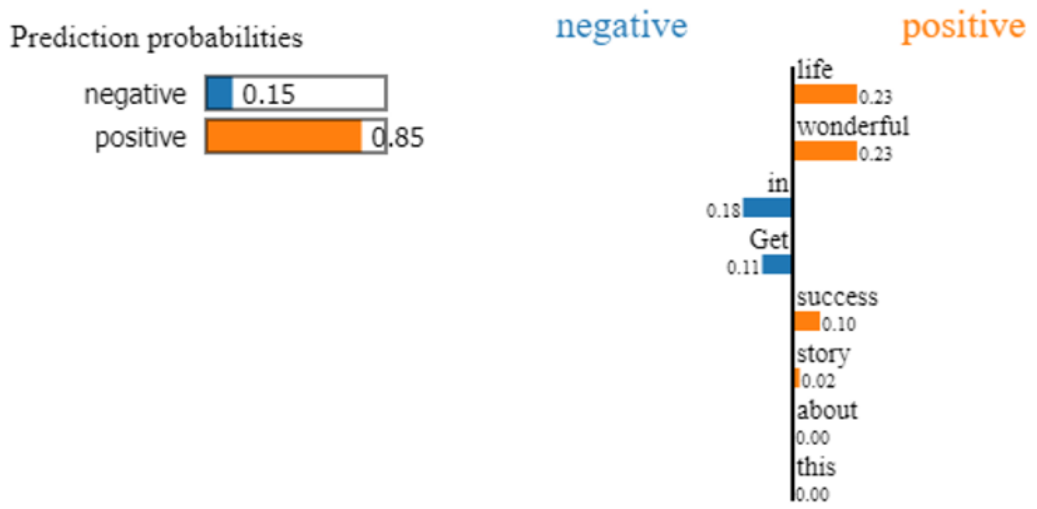
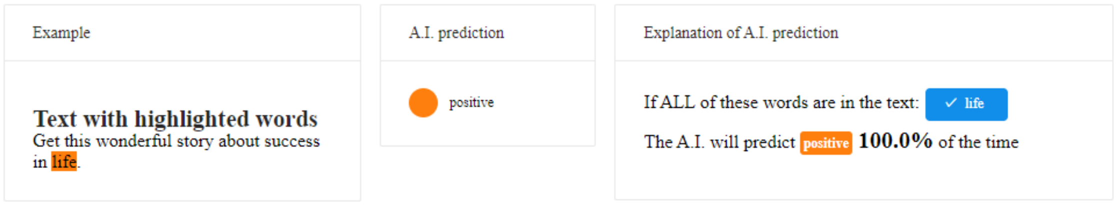


Figure 5.2: The explanation provided by LIME for an instance of the IMDB movie review dataset labeled as positive by the black-box model. Top-left horizontal bar chart presents the prediction probabilities of the black-box model. The bar chart located in the top-right displays the importance scores for each feature generated by LIME, with the features sorted in descending order of their respective importance scores. A higher absolute value in this chart indicates that LIME assigns greater importance to that feature in the classification process.

We use the same sentence to generate explanation from the same black-box model using Anchor as the underlying explainer. Figure 5.3 presents the explanation generated by Anchor. Unfortunately, Anchor is able to predict only a single word "life" as the rule, and nothing more than that. In the given context, it seems that Anchor's



Example

Text with highlighted words
Get this wonderful story about success in life.

A.I. prediction

● positive

Explanation of A.I. prediction

If ALL of these words are in the text: ✓ life

The A.I. will predict positive **100.0%** of the time

Figure 5.3: The explanation provided by Anchor for an instance of the IMDB movie review dataset labeled as positive by the black-box model.

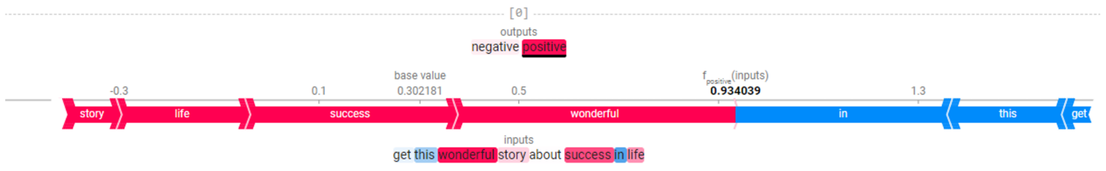


Figure 5.4: The explanation provided by SHAP for an instance of the IMDB movie review dataset labeled as positive by the black-box model. The red color in the figure represents features that have contributed positively to the prediction. Features in blue have contributed negatively to the prediction.

output fails to identify some important words that were identified by BARBE-S and LIME. Specifically, the words “wonderful” and “success” are more logically associated with positive sentiment in the context of the sentence. Anchor has generated neither a set of rules nor a conjunction of features. On the other hand, BARBE-S generates a set of rules as the explanation as shown in Figure 5.1. Like LIME, BARBE-S assigns an important score to the features extracted from the rules. But the beauty of BARBE-S is the generation of a set of rules, where each rule contains a set of features, and each rule comes with support, confidence, and statistical significance values. These additional statistics provide more information to the users to get a better understanding of the explanation generated by BARBE-S.

Finally, we compare the explanation generated by BARBE-S with SHAP. Again, we have used the same black-box and the same sentence to generate explanation by SHAP. Figure 5.4 presents the explanation generated by SHAP for the sentence: “Get this wonderful story about success in life”. SHAP aims to provide an explanation of the requested instance by determining the contribution of each feature towards determining the predicted sentiment of the text sample. The plot allows users to interpret the contribution of each feature to the sentiment prediction of the provided sentence, giving insights into the factors that influence sentiment classification. The red color words: "story", "life", "success", and "wonderful" represent the features that have contributed positively to the prediction of the black-box model. Whereas, the blue color words: "in", "this", and "get" represent the features that have contributed negatively to the prediction of the black-box model. The higher the length of the bar along the horizontal axis, the greater the importance of the word towards the prediction of the black-box model. It is evident from Figure 5.4 that the words: "wonderful" and "success" have more importance in the decision of the black-box model. SHAP is capable of identifying features in a manner similar to BARBE-S. Similar to LIME, SHAP generates features as explanations and assigns a Shapley value to each feature. However, unlike BARBE-S, SHAP does not generate rule-based explanations, which may be more interpretable and understandable for the end user. Therefore, while SHAP and LIME can provide a quantitative assessment of feature importance, BARBE-S provides a more rule-based and interpretable explanation of the model's decision-making process.

We demonstrate the explanation produced by BARBE-S for a sentence having negative sentiment in Figure 4.2 and Table 4.1. Figure 4.2A presents the sentence "A movie where tensions build and conflicts arise" having a negative sentiment as labeled by the black-box model. BARBE-S generates 5 rules to explain why the sentence has been labeled as negative by the black-box model as shown in Table 4.1. The rules are sorted in this table based on their confidence values. BARBE-S employs a red gradient to indicate the importance of the negative words within a sentence, with stronger shades of red representing the most important words for making the sentence negative as labeled by the black-box model, and lighter shades indicating less important words. Since the word "conflicts" is available in 2 out of 5 rules, it

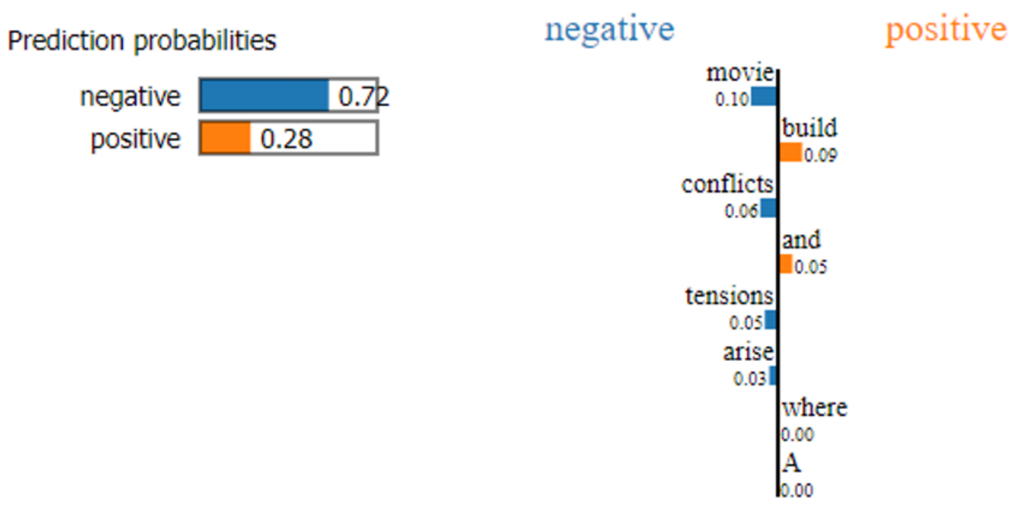


Figure 5.5: The explanation provided by LIME for an instance of the IMDB movie review dataset labeled as negative by the black-box model. Top-left horizontal bar chart presents the prediction probabilities of the black-box model. The bar chart located in the top-right displays the importance scores for each feature generated by LIME, with the features sorted in descending order of their respective importance scores. A higher absolute value in this chart indicates that LIME assigns greater importance to that feature in the classification process.

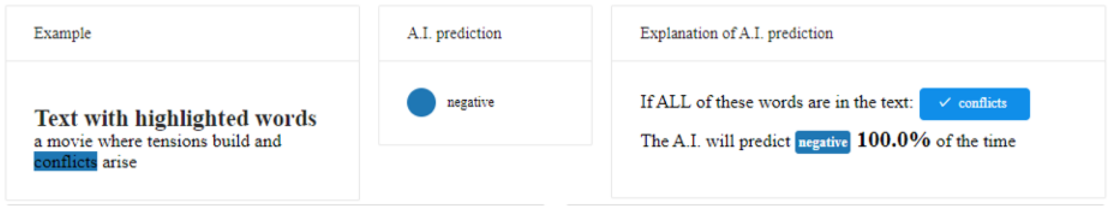

Example
Text with highlighted words
a movie where tensions build and conflicts arise
A.I. prediction
negative
Explanation of A.I. prediction
If ALL of these words are in the text: ✓ conflicts
The A.I. will predict negative 100.0% of the time

Figure 5.6: The explanation provided by Anchor for an instance of the IMDB movie review dataset labeled as negative by the black-box model.

is highlighted with strong red by BARBE-S. Other words: “movie”, “tensions” and “arise” are assigned a similar color since each of them is present in 1 out of 5 rules. The horizontal bar chart in Figure 4.2B is the prediction probabilities of the black-box model and the vertical bar chart in Figure 4.2C highlights the most important features in terms of their frequency within the rules along with the corresponding weighted confidence value.

The explanation produced by LIME and Anchor is presented in Figure 5.5 and Figure 5.6 respectively. The explanation has been generated for the same sentence using the same black-box used for BARBE-S. LIME highlights “movie” having an

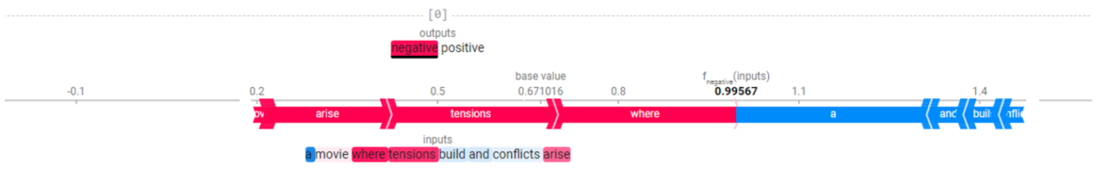


Figure 5.7: The explanation provided by SHAP for an instance of the IMDB movie review dataset labeled as negative by the black-box model. The red color in the figure represents features that have contributed positively to the prediction. Features in blue have contributed negatively to the prediction.

importance score of 0.10 as the most significant feature for contributing negatively to the prediction of the black-box model. Other significant words selected by LIME are "conflicts" and "tensions". On the other hand, BARBE-S identifies the words "conflicts" and "tensions" as the most significant features, based on the generated rules. The word "movie" might not be the significant contributing factor in impacting the decision of the black-box model since a movie can appear with a positive adjective or a negative adjective. The explanation provided by Anchor as shown in Figure 5.6 only detects the word "conflicts" as the contributing factor. Anchor completely misses the other key important words like "tensions". Figure 5.7 depicts the explanation produced by SHAP. It is noteworthy that, according to SHAP, the words "tensions" and "where" are the most important contributors that strongly align with the decision of the black-box model. Additionally, SHAP identifies the word "arise" as a slightly less significant contributor to the model's decision. The contribution of the word "where" does not make much sense in the context of the sentiment of a sentence. Furthermore, SHAP is unable to identify the word "conflicts" as a key contributor to the model's decision, unlike BARBE-S, LIME, and Anchor, which all detect this word as one of the most important in impacting the decision of the black-box model. SHAP is subject to the same limitations as Shapley values, including the potential for misinterpretation and the requirement for data access in order to compute them for new data. Besides, it is possible to intentionally manipulate the SHAP interpretation method to generate misleading explanations that may hide biases in the underlying model [106].

We have demonstrated how efficiently BARBE-S identifies the set of rules that can explain the prediction of a black-box model for the binary classification task. The sentences used by BARBE-S are shorter in terms of the number of words. We cannot use long sentences for BARBE-S because of its limitation in handling long sentences since long sentences have a large feature vector space which is not effectively processed by SigDirect, leading to high memory usage and long run time. Such limitation has been addressed by BARBE-C which uses an ensemble of base learners, with each base learner being a SigD2 classifier that processes a subset of features from the large feature vector space. This approach enables BARBE-C to generate explanations from long sentences effectively.

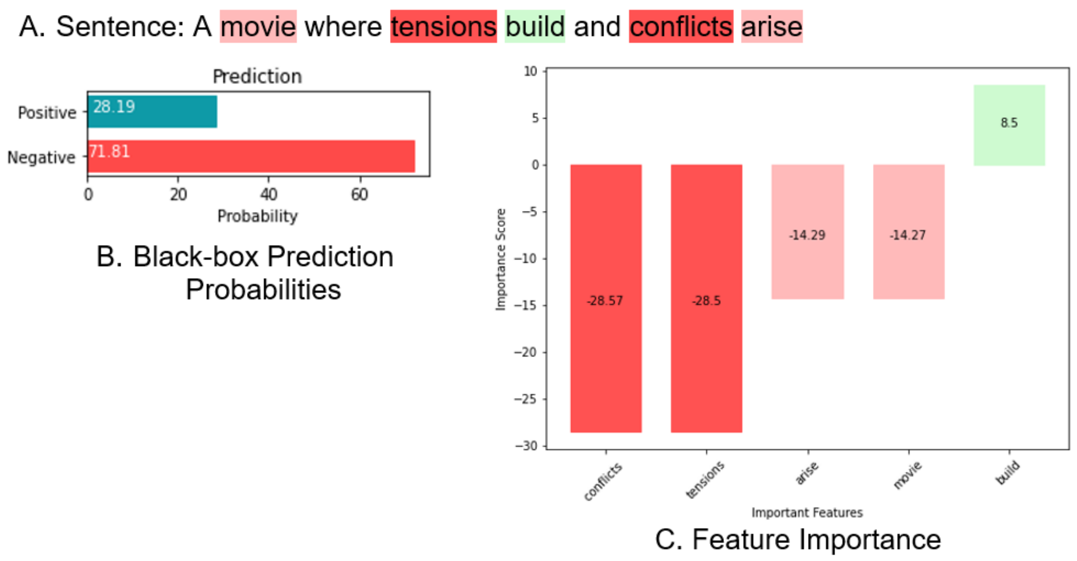


Figure 5.8: The explanation provided by BARBE-C for an instance of the IMDB movie review dataset labeled as negative by the black-box model. (A) shows the sentence with features highlighted. The red heatmap presents the negative words and the green heatmap presents the positive ones. Table 5.2 presents the set of important rules with their support, confidence, and logarithm of statistical significance values. (B) presents the prediction probability of the black-box, and (C) presents the histogram of important features ranked based on their importance.

Prior to demonstrating the explanation generated by BARBE-C for long sentences, we want to investigate the efficacy of BARBE-C on short sentences. This would provide a comparative analysis between BARBE-S and BARBE-C in handling short sentences. The same sentence, "A movie where tensions build and conflicts arise" is utilized to generate an explanation for BARBE-C, as it has been used for BARBE-S. Figure 5.8 illustrates the explanation produced by BARBE-C for this sentence. It is worth noting that BARBE-C is able to generate 2 more rules with higher confidence if compared to BARBE-S. The 2 new rules are: $tensions\ conflicts \rightarrow Negative$ and $tensions\ build \rightarrow Negative$. These two rules are not explored by BARBE-S. Besides, the confidence values of other rules are also higher for BARBE-C. Table 5.2 presents the set of rules generated by BARBE-C. This experimental result clarifies that both BARBE-C and BARBE-S are capable of generating explanations for short sentences. However, BARBE-C is more efficient in this scenario because of its ability to generate more rules with high confidence.

Figure 5.9 demonstrates the result obtained by BARBE-C for a long sentence labeled as positive by the black-box model. The sentence is shown in Figure 5.9A. Similar to BARBE-S, the strong green gradient highlights the words responsible for making the sentence positive labeled by the black-box model. It is worth noting that, BARBE-C is capable of generating more rules for long sentences as presented in Table 5.3. For space limitation in a table cell, we have truncated some parts of the text in the Sentence column of the table. Here, BARBE-C generates a total of 8 rules for the sentence: "This is a fantastic movie of three prisoners who become famous. George Clooney is awesome. Another good thing about the movie is the soundtrack. I recommend this movie to everybody." Typically, this sentence seems to have a positive sentiment because the words used in the sentence such as "fantastic", "awesome", "good" and "recommend" are all positive words that convey a positive sentiment. The phrase "recommend this movie to everybody" implies a positive evaluation of the movie overall. BARBE-C is able to generate an explanation from this long sentence and successfully highlights the positive words e.g. "good", "awesome", "fantastic", "recommend", etc. All the rules have higher confidence values especially

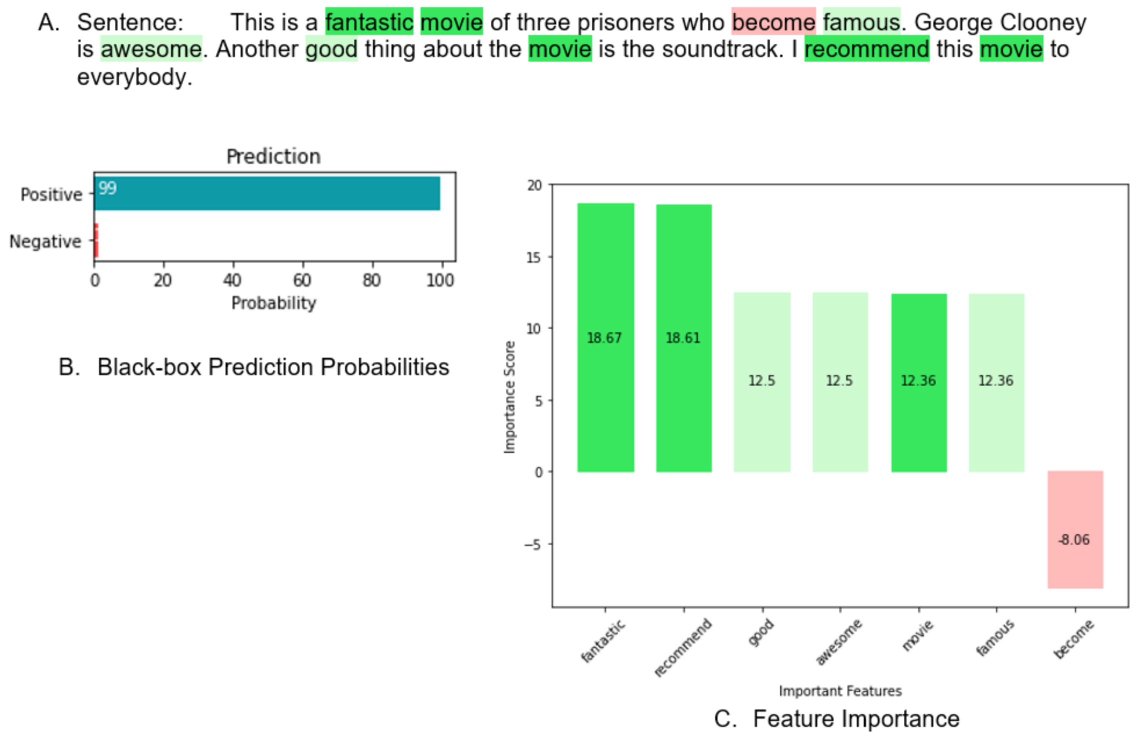
Table 5.2: Set of rules generated by BARBE-C with their support, confidence, and logarithm of statistical significance values along with the sentence with features highlighted. Rules are sorted in this table based on their confidence values. “N” denotes “Negative” label and “P” denotes “Positive” label in the Rule column. Rows with grey backgrounds indicate additional rules generated by BARBE-C, not present in BARBE-S.

Sentence	**Rule**	**Support**	**Confidence**	**Log(SS)**
A movie where tensions build and conflicts arise	$conflicts \rightarrow N$	50.08	100.00%	-142.75
A movie where tensions build and conflicts arise	$conflicts$ $arise \rightarrow N$	28.21	100.00%	-67.64
A movie where tensions build and conflicts arise	$tensions \rightarrow N$	50.51	100.00%	-144.56
A movie where tensions build and conflicts arise	$tensions$ $conflicts$ $\rightarrow N$	11.15	100.00%	-24.04
A movie where tensions build and conflicts arise	$movie \rightarrow N$	47.88	99.92%	-133.77
A movie where tensions build and conflicts arise	$tensions$ $build \rightarrow N$	28.77	98.97%	-69.24
A movie where tensions build and conflicts arise	$build \rightarrow P$	5.01	10.00%	-141.49

the first two rules have 100% confidence as presented in Table 5.3. There is a rule *become* $\rightarrow$ *Negative* which does not agree with the decision of the black-box model and appears with a smaller confidence value of 64.50%. Like, BARBE-S, BARBE-C is also capable of identifying rules containing conjunction of words that convey more semantic meaning and better understandability. As an example, the rule *fantastic movie* $\rightarrow$ *Positive* tells the user that the movie is fantastic. Similarly, the rule *recommend movie* $\rightarrow$ *Positive* tells the user about recommending a movie. Such rules convey more semantic context.

The prediction probability of the black-box model is shown in Figure 5.9B. Finally, Figure 5.9C presents the most important features based on their frequency within the rules weighted by the confidence value. Features like "fantastic", "recommend", "good", "awesome" have got higher importance according to BARBE-C. We compare the explanation of BARBE-C with LIME, Anchor, and SHAP in Figure 5.10, Figure 5.11 and Figure 5.12 respectively. LIME is able to identify the features "fantastic", "awesome", "recommend" like BARBE-C. But it has assigned a higher importance to the preposition "to" and a lower importance to the adjective "good" which may not seem logical in the context of positive sentiment labeled by the black-box. Anchor is able to identify the feature "fantastic" only with a higher probability as shown in Figure 5.11. But it completely misses a lot of other important words in the sentence. A single rule with a single word is not sufficient to justify the decision of the black-box model. SHAP is not able to identify the features like BARBE-C and LIME do. Overall, the rules presented by BARBE-C are more accurate and meaningful as compared with the other explainers.

Figure 5.13 presents the result generated by BARBE-C for the sentence: "The movie was frustrating with weird looping, hated characters, annoying little girl, and a final scene that was bad with horrible dancing." labeled as negative by the black-box model. Table 5.4 presents the rules generated by BARBE-C for this sentence. BARBE-C has identified the features such as "hated", "weird", "horrible", "frustrat-



A. Sentence: This is a fantastic movie of three prisoners who become famous. George Clooney is awesome. Another good thing about the movie is the soundtrack. I recommend this movie to everybody.

Figure 5.9: The explanation provided by BARBE-C for an instance of the IMDB movie review dataset labeled as positive by the black-box model. (A) shows the sentence with features highlighted. The green heatmap presents the positive words and the red heatmap presents the negative ones. Table 5.3 presents the set of important rules with their support, confidence, and logarithm of statistical significance values. (B) presents the prediction probability of the black-box, and (C) presents the histogram of important features ranked based on their importance.

Table 5.3: Set of rules generated by BARBE-C with their support, confidence, and logarithm of statistical significance values along with the sentence with features highlighted. Rules are sorted in this table based on their confidence values. “P” denotes “Positive” label and “N” denotes “Negative” label in the Rule column. Sentence has been truncated and includes an ellipsis to indicate that there is more to the sentence.

Sentence	**Rule**	**Support**	**Confidence**	**Log(SS)**
This is ... Another good thing about the movie is the soundtrack. I recommend ...	$good \rightarrow P$	22.52	100.00%	-41.11
This is ... George Clooney is awesome. Another good ...	$awesome \rightarrow P$	23.70	100.00%	-78.44
This is a fantastic movie of three prisoners who become famous. George Clooney ...	$fantastic \rightarrow P$	32.39	99.80%	-61.40
This is ... I recommend this movie to everybody.	$recommend \rightarrow P$	23.37	99.60%	-66.14
This is a fantastic movie of three prisoners who become famous. George Clooney ...	$fantastic\ movie \rightarrow P$	27.69	99.21%	-475.12
This is a fantastic movie of three prisoners who become famous. George Clooney ...	$famous \rightarrow P$	13.69	98.93%	-34.62
This is ... I recommend this movie to everybody.	$recommend\ movie \rightarrow P$	14.39	98.58%	-62.74
This is a fantastic movie of three prisoners who become famous. George Clooney ...	$become \rightarrow N$	18.03	64.50%	-9.63

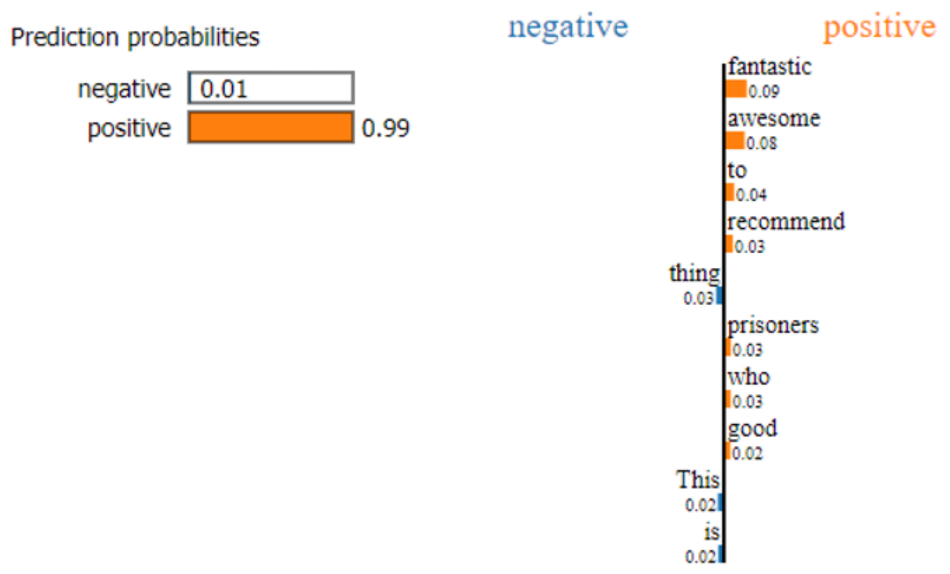
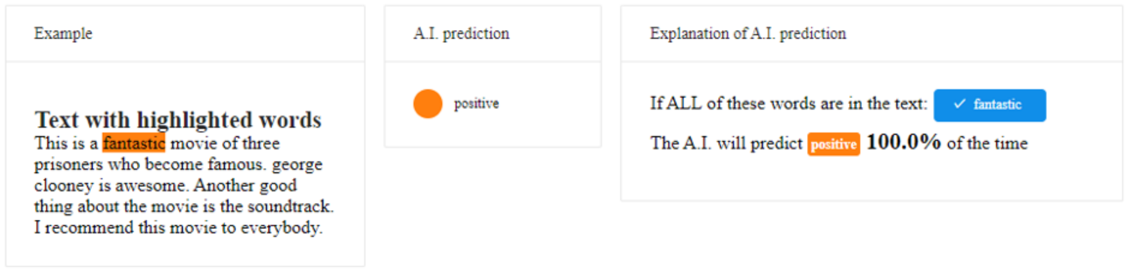


Figure 5.10: The explanation provided by LIME for an instance of the IMDB movie review dataset labeled as positive by the black-box model. Top-left horizontal bar chart presents the prediction probabilities of the black-box model. The bar chart located in the top-right displays the importance scores for each feature generated by LIME, with the features sorted in descending order of their respective importance scores.


Example

Text with highlighted words
This is a fantastic movie of three prisoners who become famous. george clooney is awesome. Another good thing about the movie is the soundtrack. I recommend this movie to everybody.

A.I. prediction

● positive

Explanation of A.I. prediction

If ALL of these words are in the text: ✓ fantastic

The A.I. will predict positive **100.0%** of the time

Figure 5.11: The explanation provided by Anchor for an instance of the IMDB movie review dataset labeled as positive by the black-box model.

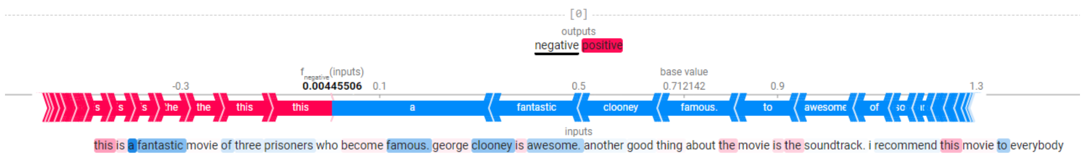


Figure 5.12: The explanation provided by SHAP for an instance of the IMDB movie review dataset labeled as positive by the black-box model. The red color in the figure represents features that have contributed positively to the prediction. Features in blue have contributed negatively to the prediction.

ing", "bad", "annoying" that are responsible for labeling the sentence as negative by the black-box model. Rules like *weird looping* $\rightarrow$ *Negative*, *horrible dancing* $\rightarrow$ *Negative* contain a conjunction of features that convey more semantic context to explain the decision of the black-box model. Moreover, rules like *hated* $\rightarrow$ *Negative*, *hated characters* $\rightarrow$ *Negative*, *frustrating* $\rightarrow$ *Negative*, *bad* $\rightarrow$ *Negative* have 100% confidence as presented in Table 5.4. We compare the explanation of BARBE-C with LIME, Anchor, and SHAP in Figure 5.14, Figure 5.15, and Figure 5.16 respectively. Most of the features identified by LIME match with the features presented within the rules generated by BARBE-C although feature like "was" has got higher importance by LIME that seem inconsistent. Anchor has been able to identify only one feature "bad" with 98.7% probability but fails to detect other important features as depicted in Figure 5.15. Although SHAP is able to identify a few important features such as "annoying", "horrible", "bad" as shown in the red gradient in Figure 5.16, it falls short compared to BARBE-C and LIME in terms of identifying other significant features.

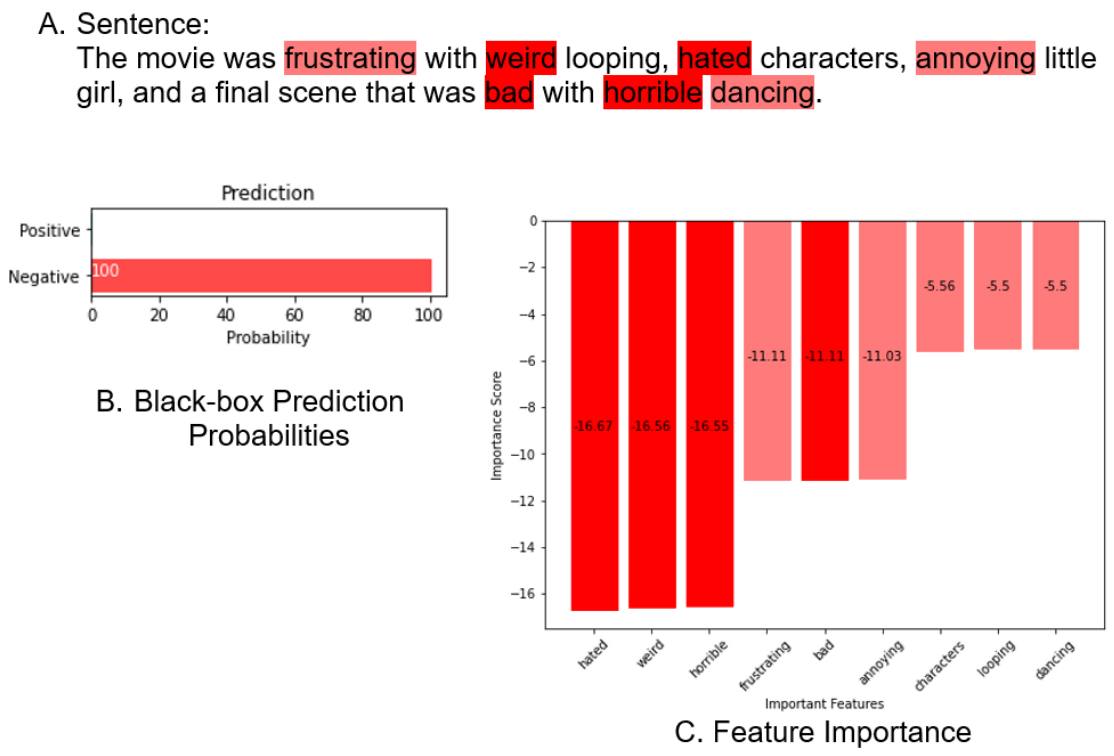


Figure 5.13: The explanation provided by BARBE-C for an instance of the IMDB movie review dataset labeled as negative by the black-box model. (A) shows the sentence with features highlighted. The red heatmap presents the negative words. Table 5.4 presents the set of important rules with their support, confidence, and logarithm of statistical significance values. (B) presents the prediction probability of the black-box, and (C) presents the histogram of important features ranked based on their importance.

Table 5.4: Set of rules generated by BARBE-C with their support, confidence, and logarithm of statistical significance values along with the sentence with features highlighted. Rules are sorted in this table based on their confidence values. "N" denotes "Negative" label in the Rule column. Sentence has been truncated and includes an ellipsis to indicate that there is more to the sentence.

Sentence	**Rule**	**Support**	**Confidence**	**Log(SS)**
The movie was frustrating with weird looping, hated characters, ...	*hated* $\rightarrow$ *N*	41.33	100.00%	-41.11
The movie was frustrating with weird looping, hated characters, annoying little girl, ...	*hated characters* $\rightarrow$ *N*	23.97	100.00%	-61.63
The movie was frustrating with weird looping, ...	*frustrating* $\rightarrow$ *N*	13.69	100.00%	-34.62
The movie was frustrating ..., and a final scene that was bad with horrible dancing.	*bad* $\rightarrow$ *N*	23.70	100.00%	-75.99
The movie was frustrating with weird looping, ...	*weird* $\rightarrow$ *N*	42.54	99.59%	-71.22
The movie was ... and a final scene that was bad with horrible dancing.	*horrible* $\rightarrow$ *N*	12.32	99.45%	-31.39
The movie was frustrating with weird looping, hated characters, annoying little girl, ...	*annoying* $\rightarrow$ *N*	25.57	99.23%	-167.69
The movie was frustrating ..., and a final scene that was bad with horrible dancing.	*horrible dancing* $\rightarrow$ *N*	8.14	99.01%	-33.56
The movie was frustrating with weird looping, hated characters, ...	*weird looping* $\rightarrow$ *N*	27.56	98.92%	-72.23

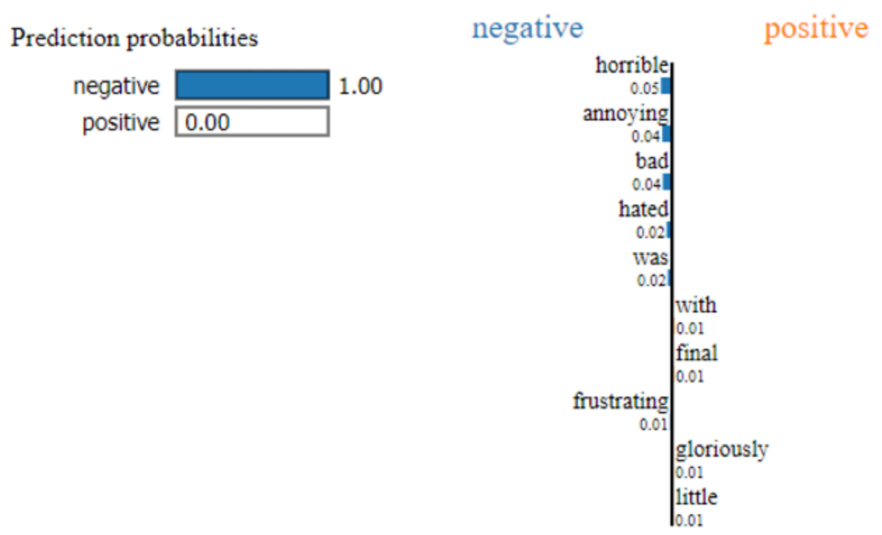
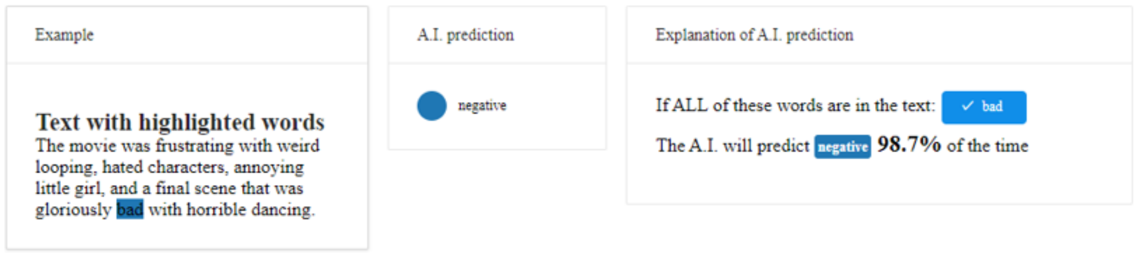


Figure 5.14: The explanation provided by LIME for an instance of the IMDB movie review dataset labeled as negative by the black-box model. Top-left horizontal bar chart presents the prediction probabilities of the black-box model. The bar chart located in the top-right displays the importance scores for each feature generated by LIME, with the features sorted in descending order of their importance scores.



Example

Text with highlighted words
The movie was frustrating with weird looping, hated characters, annoying little girl, and a final scene that was gloriously bad with horrible dancing.

A.I. prediction

● negative

Explanation of A.I. prediction

If ALL of these words are in the text: ✓ bad

The A.I. will predict negative **98.7%** of the time

Figure 5.15: The explanation provided by Anchor for an instance of the IMDB movie review dataset labeled as negative by the black-box model.

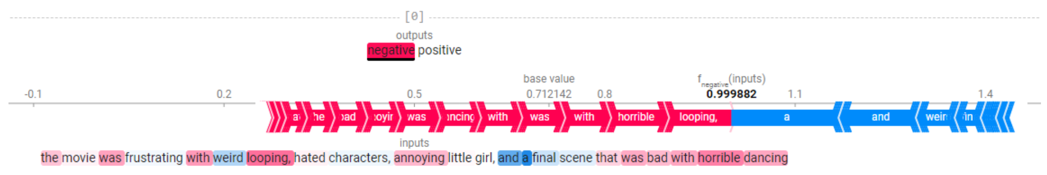


Figure 5.16: The explanation provided by SHAP for an instance of the IMDB movie review dataset labeled as positive by the black-box model. The red color in the figure represents features that have contributed positively to the prediction. Features in blue have contributed negatively to the prediction.

5.3.2 Experiments on Multiclass Classification Task

In this Section, we want to explore the performance of BARBE-S and BARBE-C in explaining the prediction of the black-box model for multiclass classification tasks. We are going to use CNN as the black-box model that has been trained with AG's news topic classification dataset. Figure 5.17 illustrates the explanation generated by BARBE-S for an instance of the news dataset labeled as "Sport" by the black-box model. The sentence: "He could not record goal because of injury" is shown in Figure 5.17A with the features highlighted. Blue shade, green shade, yellow shade, and red shade represent "Sports", "Business", "World" and "Sci/Tech" classes respectively as labeled by the black-box model. Figure 5.17B displays the prediction probability of the black-box model. The color of the horizontal bar chart in Figure 5.17B represents the corresponding classes. Class "Sports" achieves the highest probability here. Following that, BARBE-S is able to detect 2 out of 4 words in the sentence that belongs to class "Sports". Words such as "goal" and "injury" are more closely associated with sports. The word "record" belongs to the class "Business" and "because" belongs to the class "World" as highlighted by BARBE. Figure 5.17C displays the important features ranked based on their important score. Table 5.5 shows the set of rules generated by BARBE-S with their support, confidence, and logarithm of statistical significance values.

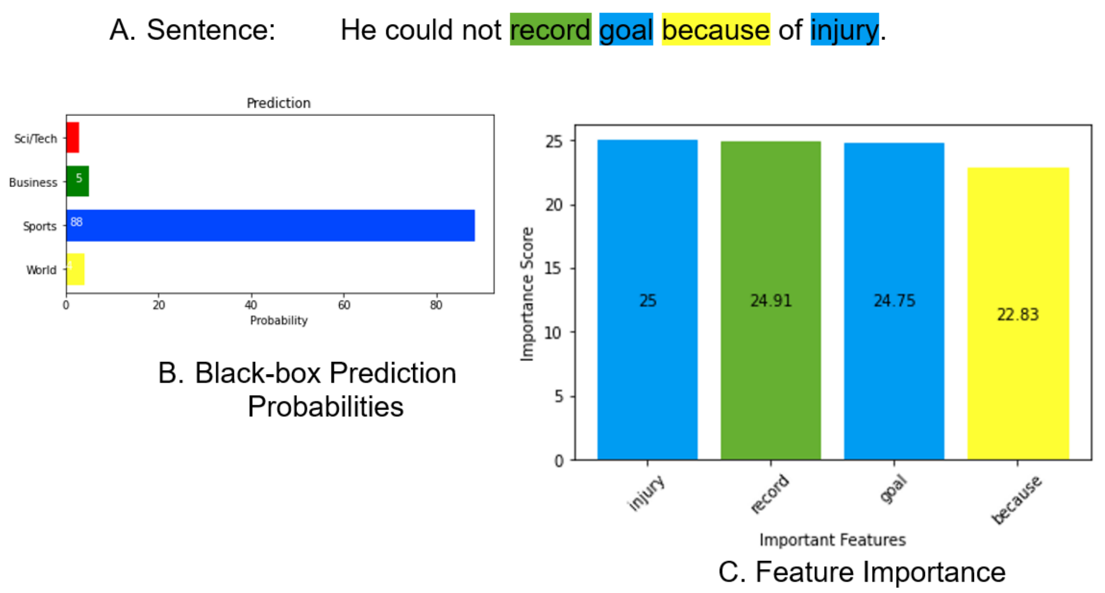


Figure 5.17: The explanation provided by BARBE-S for an instance of the news classification dataset. (A) shows the sentence with features highlighted. Table 5.5 presents the set of important rules with their support, confidence, and logarithm of statistical significance values. (B) presents the prediction probability of the black-box, and (C) presents the histogram of important features ranked based on their importance.

Table 5.5: Set of rules generated by BARBE-S with their support, confidence, and logarithm of statistical significance values along with the sentence with features highlighted. Rules are sorted in this table based on their confidence values. "S" denotes "Sports", "B" denotes "Business", "W" denotes "World" label in the Rule column.

Sentence	**Rule**	**Support**	**Confidence**	**Log(SS)**
He could not record goal because of injury	*injury* $\rightarrow S$	23.49	100.00%	-119.32
He could not record goal because of injury	*record* $\rightarrow B$	22.85	99.63%	-77.29
He could not record goal because of injury	*goal* $\rightarrow S$	21.31	99.00%	-118.70
He could not record goal because of injury	*because* $\rightarrow W$	19.80	91.33%	-76.66

We compare the explanation generated by BARBE-S for multiclass classification with LIME, Anchor, and SHAP. Figure 5.18 presents the explanation produced by

LIME for the same sentence: “He could not record goal because of injury”. LIME generates one chart for each class representing the words on the right side of the line are positive, and the words on the left side are negative. Like BARBE-S, LIME identifies the features “goal” and “injury” mostly belong to the “Sports” class. Similarly, “record” belongs to the “Business” class. One advantage of BARBE-S is that instead of generating the explanation as “one-vs-rest” approach, it simply handles multiclass natively. Like binary text classification tasks, BARBE-S can generate rules for multiclass classification tasks as well where each rule comes with support, confidence, and statistical significant values. BARBE-S simply assigns the class name on the right-hand side of each rule. Figure 5.19 shows the explanation provided by Anhcor for the same sentence. Anchor is able to identify only a single feature “injury” for which the black-box is labeling the sentence as “Sports” class. But it misses other significant features as identified by BARBE-S and LIME. Figure 5.20 presents the explanation generated by SHAP. It follows the “one-vs-rest” approach like LIME to identify the features impacting the decision of the black-box model. SHAP is capable of identifying specific words such as “goal” and “injury” that contribute to a sentence being classified as “Sports” class. But it is not effective in identifying meaningful features for the other classes.

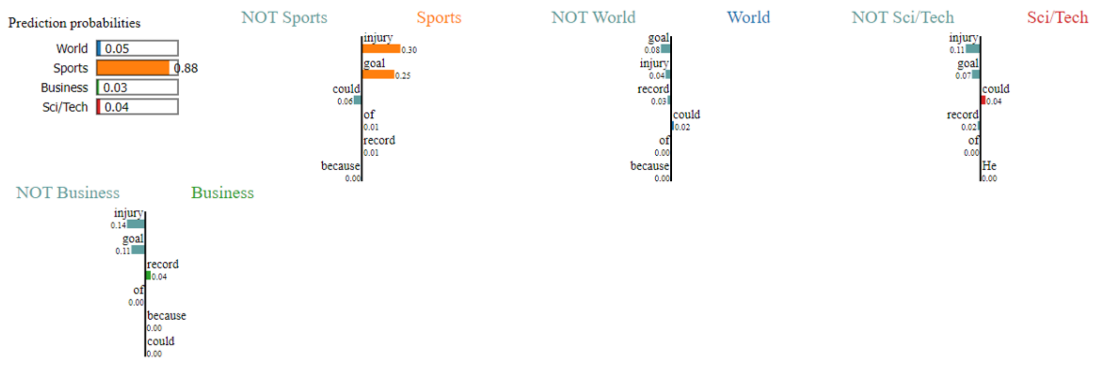
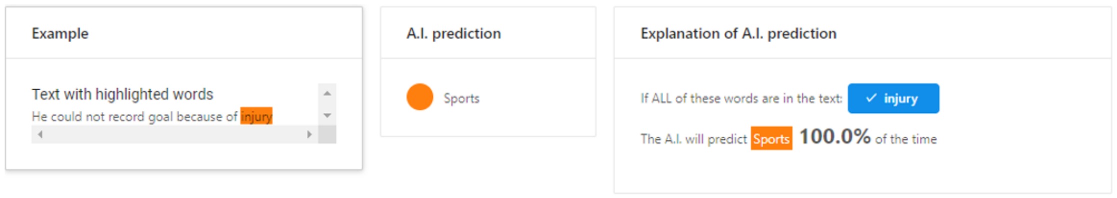


Figure 5.18: The explanation provided by LIME for an instance of the news classification dataset. Top-left horizontal bar chart presents the prediction probabilities of the black-box model for all the classes. The other charts display the importance scores for each feature generated by LIME, with the features sorted in descending order of their respective importance scores. For each class, the words on the right side of the line are positive, and the words on the left side are negative.


Example

Text with highlighted words

He could not record goal because of injury

A.I. prediction

● Sports

Explanation of A.I. prediction

If ALL of these words are in the text: ✓ injury

The A.I. will predict Sports **100.0%** of the time

Figure 5.19: The explanation provided by Anchor for an instance of the news classification dataset.

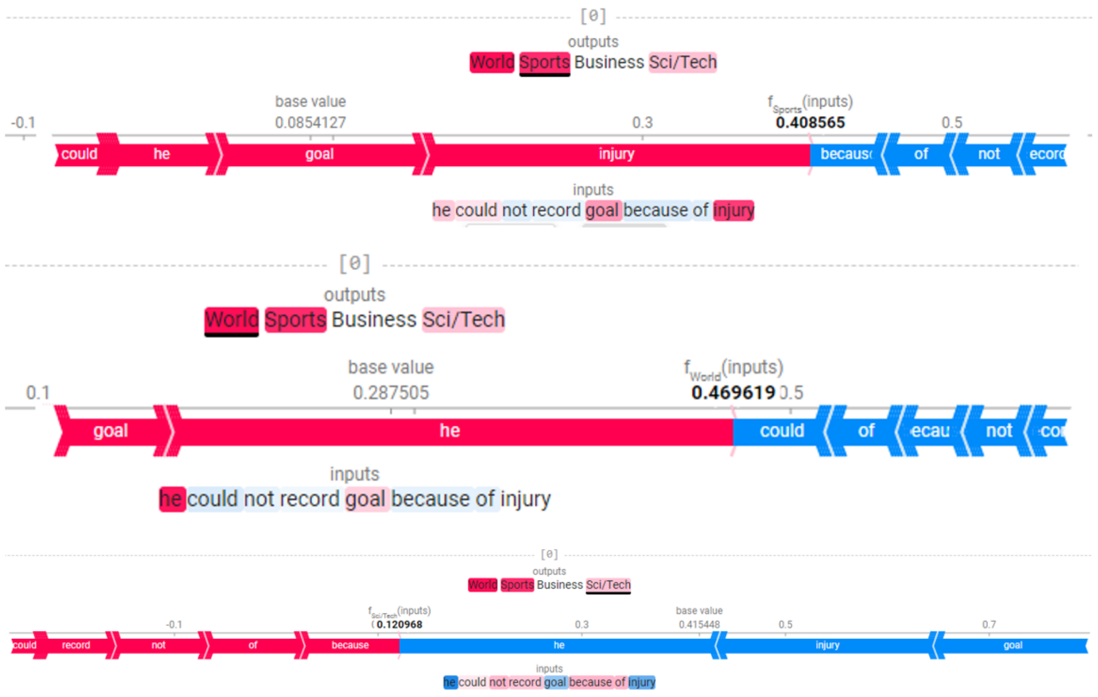


Figure 5.20: The explanation provided by SHAP for an instance of the news classification dataset. The red color in the figure represents features that have contributed positively to the prediction. Features in blue have contributed negatively to the prediction.

The sentence we used to generate explanation by BARBE-S for the multiclass classification task is short in length. In this regard, we again demonstrate the effectiveness of BARBE-C in generating explanation for a long sentence: "Nike, being a leading sportswear brands, is anticipating boost in profits in the next quarter by sponsoring England team for scoring tremendous performance in last league." in Figure 5.21. BARBE-C identifies the features within the sentence from "Sports", "Business", "Sci/Tech" classes. The same color to distinguish the different classes has been used for both BARBE-S and BARBE-C. Figure 5.21A displays the sentence with features highlighted. Figure 5.21B shows the prediction probabilities of all four classes. Finally, Figure 5.21C ranks the important features based on their important score. Table 5.6 presents the set of rules generated by BARBE-C with their support, confidence, and logarithm of statistical significance values. BARBE-C generates a total of 13 rules to explain the sentence as labeled by the black-box model. A good number of rules have 100% confidence as reported by BARBE-C in Table 5.6. Rules like $tremendousperformance \rightarrow Sports$, $nextquarter \rightarrow Sci/Tech$, $boostprofits \rightarrow Business$ contain a conjunction of features. The same sentence has been used to generate explanation using LIME, Anchor and SHAP in Figure 5.22, Figure 5.23, and Figure 5.24 respectively. BARBE-C identifies the features that contribute to the decision of the black-box model and presents them simply like it does for binary classification tasks.

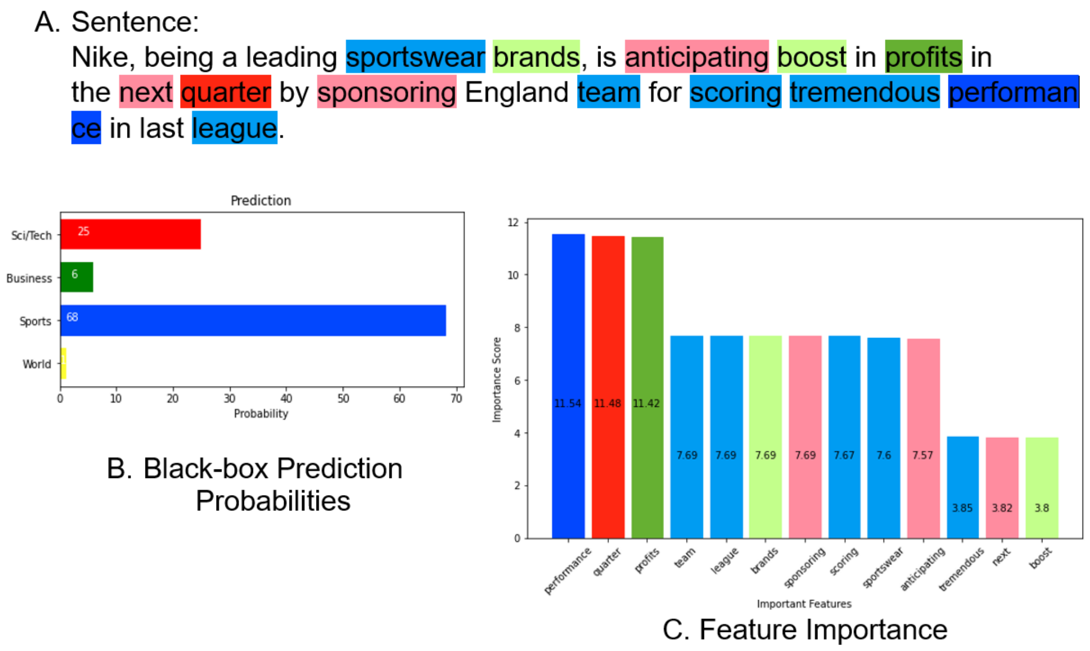

A. Sentence:
Nike, being a leading sportswear brands, is anticipating boost in profits in the next quarter by sponsoring England team for scoring tremendous performan ce in last league.

Figure 5.21: The explanation provided by BARBE-C for an instance of the news classification dataset. (A) shows the sentence with features highlighted. Table 5.6 presents the set of important rules with their support, confidence, and logarithm of statistical significance values. (B) presents the prediction probability of the black-box, and (C) presents the histogram of important features ranked based on their importance.

Table 5.6: Set of rules generated by BARBE-C with their support, confidence, and logarithm of statistical significance values along with the sentence with features highlighted. Rules are sorted in this table based on their confidence values. "S" denotes "Sports", "B" denotes "Business", "T" denotes "Sci/Tech" label in the Rule column. Sentence has been truncated and includes an ellipsis to indicate that there is more to the sentence.

Sentence	**Rule**	**Support**	**Confidence**	**Log(SS)**
Nike, .. England team for scoring ...	$team \rightarrow S$	18.22	100.00%	-215.54
Nike, ... tremendous performance in last ...	$performance \rightarrow S$	17.92	100.00%	-79.31
Nike, ... scoring tremendous performance in last league.	$league \rightarrow S$	19.87	100.00%	-71.62
Nike, ... tremendous performance in last league.	$tremendous\ performance \rightarrow S$	9.57	100.00%	-71.72
Nike, being a leading sportswear brands, is ...	$brands \rightarrow B$	21.44	100.00%	-93.29
Nike, ... next quarter by sponsoring England team ...	$sponsoring \rightarrow T$	15.39	100.00%	-115.21
Nike, ... team for scoring tremendous performance ...	$scoring \rightarrow S$	22.25	99.68%	-213.33
Nike, ... next quarter by sponsoring England team ...	$quarter \rightarrow T$	14.22	99.50%	-115.79
Nike, ... next quarter by sponsoring England team ...	$next\ quarter \rightarrow T$	8.92	99.37%	-108.02
Nike, ... profits in the next quarter ...	$profits \rightarrow B$	18.29	99.05%	-94.19
Nike, being a leading sportswear brands, ...	$sportswear \rightarrow S$	16.51	98.79%	-215.54
Nike, ..., is anticipating boost in profits in the ...	$boost\ profits \rightarrow B$	8.77	98.74%	-93.67
Nike, ..., is anticipating boost in profits ...	$anticipating \rightarrow T$	17.07	98.43%	-108.92

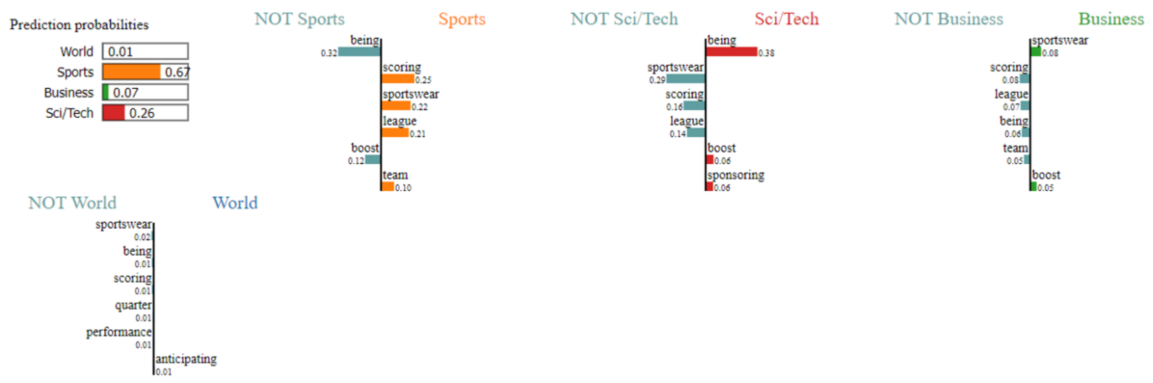
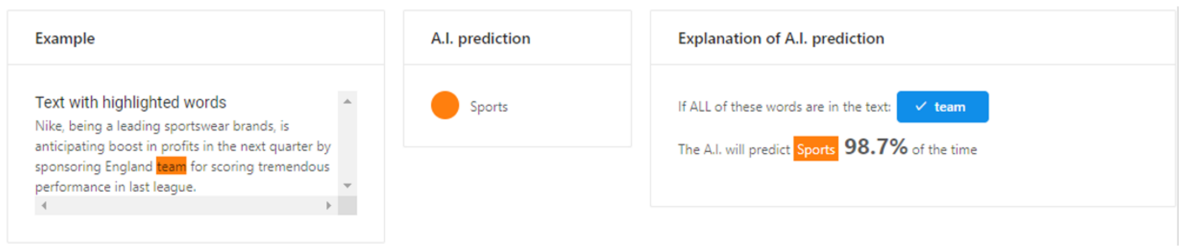


Figure 5.22: The explanation provided by LIME for an instance of the news classification dataset. Top-left horizontal bar chart presents the prediction probabilities of the black-box model for all the classes. The other charts display the importance scores for each feature generated by LIME, with the features sorted in descending order of their respective importance scores. For each class, the words on the right side of the line are positive, and the words on the left side are negative.



Example

Text with highlighted words

Nike, being a leading sportswear brands, is anticipating boost in profits in the next quarter by sponsoring England team for scoring tremendous performance in last league.

A.I. prediction

● Sports

Explanation of A.I. prediction

If ALL of these words are in the text: ✓ team

The A.I. will predict Sports 98.7% of the time

Figure 5.23: The explanation provided by Anchor for an instance of the news classification dataset.

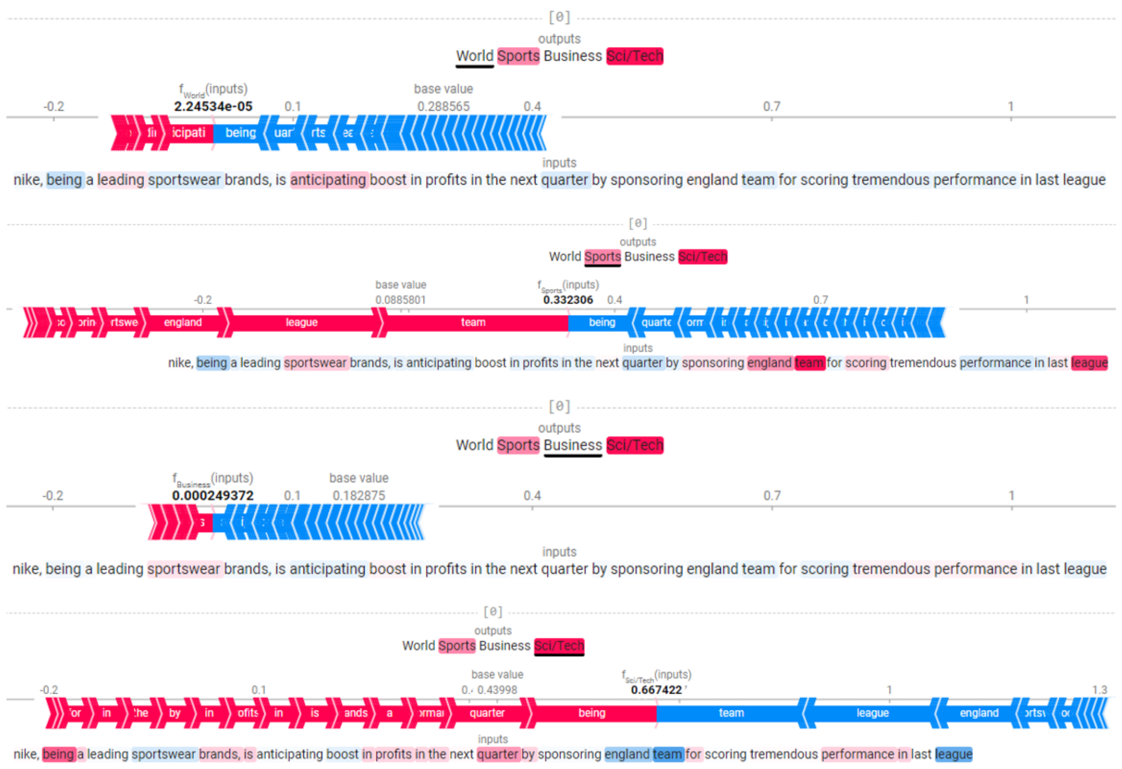


Figure 5.24: The explanation provided by SHAP for an instance of the news classification dataset. The red color in the figure represents features that have contributed positively to the prediction. Features in blue have contributed negatively to the prediction.

5.4 Using Parse Tree for Generating Neighborhood

We have discussed the neighborhood generation technique of BARBE in Section 4.2.2. Both BARBE-S and BARBE-C use the same approach to construct the neighborhood dataset. While generating the neighborhood dataset, we use the Algorithm 1 which mainly focuses on removing words randomly from the sentence to create a new sentence. Random word removal is a technique used for data augmentation in NLP, but it has potential limitations that can impact the completeness of the resulting data. It may remove words that are crucial to the overall meaning of the sentence, resulting in incomplete or inaccurate representations of the text. Moreover, random word removal may not consider the syntactic or semantic relationships between the words in the original sentence, which can further undermine the usefulness of the augmented data for subsequent modeling and analysis. As a result, it is important to be aware of these limitations when using random word removal as a technique for data augmentation and to consider alternative approaches that may improve the overall meaning and structure of the original text.

When implementing the random word removal technique, one way to optimize the process is to utilize a parse tree [107], [108]. A parse tree is a data structure used in NLP to represent the syntactic structure of a sentence. It is a way of breaking down a sentence into its constituent parts, such as words, phrases, and clauses and showing how they are related to one another based on the rules of grammar. Using a parse tree can help ensure that the dropped words do not significantly impact the overall meaning or structure of the sentence. By leveraging the hierarchical representation provided by a parse tree, we can more accurately identify words or phrases that can be safely removed without much impacting the semantic structure of the sentence. Therefore, incorporating a parse tree into the process of random word removal can lead to more effective neighborhood set generation for BARBE.

The parse tree is typically visualized as a tree diagram, where each node represents a constituent part of the sentence, and the branches represent the relationships between the different parts. The root node represents the entire sentence, while the leaf nodes represent individual words. The internal nodes of the tree represent phrases or clauses, and the edges or branches connect these nodes to show the relationships between them. The parse tree is commonly represented as a tree diagram, with each node representing a constituent portion of the phrase and the branches representing the relationships between the various parts. The root node represents the complete sentence, whereas the leaf nodes represent specific words. The tree's internal nodes represent words or clauses, and the edges or branches connect these nodes to demonstrate their relationships. The parse tree is commonly used in various NLP tasks such as syntactic analysis, dependency parsing, and semantic analysis. It can help identify the grammatical structure of a sentence, determine the relationship between different

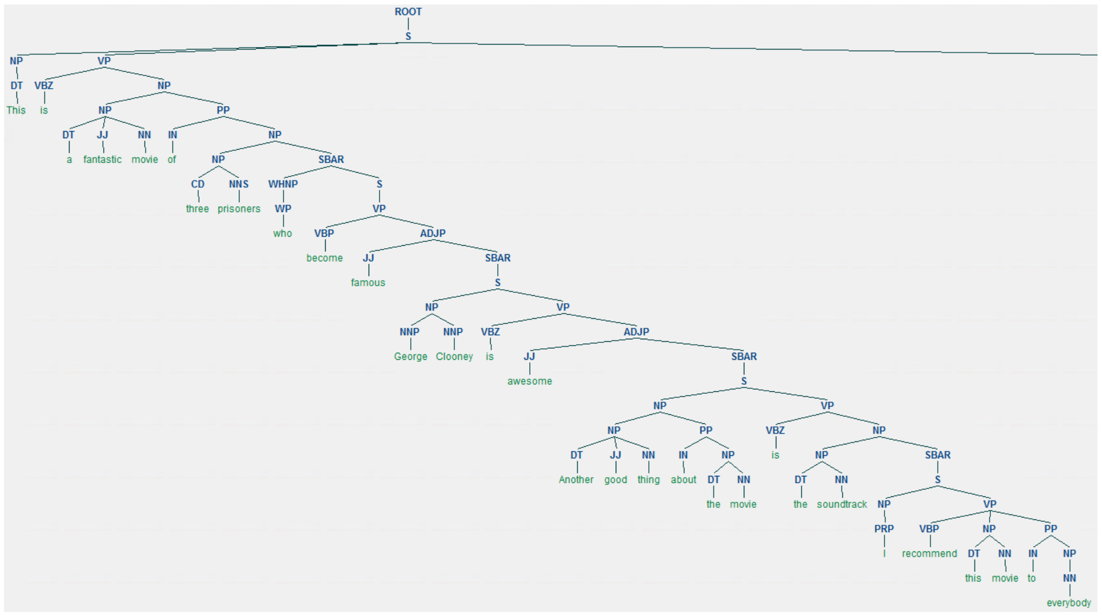


Figure 5.25: Parse tree representation of a sentence.

parts of the sentence, and ultimately aid in the interpretation and understanding of natural language text. Figure 5.25 shows the parse tree representation of a sentence "This is a fantastic movie of three prisoners who become famous. George Clooney is awesome. Another good thing about the movie is the soundtrack. I recommend this movie to everybody.". We have used Stanford CoreNLP [109] to generate the parse tree. The Stanford CoreNLP toolkit uses a combination of rule-based and machine learning-based methods to generate parse trees. It includes a dependency parser that uses a statistical model to predict the relationships between words in a sentence, as well as a constituency parser that uses a combination of rule-based and machine learning-based methods to identify the phrases and clauses in a sentence. It is evident from the tree in Figure 5.25 that the leaf node siblings "a fantastic movie", "three prisoners", "the movie" etc. should stay together. This tells us to consider the leaf sibling words as indivisible units while removing words randomly from the sentence, and if one of the words in the siblings is selected for removal, then the entire siblings should be removed. Conversely, if the leaf sibling is not selected for removal, all the sibling words should remain together in the augmented sentence.

Using a parse tree can be helpful for long sentences. For the case of long sentences, the random word removal technique comprehended by the parse tree can create synthetic sentences that still contain a sufficient number of words from the original sentence. For shorter sentences, this approach does not produce synthetic sentences that contain a sufficient number of words.

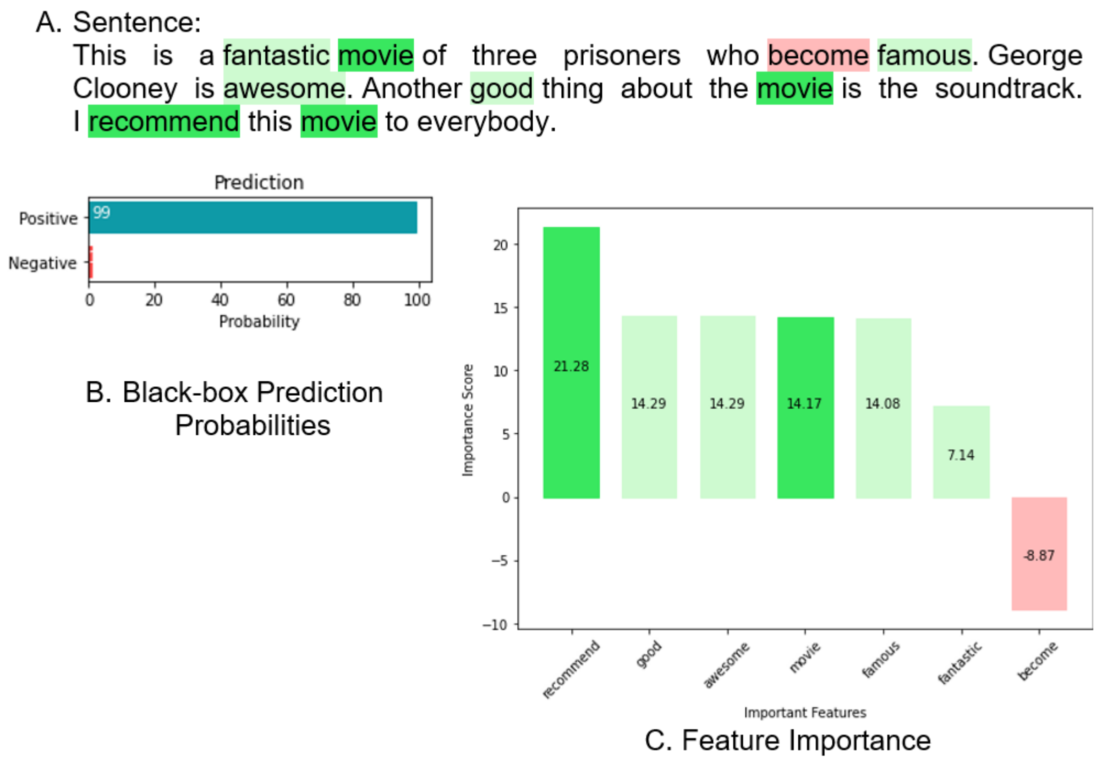
SigDirect, the core of BARBE, is a rule-based classifier that may face limitations in terms of required memory and runtime if the feature vector of the input feature space is large. This is because as the number of features increases (e.g. for long sentences), the number of possible association rules also increases exponentially, which can cause the classifiers to become computationally expensive and require long run time. As a result, the classifiers may not be able to handle large feature vectors efficiently, which can impact their overall performance. Because of such limitations, BARBE is not able to handle long sentences. But this limitation has been overcome by BARBE with CFAR. CFAR employs an ensemble of associative classifiers as base learners, with each base learner trained on a subset of the feature vector. This approach enables CFAR to efficiently handle large input feature space, which is a limitation of the SigDirect classifier. Thus we want to explore the parse tree-based random word removal technique for BARBE with CFAR. We are interested to see how the parse tree can tailor the number of rules generated by BARBE-C. In this regard, we modify the Algorithm 1 to incorporate parse tree. The modified Algorithm 2 ensures that while randomly dropping words from a sentence if the word belongs to a leaf sibling set, all the words of the leaf sibling set also dropped from the sentence. This means that if a random word e.g. "movie" is selected to drop, the whole sibling set "a fantastic movie" is dropped from the sentence.

We regenerate the explanation using BARBE-C for the sentence "This is a fantastic movie of three prisoners who become famous. George Clooney is awesome. Another good thing about the movie is the soundtrack. I recommend this movie to everybody." using parse-tree based random word removal. The explanation generated by BARBE-

Algorithm 2 BARBE Neighborhood Generation Algorithm Using Parse Tree

```
Input: $inputSentence, n$
function GenerateLeafSiblignsUsingParseTree($inputSentence$)
    $leafSiblingsSet$ = Sets of all the leaf nodes that are siblings
    return $leafSiblingsSet$
end function
$leafSiblingsSet = GenerateLeafSiblignsUsingParseTree(inputSentence)$
for $i \leftarrow 1$ to $n$ do
    $sentence \leftarrow inputSentence$
    $nbWordsToDelete \leftarrow random[0, numberOfWords(inputSentence) - 1]$
    for $j \leftarrow 1$ to $nbWordsToDelete$ do
        $t$ = Select a word randomly and remove it from $inputSentence$. Drop all the words if it belongs to $leafSiblingsSet$
        $outputSentence = outputSentence \cup t$
    end for
end for
return $outputSentence$
```

C with parse tree has been illustrated in Figure 5.26. The set of rules reported by BARBE-C has been displayed in Table 5.7. BARBE-C has generated 7 rules as shown in Table 5.7. If we compare the rules in this Table with the rules in Table 5.3, we notice that using parse tree has deducted the rule: $fantastic \rightarrow Positive$ and assigns a higher confidence value of 100% to the rule $fantasticmovie \rightarrow Positive$. This happens because the word "fantastic" belongs to the leaf sibling set "a fantastic movie". The synthetic dataset does not contain the word "fantastic" alone inside the perturbed instances. If "fantastic" is present, it is present with all of its siblings, otherwise if "fantastic" is dropped, it is dropped with its siblings. Such methodology helps BARBE-C to generate rules that convey more semantic context to the users. Since the parse tree approach has reduced one rule, the feature importance histogram in Figure 5.26C differs from that in Figure 5.9C. The word "fantastic" has got a lower importance score as shown in Figure 5.26C since the rule $fantastic \rightarrow Positive$ has been removed by BARBE-C with parse tree.



A. Sentence:

This is a fantastic movie of three prisoners who become famous. George Clooney is awesome. Another good thing about the movie is the soundtrack. I recommend this movie to everybody.

Figure 5.26: The explanation provided by BARBE-C with parse tree for an instance of the IMDB movie review dataset labeled as positive by the black-box model. (A) shows the sentence with features highlighted. The green heatmap presents the positive words and the red heatmap presents the negative ones. Table 5.7 presents the set of important rules with their support, confidence, and logarithm of statistical significance values. (B) presents the prediction probability of the black-box, and (C) presents the histogram of important features ranked based on their importance.

Table 5.7: Set of rules generated by BARBE-C with parse tree with their support, confidence, and logarithm of statistical significance values along with the sentence with features highlighted. Rules are sorted in this table based on their confidence values. "P" denotes "Positive" label and "N" denotes "Negative" label in the Rule column. Sentence has been truncated and includes an ellipsis to indicate that there is more to the sentence.

Sentence	**Rule**	**Support**	**Confidence**	**Log(SS)**
This is ... Another good thing about the movie is the soundtrack. I recommend ...	*good* $\rightarrow$ *P*	21.07	100.00%	-57.01
This is ... George Clooney is awesome. Another good ...	*awesome* $\rightarrow$ *P*	21.98	100.00%	-57.54
This is a fantastic movie of three prisoners who become famous ...	*fantastic movie* $\rightarrow$ *P*	20.94	100.00%	-112.05
This is ... I recommend this movie to everybody.	*recommend* $\rightarrow$ *P*	22.54	99.75%	-58.29
This is a fantastic movie of three prisoners who become famous. George Clooney ...	*famous* $\rightarrow$ *P*	12.09	98.54%	-58.71
This is ... I recommend this movie to everybody.	*recommend movie* $\rightarrow$ *P*	15.26	98.41%	-55.23
This is a fantastic movie of three prisoners who become famous ...	*become* $\rightarrow$ *N*	17.07	62.08%	-59.87

Chapter 6

Using BARBE for Cyberbullying Detection in Low-Resource Languages

This chapter presents the application of BARBE for detecting cyberbullying in resource-constraint language. We first discuss what cyberbullying is, and the context of cyberbullying in terms of resource-constraint language like Bengali. Then we introduce a multi-class cyberbullying classification model for the Bengali language that achieves state-of-the-art performance. Furthermore, we demonstrate that the linguistic differences and complexities between high-resource languages like English and resource-constrained languages like Bengali lead to performance and accuracy differences. In this regard, we develop a novel embedding model called informal embedding for Bengali to improve cyberbullying detection accuracy. Finally, we apply BARBE to extract rules composed of a feature or conjunction of features to create more human-readable explanations.

Cyberbullying, also known as online harassment, is using electronic communication to threaten another person, usually by sending scary or compromising communications. The use of inflammatory and hostile language has expanded dramatically in the age of social media and internet networking. Examples of cyberbullying are derogatory emails, texts, photos, and videos sent over various social media platforms. Cyberbullying affects a large number of children all around the world. Over 80%

of children own a mobile phone and use social networking sites [110]; their online presence also exposes them to threats and social misbehaviors such as cyberbullying. Nearly 57% admitted to having experienced cyberbullying, and 60% reported witnessing bullying on social media [111]. 36.5% of adults believe they have been cyberbullied at least once in their lives, a figure that has more than doubled in recent years [111]. According to the Bangladesh Institute of ICT, Bangladesh has an 80% rate of cyberbullying victims who use the internet and social media, among them 64% of girls receiving sexually explicit videos, texts, and images [112]. Harassment and defamation account for 18% of the harassment complaints and cases brought before the country's only cybercrime tribunal [113].

To detect cyberbullying, the majority of recent studies have relied on traditional ML models [114], [115], [116], [117]. Some recent works use Deep Neural Network (DNN) models to detect cyberbullying. Agrawal et al. [118] employ DNN models to detect cyberbullying and claim their models outperform typical ML models. Multilingual cyberbullying detection has drawn considerable attention in recent years, with numerous studies published in major languages such as Dutch, Indian, Chinese, and Arabic [119], [120], [121]. Researchers have utilized lexical resources, modeling word and sentence relationships, and various word embedding techniques for feature extractions. Word embedding, in particular, has emerged as a commonly used method for NLP tasks, representing words in a vector space that encodes their meaning. However, the task of detecting cyberbullying in low-resource languages, such as Bengali, poses a significant challenge. With over 160 distinct inflected forms for verbs, 36 various forms for nouns, and 24 other forms for pronouns [122], Cyberbullying Detection in Bengali (CDB) requires sophisticated analysis methods. Although 230 million people worldwide speak in Bengali [123], there have been a few studies on CDB, and there are scopes for improvement in cyberbullying detection for such a low-resource language context. Most recent work on cyberbullying detection focuses on pre-trained embedding and deep neural network models for high-resource language.

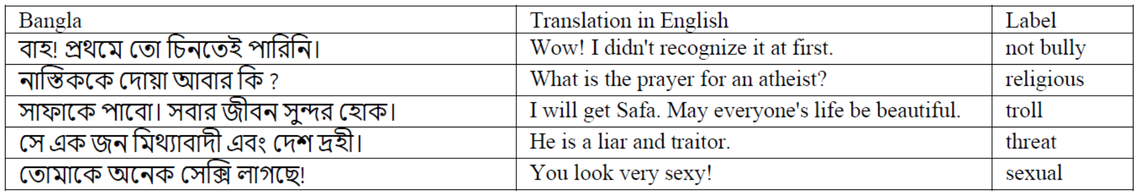
Bengali differs from high-resource languages as a resource-constrained language due to limited resources and complicated linguistic patterns [123]. Moreover, most of these pre-train embeddings do not work well in Bengali, and existing research on CDB has only considered binary classes, ignoring data imbalances. Thus, there is a pressing need to develop more effective techniques for detecting cyberbullying in low-resource language contexts such as Bengali.

Our key contributions in this chapter are:

- Developing a multi-class cyberbullying classification model for the Bengali language that achieves state-of-the-art performance. To improve cyberbullying detection accuracy, we develop a novel embedding model called informal embedding for Bengali.
- Utilizing BARBE for generating explanations in the form of rules consisting of a single feature or conjunction of features for better human understanding.

6.1 Dataset

We use the standard Bengali text dataset prepared by Ahmed et al. [124] by collecting hand-annotated comments on public Facebook posts from celebrities, government leaders, and athletes. The total number of comments is 44001. After removing duplicates, the comments are divided into five categories, with four subcategories in the bully category. A representative sample of the dataset is shown in Figure 6.1. Bullies are classified as sexual, threat, troll, and religious. If the comments are not fit into any of the criteria, they are labeled as not-bully. According to the dataset, the actors' class is the most harassed on social media, followed by social influencers with 61.25% and 21.31%, respectively [124]. Furthermore, we can see that 31.9% of the comments are directed at male victims, while 68.1% are directed toward female victims. Moreover, the dataset is not properly balanced among all classes.



Bangla	Translation in English	Label
বাহ! প্রথমে তো চিনতেই পারিনি।	Wow! I didn't recognize it at first.	not bully
নাস্তিককে দোয়া আবার কি ?	What is the prayer for an atheist?	religious
সাফাকে পাবো। সবার জীবন সুন্দর হোক।	I will get Safa. May everyone's life be beautiful.	troll
সে এক জন মিথ্যাবাদী এবং দেশ দ্রহী।	He is a liar and traitor.	threat
তোমাকে অনেক সেক্সি লাগছে!	You look very sexy!	sexual

Figure 6.1: Example dataset with English translation: one instance from each class

6.2 Methodology

This section explains how we have processed our dataset by separating bully from non-bully texts within the dataset, feature extraction approaches, selections of ML models, and our final methods. To get the best-performing architecture, we conduct an extensive analysis between various combinations of embedding techniques and deep ML model architectures. We describe some state-of-the-art XAI techniques that are used to explain the prediction of our ML models.

6.2.1 Dataset Preprocessing

Data pre-processing is a crucial step for preparing the data that can be analyzed conveniently and utilized for classification tasks. After gathering the comments, we clean up the raw data by removing inappropriate characters, punctuation, and emojis and pre-process the data before feeding it to our neural network.

6.2.2 Feature Extraction

The effectiveness of a model in a text classification task is heavily dependent on the feature extraction process, particularly for low-resource language text. In all text-processing tasks, words serve as discrete, categorical features. To address the issue of words that occur frequently but carry little significance, we utilized the TFIDF vectorizer [125] for our traditional models. The TFIDF value increases with the frequency of a word appearing in a single document and decreases proportionally to the number of documents in the corpus containing the word, resolving the aforementioned issue with the bag-of-words approach. We employed word n-grams within a range of

(1,5) and $min_{df} = 8$ with l2 normalization. Despite removing stop-words, we observe no significant difference in performance when utilizing the TFIDF vectorizer.

6.2.3 Machine Learning Classifiers

Our study involves experimenting with four supervised ML classifiers: Multinomial Naive Bayes [126], SVM [127], Random Forest [128], and XGBoost [129] because of their reputation in the classification task.

6.2.4 Deep Neural Network Models

Neural networks demonstrate outstanding performance in a broad range of text classification and generative tasks. Given a large amount of training data, such models typically achieve higher accuracy than linguistic feature-based methods and other traditional ML models. Therefore, we compare two simple and representative models, CNN [130] and LSTM [29], and experiment with these models that have been used as benchmark models in various text classification tasks.

6.2.5 Word Embedding Models

The embedding layer works with a set of words having a specific length. Word embeddings are a type of real-valued vector that represents each word in the vector space. We explore Word2Vec [131], fastText [132], and fastText built on informal texts as initialization approaches for word embeddings. Using initial word embeddings during training can assist the model in acquiring task-specific word embeddings that can aid in detecting specific types of bullying in multiclass prediction.

6.2.6 Explainability

We choose BiLSTM as the black-box model to generate explanations for the cyberbullying dataset. We apply Anchor and BARBE to the learned model for producing explanations that are easy to understand for readers. Section 6.3.4 and 6.3.5 contain more details on the explanation generated by Anchor and BARBE respectively.

6.2.7 Evaluation Metrics

As the evaluation metrics, we use the accuracy(A), precision(P), recall(R), and F1-Score(F1) of our models. As the dataset is imbalanced, F1-Score is a better metric for evaluating the results. Furthermore, as our dataset is imbalanced during the trial, we use macro average accuracy as a performance parameter.

6.2.8 Proposed Workflow

There are two aspects to our proposed model. We begin by experimenting with four traditional models for multiclass scenarios based on the descriptors: "Not bully", "sexual", "threat", "troll", and "religious". In the second part, while experimenting with deep learning models, we separate the second section again into two parts. In the first part, we consider these labels - "sexual", "threat", "troll", "religious" as "bully" and another part as "Not bully". Then we apply binary classification models using BiLSTM and CNN to identify whether the comment is harassing or not. We redesign the same models for multiclass prediction in the later part of this study. We have presented our workflow diagram in Figure 6.2. In the following subsections, we review each step of our approach.

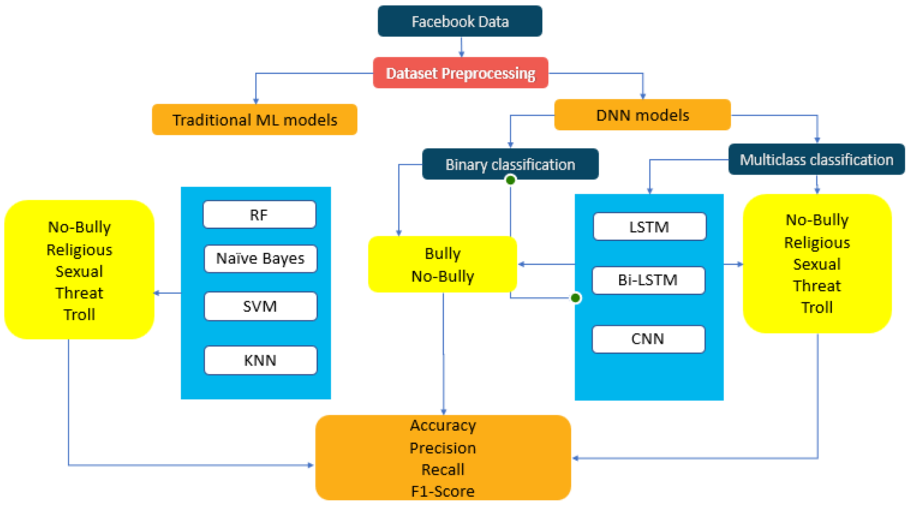

Facebook Data
Dataset Preprocessing
Traditional ML models
DNN models
Binary classification
Multiclass classification
No-Bully Religious Sexual Threat Troll
RF
Naïve Bayes
SVM
KNN
Bully No-Bully
LSTM
Bi-LSTM
CNN
No-Bully Religious Sexual Threat Troll
Accuracy Precision Recall F1-Score

Figure 6.2: Overview of complete workflow

6.3 Result and Analysis

In order to obtain the best-performing models, we evaluate the performance of several machine learning models on an identical dataset for both English and Bengali languages. We compare the performance of four traditional ML models (SVM, Random Forest, NB, and XGBoost) in terms of Precision (P), Recall (R), and F1-Score (F1) to compare the performance with Ahmed et al. [124]. Later, we use the same dataset to test multiple deep learning models (LSTM, BiLSTM, and CNN) with different word embeddings. We build our own embedding with informal text and comments collected from various online social platforms representing real-world data. We use the term informal embedding or ***fastText_informal*** throughout the paper, describing the embedding model we have developed.

6.3.1 Experiment on English and Bengali Datasets for Traditional ML Models

We conduct a comprehensive experiment on both the English and Bengali datasets in this section. We choose the Twitter dataset [1] as the English language cyberbullying dataset of 47,691 comments with five classes ("not-cyberbullying", "gender", "age", "religion", and "ethnicity") which is identical to our Bengali dataset. We divide our dataset into 80% for training and 20% for testing. We report our experimental results on both English and Bengali for binary classification tasks in Table 6.1. It is evident from the Table that SVM outperforms all other conventional models to predict cyberbullying in the binary class scenarios for both datasets.

For multiclass classification, our study finds that SVM outperforms all traditional models. To obtain the optimal performance, we have used GridSearchCV [2] tools from Scikit-learn for fine-tuning SVM with various parameters. We have performed cross-validation using sklearn StratifiedKFold to find the optimal setting and experimented

[1]https://rb.gy/wtumhy

[2]https://scikit-learn.org/stable/modules/generated/sklearn.model$_{s}election.GridSearchCV.html$

Table 6.1: Cyberbullying Classification Results: Precision (P), Recall (R), and F1-Score (F1) generated by traditional machine learning models for Bengali and English datasets.

Algorithm	Dataset	P	R	F1
Random Forest	Bengali	84.00	84.00	84.00
	English	82.40	84.10	81.60
SVM	Bengali	**85.00**	**85.00**	**84.00**
	English	**89.00**	**90.40**	**89.60**
XGBoost	Bengali	85.00	84.00	84.00
	English	81.80	84.60	82.40
Naive Bayes	Bengali	78.00	79.00	79.00
	English	80.80	79.00	79.80

with different kernels to find the optimal SVM output. We identify that SVM classifier with Radial Based Kernel achieves the overall best performance among other kernels. Both Bengali and English datasets have the same five classes: Not bully, Religious, Sexual, Threat, Troll and Not bully, Religion, Gender, Ethnicity, Age respectively. Note that, the classes of the Bengali and English datasets are not exactly the same but essentially represent the name as the same for better presentation. Here, Not bully (Class-N), religious or religion (Class-R), Sexual or gender (Class-G), Threat or ethnicity (Class E), Troll or Age (Class-A) encompass all five classes from both datasets. Table 6.2 reports the performance of SVM models on 5 classes for both Bengali and English datasets.

From Table 6.2, we observe that Class-R which is the religious or religion category gets the highest F1-Score for both Bengali and English datasets. On the other hand, Class-E achieves the lowest for the Bengali dataset and Class-N achieves the lowest for the English dataset. The difference found in Class-E and Class-N is due to the high imbalance in both datasets. As the dataset is collected from social media, more work can be done to make it a gold standard dataset.

Table 6.2: Precision, Recall, and F1-Score generated by SVM (Radial Basis Kernel) for Bengali and English datasets. (Precision, Recall, and F1-Score are represented as P, R, and F1 respectively)

Class	Dataset	P	R	F1
Class-N	Bengali	87.00	91.00	89.00
	English	63.00	53.00	58.00
Class-R	Bengali	97.00	95.00	96.00
	English	91.00	85.00	88.00
Class-G	Bengali	93.00	85.00	89.00
	English	82.40	84.10	81.60
Class-E	Bengali	90.00	50.00	65.00
	English	99.00	98.00	98.00
Class-A	Bengali	77.00	84.00	80.00
	English	95.00	98.00	97.00

6.3.2 Experiment with Word-Embeddings and DNN Models

In this section, we examine three deep neural network models (LSTM, BiLSTM, and CNN) with different word embedding for feature extractions to detect cyberbullying. We analyze the performance of CNN, LSTM, and BiLSTM models on both English and Bengali datasets in terms of Precision, Recall, and F1-Score using two different embeddings Word2Vec, and fastText respectively. It is evident from Table 6.3 that BiLSTM and fastText embeddings work best for both datasets in terms of Precision, Recall, and F1-Score. The average F1-Score for fastText embedding and BiLSTM is 95.20% and 83.45% respectively for English and Bengali, which is much higher than the performance of any traditional model.

For multiclass classification, we examine all the deep learning models used for binary classification and get similar findings that BiLSTM with fastText embedding outperforms all other models. Table 6.4 presents the class-wise precision, recall, and F1-Score for the Bengali dataset.

Table 6.3: Precision, Recall, and F1-Score generated by CNN, LSTM, and BiLSTM with word2Vec, and fastText embeddings respectively for binary classification. (Precision, Recall, and F1-Score are represented as P, R, and F1 respectively)

Algorithm	Dataset	P	R	F1
CNN+word2Vec	Bengali	76.39	73.28	74.78
	English	93.34	91.55	92.20
CNN+fastText	Bengali	80.93	77.45	79.16
	English	94.51	92.65	93.81
LSTM+word2Vec	Bengali	85.99	74.16	79.64
	English	93.29	92.51	92.09
LSTM+fastText	Bengali	86.42	75.42	80.55
	English	94.64	93.87	94.00
BiLSTM+word2Vec	Bengali	90.05	75.16	82.16
	English	95.20	92.50	94.33
BiLSTM+fastText	Bengali	90.10	77.20	83.45
	English	96.55	93.31	95.20

6.3.3 Impact of Embedding Models

In this section, we examine word2Vec and fastText embedding with CNN, LSTM, and BiLSTM. Experimental results from Table 6.3 clearly demonstrate that fastText embedding works better than the word2Vec. In addition, it shows that BiLSTM and fastText achieve **95.2%** F1-Score for the English dataset while the performance is **83.45%** for the Bengali dataset. We investigate the reason behind the huge performance drop for the same deep learning models and embedding. We hypothesize that the embedding model for Bengali and English can be different due to their linguistic complexity and Bengali has more regional words with semantically the same meaning than English. To solve this problem, we build our own fastText informal embedding. Table 6.5 represents the performance of BiLSTM and CNN models for binary classification using Word2Vec, fastText, and fastText built on informal text (fastTextInf)

Table 6.4: Precision, Recall, and F1-Score generated by BiLSTM model with word2Vec, fastText, and fastText informal embeddings respectively for multiclass classification with Bengali dataset. (Precision, Recall, and F1-Score are represented as P, R, and F1 respectively)

Class	Embedding	P	R	F1
Not bully	Word2Vec	75.00	**82.00**	78.00
	fastText	77.00	**82.00**	79.00
Religious	Word2Vec	**91.00**	80.00	85.00
	fastText	**91.00**	85.00	88.00
Sexual	Word2Vec	**86.00**	57.00	69.00
	fastText	**72.00**	68.00	70.00
Threat	Word2Vec	**84.00**	64.00	72.00
	fastText	72.00	72.00	72.00
Troll	Word2Vec	**70.00**	59.00	64.00
	fastText	**71.00**	61.00	66.00

in terms of Precision, Recall, F1-Score, and Accuracy. We find that both the models using fastText on informal text perform the best. BiLSTM achieves Precision, and Recall of 89.54%, 84.79% respectively, as well as an overall F1-Score 87.99%, the highest of all the embeddings. CNN also shows a similar result in the case of informal text embedding. CNN achieves Precision and Recall of 86.89%, 86.66% respectively, as well as an overall F1-Score of 86.77%. Among the other two, fastText performs well in the case of the Bengali language for cyberbullying detection with nearly 80.55 % of F1-Score. Results show that informal embedding positively influences the performance of both CNN and LSTM architecture by more than **7%**. The best results are highlighted in bold font.

Table 6.6 represents the performance of BiLSTM and CNN models for the multiclass scenario for the three embedding models in terms of Precision, Recall, and F1-Score. The best results are highlighted in bold font. For multiclass classifica-

Table 6.5: Precision, Recall, and F1-Score generated by BiLSTM and CNN models with Word2Vec, fastText, and fastTextInf embeddings for binary classification

Algorithms	Embeddings	P	R	F1
BiLSTM	Word2Vec	85.99	74.16	79.64
	fastText	86.42	75.42	80.55
	fastTextInf	**89.54**	**84.79**	**87.99**
CNN	Word2Vec	76.39	73.28	74.78
	fastText	80.93	77.45	79.16
	fastTextInf	**86.89**	**86.66**	**86.77**

tion, both the models using fastText on informal text (fastTextInf) achieve the best performance, similar to the result for binary class classification. BiLSTM achieves Precision and Recall of 84.00% and 83.00% respectively, as well as an overall F1-Score of 84.00%, the highest of all the embeddings. CNN also shows a similar result in the case of informal text embedding. CNN achieves Precision and Recall of 79.00% and 78.00% respectively, as well as an overall F1-Score of 79.00%. Among the other two, for BiLSTM and CNN, the fastText model achieves an F1-Score of 79.00% and 71.00% respectively in CDB texts.

Table 6.6: Precision, Recall, and F1-Score generated by BiLSTM and CNN models with Word2Vec, fastText, and fastTextInf embeddings for multiclass classification

Algorithms	Embeddings	P	R	F1
BiLSTM	Word2Vec	78.00	70.00	74.00
	fastText	77.00	74.00	75.00
	fastTextInf	**84.00**	**83.00**	**84.00**
CNN	Word2Vec	76.00	67.00	69.00
	fastText	72.00	69.00	71.00
	fastTextInf	**79.00**	**78.00**	**79.00**

Table 6.7 illustrates the classification results for BiLSTM model based on fastText

embedding built on informal text for multiclass classification. Based on precision, recall, and F1-Score from Table 6.6, we find that the BiLSTM model performs better with fastText embedding than other embeddings in properly identifying all the classes while utilizing informal texts.

Table 6.7: Precision, Recall, and F1-Score generated by BiLSTM with informal text embedding for Bengali dataset

Class	P	R	F1
Not bully	86.00	86.00	86.00
Religious	92.00	91.00	92.00
Sexual	83.00	82.00	83.00
Threat	77.00	76.00	76.00
Troll	79.00	75.00	77.00

6.3.4 Explainability with Formal and Informal Embedding

We have conducted a few experiments with LIME and Anchor to explain the results obtained by the black-box models. We present the explanation for the BiLSTM classifier using Anchor not LIME since our findings show that the explanation generated by LIME for resource-constraint language is not clear and meaningful. Figure 6.3 shows an example of the explanation generated by LIME for an instance of the Bengali cyberbullying dataset. LIME is not able to identify the features responsible for the decision of the black-box model.

We provide two examples of Bengali sentences as shown in Figure 6.4 to explain with Anchor. In this example, the first sentence is truly labeled as “not bully” and the second sentence is labeled as “bully”. The best-performing model BiLSTM with formal embedding predicts both of them incorrectly. However, BiLSTM with informal embedding successfully classifies it. The explanation of the two sentences of Figure 6.4 has been provided in Figure 6.5 and Figure 6.6 respectively. The first illustration in Figure 6.5(a) belongs to the class “not bully”. Its English translation is “Sudipta

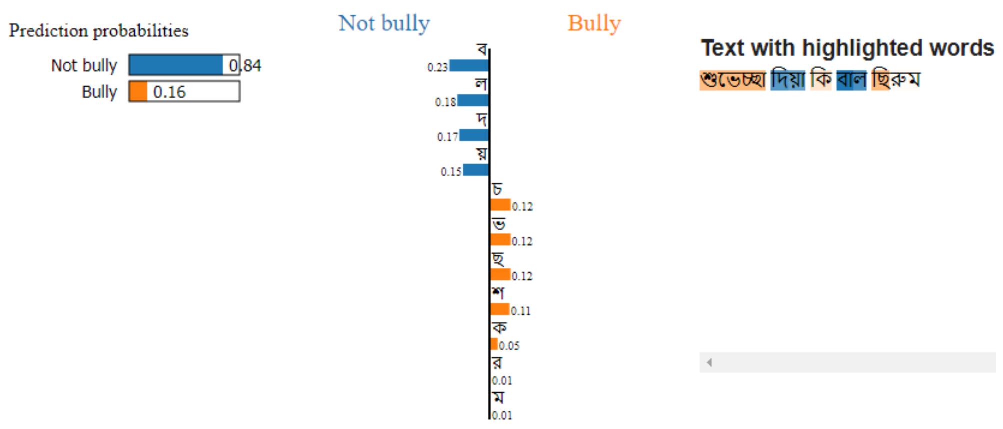


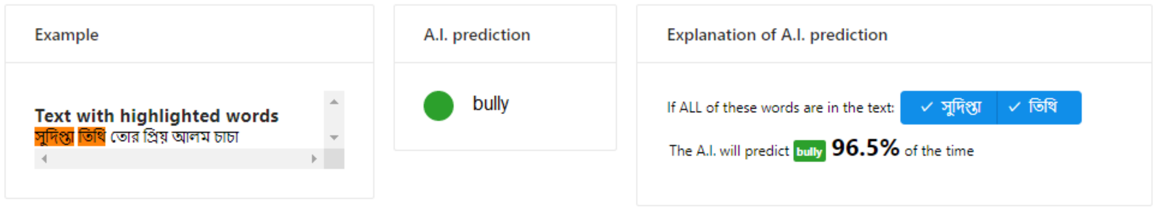
Figure 6.3: Explanation generated by LIME for an instance of the cyberbullying dataset. The characters within the words are broken.

Example Sentence	English Translation	Model Name	Model Output	Actual Output
সুদিপ্তা তিথি তোর প্রিয় আলম চাচা	Sudipta Tithi is your dear Alam Uncle	Formal - BiLSTM	bully	not bully
		Informal - BiLSTM	not bully	
তুই মাগি কিবলতেচাস তুইত জারজ	You prostitute what you want to say you bastard	Formal - BiLSTM	not bully	bully
		Informal - BiLSTM	bully	

Figure 6.4: Example sentences with the prediction from the BiLSTM model for both formal and informal embedding.

Tithi is your dear Alam Uncle". The words "Sudipta" and "Tithi" are two Bengali names. The word "Alam" is a common middle name of Bengali people. In general, this sentence is not a bully because there is no indicative word in this sentence that makes it "bully". When we use the BiLSTM classifier as the desired black box model and use Anchor to explain it with both formal embedding and informal embedding setup, the explanation for the formal embedding is "bully" which is wrongly predicted. This happens because the words that made the prediction "bully" are rarely available in formal literature. For the illustration in Figure 6.5(b), the output of the informal embedding is "not bully" which matches the original label of the sentence. The explanation for the second sentence of Figure 6.4 is presented in Figure 6.6. Its English translation is "You prostitute what to say, you bastard". It is clearly evident that there are a couple of slang words in this sentence which are slang words in Bengali conversation. Our experiment with BiLSTM informal embedding correctly predicts the label of the sentence as shown in Figure 6.6(b). The prediction of the

BiLSTM with informal embedding is "bully" but BiLSTM with formal embedding fails to predict it. Instead, it predicts the sentence as "not bully" which is not correct. Figure 6.6(a) also highlights the words that turn the sentence into "bully".



Example

Text with highlighted words
সুদিপ্তা তিথি তোর প্রিয় আলম চাচা

A.I. prediction

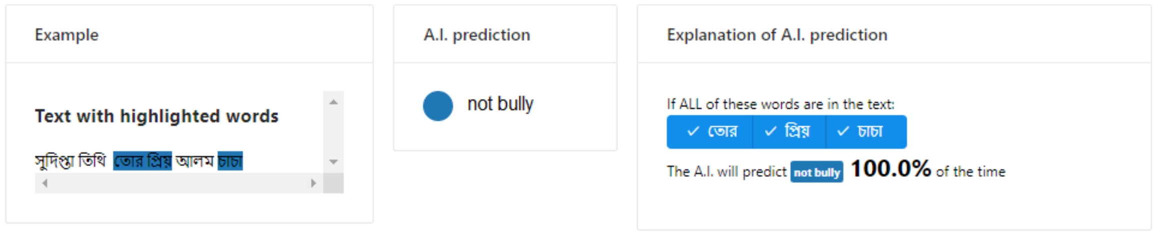
● bully

Explanation of A.I. prediction

If ALL of these words are in the text: ✓ সুদিপ্তা ✓ তিথি

The A.I. will predict bully **96.5%** of the time

(a) BiLSTM using Formal Embedding



Example

Text with highlighted words

সুদিপ্তা তিথি তোর প্রিয় আলম চাচা

A.I. prediction

● not bully

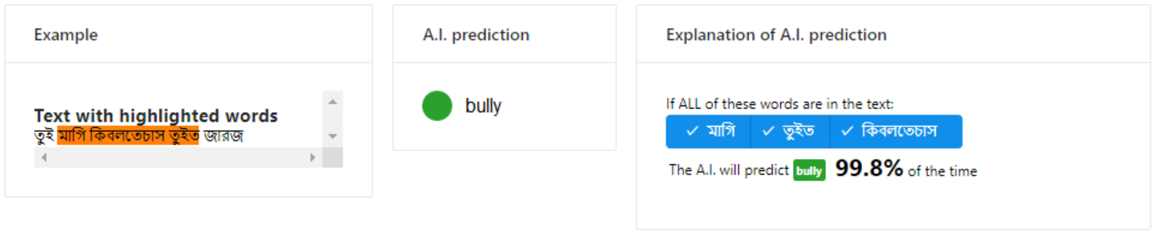
Explanation of A.I. prediction

If ALL of these words are in the text:
✓ তোর ✓ প্রিয় ✓ চাচা

The A.I. will predict not bully **100.0%** of the time

(b) BiLSTM using Informal Embedding

Figure 6.5: The explanation generated by Anchor using BiLSTM classifier as the black-box model with formal and informal embedding. The original label is "not bully" and the informal embedding predicts "not bully"



Example

Text with highlighted words
তুই মাগি কিবলতেচাস তুইত জারজ

A.I. prediction

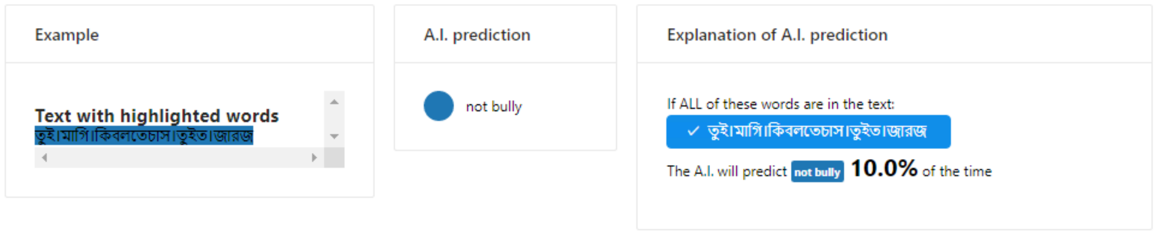
● bully

Explanation of A.I. prediction

If ALL of these words are in the text:
✓ মাগি ✓ তুইত ✓ কিবলতেচাস

The A.I. will predict bully **99.8%** of the time

(a) BiLSTM using Informal Embedding



Example

Text with highlighted words
তুই।মাগি।কিবলতেচাস।তুইত।জারজ

A.I. prediction

● not bully

Explanation of A.I. prediction

If ALL of these words are in the text:
✓ তুই।মাগি।কিবলতেচাস।তুইত।জারজ

The A.I. will predict not bully **10.0%** of the time

(b) BiLSTM using Formal Embedding

Figure 6.6: The explanation generated by Anchor using BiLSTM classifier as the black-box model with formal and informal embedding. The original label is "bully" and the informal embedding predicts "bully"

6.3.5 Model Interpretation Using BARBE

In order to generate a more meaningful and better readable explanation, we use BARBE to generate the set of rules within informal embedding. Figure 6.7 and 6.8 demonstrate the explanations generated by BARBE. For convenience, we also provide the English translation of the Bengali sentences used in these figures. BARBE highlights the indicative words that are responsible for classifying the sentence as either "bully" or "not bully". According to Figure 6.7, the original label of the sentence is "not bully". BARBE extracts the rules that are making the sentence "not bully" as presented in 6.7(b). Figure 6.7(b) shows support and confidence values of the rules generated by BARBE. The horizontal bar chart in Figure 6.7(c) represents prediction probabilities of the black-box model whereas the vertical bar chart in Figure 6.7(d) shows important features based on the word frequency within the rules. Figure 6.8 demonstrates another example used in BARBE where the original label of the sentence is "bully" and BARBE significantly extracts the rules that make the sentence classified as "bully". Unlike Anchor, BARBE generates a set of rules along with the frequency and confidence values which clearly explain the decision of the black-box model by improving the understandability and clarity of the prediction. The vertical bar chart in Figure 6.8(d) also signifies the critical words in the sentence with a heatmap visualization of the sentence presented in Figure 6.8(a).

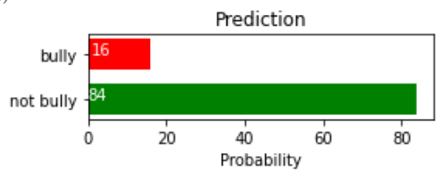
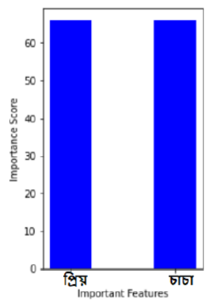
Sentence highlighted with indicative words	
a) সুদিপ্তা তিথি তোর প্রিয় আলম চাচা	(English: Sudipta Tithi is your dear Alam Uncle)
Rules with Support and Confidence Values	
b) সুদিপ্তা তিথি তোর প্রিয় আলম চাচা	প্রিয় -> not bully (93, 100%)
সুদিপ্তা তিথি তোর প্রিয় আলম চাচা	চাচা -> not bully (89, 98%)
সুদিপ্তা তিথি তোর প্রিয় আলম চাচা	প্রিয় চাচা -> not bully (90, 95%)
c) 	d) 

Figure 6.7: The explanation provided by BARBE for an instance of the cyberbullying dataset where the actual label is “not bully”. (a) shows the heatmap of highlighted words. (b) shows the rules extracted by BARBE with support and confidence values. (c) shows the black-box probabilities. (d) presents the important features with their importance score.

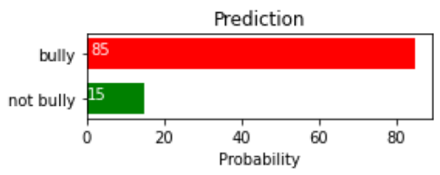
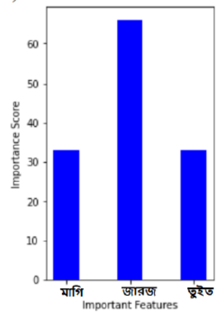
Sentence highlighted with indicative words	
a) তুই মাগি কিবলতেচাস তুইত জারজ	(English: You prostitute what you want to say you bastard)
Rules with Support and Confidence Values	
b) তুই **মাগি** কিবলতেচাস তুইত জারজ	মাগি -> bully (92, 100%)
তুই মাগি কিবলতেচাস তুইত **জারজ**	জারজ -> bully (94, 100%)
তুই মাগি কিবলতেচাস **তুইত জারজ**	তুইত জারজ -> bully (87, 98%)
c) 	d) 

Figure 6.8: The explanation provided by BARBE for an instance of the cyberbullying dataset where the actual label is “bully”. (a) shows the heatmap of highlighted words. (b) shows the rules extracted by BARBE with support and confidence values. (c) shows the black-box probabilities. (d) presents the important features with their importance score.

Chapter 7

Systematic Comparison of the Explanations

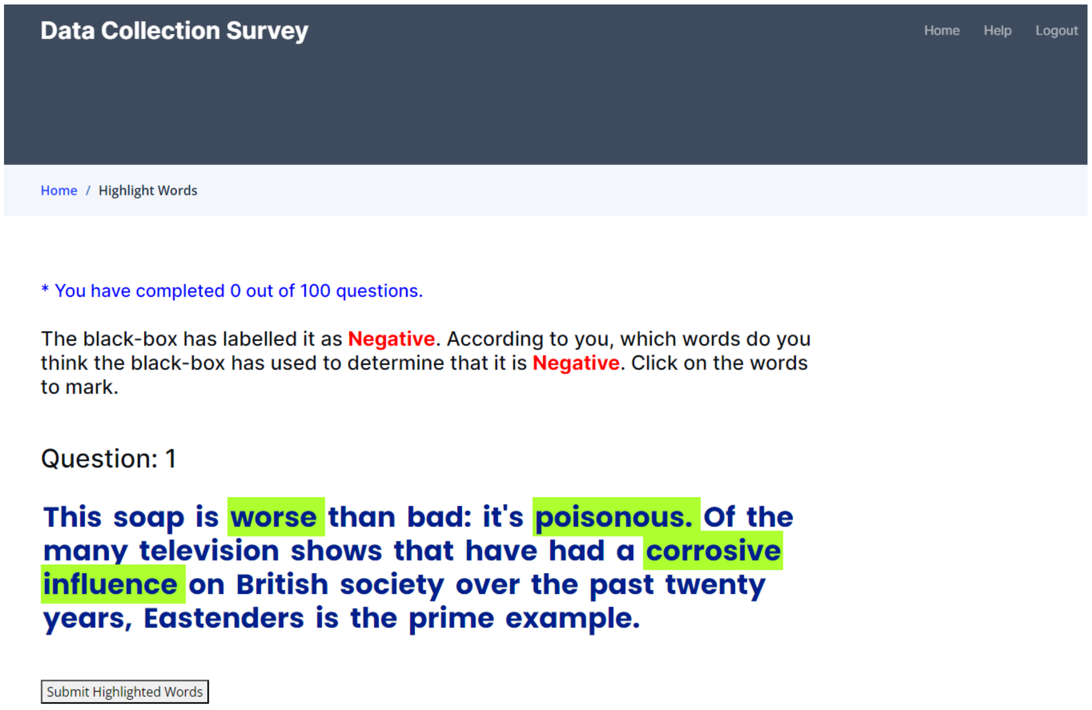
To systematically compare the explanations provided by BARBE and other systems like LIME, we conducted a human evaluation in which human participants created a "ground truth" that we use to estimate the explanation performance on sentiment analysis task explanations. In this evaluation, participants perform the task of data annotation where they mark the features that they think influence the black-box to come to a decision. We use the same black-box that we use for BARBE-C (BARBE with CFAR). We use a set of 200 movie reviews labeled by the black-box as either positive or negative. Participants are asked to annotate the words or a set of words i.e. conjunction of words that they think led the black-box to its decision. This evaluation helps us to construct the ground truth. Participants are presented with a series of movie reviews and the corresponding sentiment predicted by the black-box model. For example, consider the following movie review:

"This movie is a perfect blend of action, drama, and romance. The actors deliver outstanding performances, making the story both thrilling and heart-wrenching at the same time."

that has been predicted as "Positive" by the black-box model. The participants might consider the words like "perfect", "perfect blend", "outstanding performance", "heart-wrenching" as the words for which the black-box could have labeled this review as “Positive”. We employ BARBE-C to extract the rules for this review and sort the rules by their frequency, and confidence values, and finally extract the words identified by BARBE-C. We also use LIME to extract the words for each of the reviews used in the study. Eventually, we come up with a metric to measure the participant’s observation with BARBE and LIME that we discuss in this chapter. We start with the data annotation process in Section 7.1. Then we discuss the agreement between annotators in Section 7.2. Finally, we compare the explanations provided by BARBE and LIME by proposing a metric in Section 7.3. We do not compare against Anchor and SHAP here as they produce a limited set of words as explanations and would de-facto perform poorly with our metric.

7.1 Data Annotation

In order to conduct this evaluation, we randomly choose 200 IMDB movie reviews. We invited 6 human annotators for our study. They participated in this study voluntarily. We divided the users into two groups. Three users from Group 1 annotated the first 100 reviews and the other three users from Group 2 annotated the next 100 reviews. We developed a web application to collect the annotations through an interactive interface. When users first sign up for the evaluation, they are assigned a unique username and password which they use to log in to the web application. Upon successful login, they are immediately presented with a sentence. A sentence is basically an IMDB movie review. This sentence is labeled either positive or negative by the black-box model. As shown in Figure 7.1, the user interface presents this question along with the decision of the black-box (i.e., whether the review is deemed positive or negative). The user clicks on a word (or multiple words) to mark it as a potential influence on the black-box decision. Once clicked on a word, it will turn green meaning that the user has selected this word to be influential for black-box decisions. However, it is essential to note that users will not judge their decision whether a review is positive or negative. Instead, the user has to annotate the words for which the black-box has predicted that review as positive or negative. By identifying these influential words, the annotators help us to develop the ground truth for our evaluation. The annotators are independent, they don't see each other annotations. Each sentence is annotated by 3 annotators.



* You have completed 0 out of 100 questions.

The black-box has labelled it as Negative. According to you, which words do you think the black-box has used to determine that it is Negative. Click on the words to mark.

Question: 1

This soap is worse than bad: it's poisonous. Of the many television shows that have had a corrosive influence on British society over the past twenty years, Eastenders is the prime example.

Figure 7.1: User interface of our evaluation

Once users mark i.e. annotate the words for a review, they click the "Submit Highlighted Words" button to move to the next review as shown in Figure 7.1. After they annotate 100 reviews, they are done with the evaluation. We save the annotations in our database. When all six users are done with their tasks, we analyze their agreement to construct the ground truth.

7.2 Agreement between Annotators

We analyze the annotations provided by the annotators to calculate the agreement between annotators. Table 7.1 demonstrates the summary of the evaluation. There are a total of six annotators, divided into two groups. They annotated a total of

188 questions. Each sentence was annotated by 3 annotators. We noticed that 12 reviews had no annotations at all. This could happen because the annotators might not find the words that they think are relevant to annotate. As a result, we record annotations for 188 reviews. A full agreement is when the three annotators select the same words for a sentence and majority agreement is when two out of three annotators select a word as influential in a sentence for the black box decision. For example, for this sentence: *"This movie is a perfect blend of action, drama, and romance. The actors deliver outstanding performances, making the story both thrilling and heart-wrenching at the same time."*, Annotator 1 marks "stunning, visual, auditory, feast, awe", Annotator 2 marks "stunning, visual, and, auditory, feast, awe", and Annotator 3 marks "stunning, awe". All users agree on "stunning, awe" and the majority of users agree on "stunning, visual, auditory, feast, awe". This is how we define full agreement and majority agreement in our evaluation. Among the 188 reviews as shown in Table 7.1, we see that for only 32 reviews, all the annotators agree. But the majority of the annotators agree on 149 reviews. We take this agreement as the ground truth for our evaluation. The last column in this table shows the average number of words in majority agreement – i.e. 2 out of 3 annotators agree on 5 words on average. Our average is 5 in our study.

We calculate the kappa agreement [133] to demonstrate the level of agreement between the annotators. We obtain 0.635 for the kappa agreement between Annotator 1 and Annotator 2, 0.644 between Annotator 2 and Annotator 3, and 0.675 between Annotator 1 and Annotator 3. The final kappa agreement we obtain by taking the average is 0.651. This can be considered as a good kappa agreement between the annotators[1].

[1]https://elentra.healthsci.queensu.ca/assets/modules/reproducibility/kappa$_v$*alues.html*

Table 7.1: Summary of user annotations

Number of Annotators	**Total Questions**	**Responses Given**	**Full Agreement**	**Majority Agreement**	**Avg. Word in Majority Agreement**
6/2=3	200	188	32	149	5

7.3 Comparison of Explanations

We generate the explanations using BARBE-C and LIME for all 188 reviews. In order to compare the explanations provided by BARBE and LIME using the ground truth (majority agreement), we propose a metric to assign scores to BARBE and LIME. We call it WSMA: Word Selection Majority Agreement. This metric reflects the presence of words in the explanation, according to their agreement with the majority and takes into account the ordering of the words in the explanations since in the explanation, words are ranked. We extend the same metric with the presence of co-locations of words in the explanation if these words are in the ground truth. We call this new extension: WSMA+. We explain the algorithm of WSMA and WSMA+ in this section that we use to assign scores to both BARBE and LIME.

Both BARBE and LIME generate ranked lists of words. LIME ranks the words based on some probability values and BARBE ranks them using the support and confidence values obtained from the association rules. We construct a list of words based on majority agreement that we call our ground truth. This list serves as the benchmark against which the model outputs are compared. The ground truth is based on the "majority agreement". Each word in the model output list is compared to the ground truth. If the word exists in the ground truth, it's assigned a score, depending on its rank. The top-ranked word that also exists in the ground truth gets a score equal to the total number of words (n) in the ground truth. The second-ranked word gets $n-1$ points if it is in the ground truth, the third $n-2$, and so forth, until we check the first n ranked words, or the model's list ends. If a word from the model's list is not in the ground truth, it's assigned a score of zero. The "best case scenario" for a model is if all its words match those in the ground truth, in which case its total score would be the sum of the first n natural numbers, which can be calculated as $n \times (n+1)/2$. Since we obtain 5 as the average number of words in the majority agreement as presented in Table 7.1, the "best case scenario" when LIME matches the ground truth or BARBE matches the ground truth is 15 on average. WSMA+ also considers word co-locations. If a model output includes a co-location that also exists in the ground truth, it gets an additional point. This appears to be more advantageous for BARBE since the association rules generated by BARBE contain conjunction of words, but to prevent over-penalizing LIME, which does not, the additional point for co-locations is kept minimal. Such scoring methods quantify the models' effectiveness when compared to the ground truth.

WSMA and WSMA+ compare the output of BARBE and LIME using the majority-agreed ground truth from the evaluation results. After applying WSAM and WSMA+ to all 188 reviews, we observe that BARBE significantly outperforms LIME, demonstrating a higher degree of agreement with the human participants' perceptions. We obtain 9.05 for BARBE and 8.73 for LIME as shown in Table 7.2 when we use WSMA by averaging over 188 reviews. Using WSMA+ gives 10.29 which is an advantage for BARBE since it generates rules containing a conjunction of words. LIME remains the same when WSMA+ is applied since LIME only identifies one single word at a time. We obtain 5 as the average number of words in the majority agreement as presented in Table 7.1. Thus 15 is the best-case scenario on average for our evaluation. It is evident that BARBE with WSMA and WSMA+ scoring is close to this number than LIME. This demonstrates a clear win for BARBE. This result suggests that the explanations provided by BARBE are more useful, and more human-aligned compared to those provided by LIME. In other words, BARBE appears to better mimic human reasoning in these instances. The rules or features highlighted by BARBE as important for the black-box model's decision-making process are similar to those identified by the majority of the participants. This alignment is crucial in the field of XAI, as it indicates that BARBE's explanations are not only understandable to humans but also correspond with their reasoning. By analyzing the marked words across all the reviews, we were able to construct a set of association rules that reflect the human perception of the black-box model. Moreover, the evaluation aims to serve as a benchmark for evaluating the quality of explanations generated by our proposed method.

We have not chosen Anchor and SHAP to compare with BARBE in this evaluation. Our experimental analysis finds that Anchor mostly chooses one or two words from a sentence. It misses many of the significant influential words (e.g. Figure 5.19). Whereas BARBE selects all the influential words and LIME ranks all words. It is important to note that LIME can select all the words from the sentence since it has a parameter to select the words. As a result, LIME picks up non-influential words as well and obtains less score using our scoring method since the non-influential words are mostly not chosen by the participants. SHAP's calculation involves a combinatorial problem, meaning that it can get computationally expensive as the number of words grows in a sentence. SHAP values assume that features interact additively, which might not be the case in all black-box models where features could have interaction effects.

Table 7.2: Comparative evaluation of BARBE and LIME using WSMA and WSMA+

	BARBE	LIME
WSMA	9.05	8.73
WSMA+	10.29	8.73

Chapter 8

Conclusion

In recent years, there has been a significant interest in the field of eXplainable Artificial Intelligence (XAI). Several researchers have focused on developing various explanation frameworks to provide explanations for different deep neural network architectures. However, despite their reasonable performance in terms of accuracy, these methods are restricted to specific DNN architectures, limiting their applicability. LIME was first proposed by Ribeiro et al. [4] as a versatile framework capable of providing explanations for any black-box model. Despite their assertion that LIME is model-agnostic and can be applied to any model, the method actually relies on the black-box model to provide a probability score for each class. Furthermore, the effectiveness of LIME in providing explanations and generalizing it across different datasets remained an open research problem. We also evaluated Anchor [5] and SHAP [6] as the explainers to generate quality explanations. The drawback of Anchor is that it does not provide insights into the interactions between the most important features it captures or how their joint effects contribute to the prediction. It also does not express the associations among the features and does not expose the importance scores or weights of the features like LIME. On the other hand, SHAP can be computationally expensive, especially for large datasets or complex models, which can make it impractical for some use cases.

In Chapter 4, we introduced Black-box Association Rule-Based Explanations (BARBE) which was originally proposed by Shabestari [1] in his thesis. We extended BARBE to work with text in our thesis. BARBE is an explainable framework that is not dependent on any particular model and delivers explanations with a high degree of precision in the form of rules, along with a ranked list of important features. The rules generated by BARBE not only reveal the important features responsible for the decision of the black-box model but also provide insight into the associations among these features. BARBE, in contrast to LIME, is not dependent on the probability scores generated by some classifiers and has the ability to generate an explanation for any black-box classifier. We discussed two variants of BARBE in this manuscript. One is BARBE with SigDirect and another is BARBE with CFAR. BARBE with SigDirect uses SigDirect as its underlying associative classifier. Although SigDirect is effective for processing short sentences, it may encounter limitations in dealing with high-dimensional datasets that have a large feature vector space, as the number of possible association rules increases exponentially with the number of features. This can result in computationally expensive classifiers that require a significant amount of memory and run-time to process. To overcome this limitation, BARBE with CFAR (Classification by Frequent Association Rules) was proposed, which uses an ensemble approach that utilizes a set of base learners where each base learner is an associative classifier. This approach enables BARBE with CFAR to efficiently handle high-dimensional datasets with large feature vector spaces by distributing the feature space among base learners, each of which is trained on a subset of the feature vector space.

We explored different data augmentation techniques in Chapter 2. We particularly focused on the EDA to generate neighborhood datasets around the provided instance for BARBE. However, we could not incorporate the EDA methods that increase the vocabulary by adding new words within the sentence. Also, we were not able to utilize the random swap technique of EDA since it changes the semantic and syntactic meaning of the sentence. We primarily selected the random word removal technique for BARBE to generate the neighborhood instances since this technique does not increase the vocabulary. But our methodology ensured that this approach should not create empty sentences which happened for LIME.

Our explainer was evaluated against other state-of-the-art explainers, including LIME, Anchor, and SHAP. The experimental analysis in Chapter 5 demonstrated that BARBE outperformed other explainers in generating high-quality explanations. BARBE generates rules in an “if-then-else” format that is highly readable for humans, setting it apart from other explainers such as LIME, Anchor, and SHAP. Each rule comes with support, confidence, and statistical significance value. BARBE also highlights the feature using a color gradient to denote the importance of the feature within the provided text along with a feature importance score assigned to the important features. We provided an application of BARBE in detecting cyberbullying for resource-constraint languages in Chapter 6.

We conducted a human evaluation where human participants created a “ground truth” to systematically compare the explanations provided by BARBE and LIME. The “ground truth” was used to estimate the explanation performance on sentiment analysis task explanations.

Our research work paves the way for further research in various directions. One important area of research is to improve the ability of BARBE with SigDirect to handle datasets with a large number of features. This will involve exploring different methods for organizing the feature space to improve the efficiency and accuracy of the method. Expanding the scope of BARBE with SigDirect and BARBE with CFAR to handle different types of data such as speech, and images are another potential direction to extend our research.

Bibliography

[1] M. H. M. Shabestari, *Explaining decisions of black-box models using association rules*, 2020.

[2] J. Li and O. R. Zaiane, "Exploiting statistically significant dependent rules for associative classification," *Intell. Data Anal.*, vol. 21, no. 5, 1155–1172, 2017, ISSN: 1088-467X. DOI: 10.3233/IDA-163141. [Online]. Available: https://doi.org/10.3233/IDA-163141.

[3] M. R. Kabir and O. R. Zaiane, "Classification by frequent association rules," in *Proceedings of the 38th ACM/SIGAPP Symposium on Applied Computing*, Tallinn, Estonia, 2023.

[4] M. T. Ribeiro, S. Singh, and C. Guestrin, ""why should I trust you?": Explaining the predictions of any classifier," *CoRR*, vol. abs/1602.04938, 2016. arXiv: 1602.04938. [Online]. Available: http://arxiv.org/abs/1602.04938.

[5] M. T. Ribeiro, S. Singh, and C. Guestrin, "Anchors: High-precision model-agnostic explanations," in *AAAI Conference on Artificial Intelligence (AAAI)*, 2018.

[6] S. M. Lundberg and S. Lee, "A unified approach to interpreting model predictions," *CoRR*, vol. abs/1705.07874, 2017. arXiv: 1705.07874. [Online]. Available: http://arxiv.org/abs/1705.07874.

[7] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, and W. Samek, "On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation," *PloS one*, vol. 10, no. 7, e0130140, 2015.

[8] A. Shrikumar, P. Greenside, and A. Kundaje, "Learning important features through propagating activation differences," *CoRR*, vol. abs/1704.02685, 2017. arXiv: 1704.02685. [Online]. Available: http://arxiv.org/abs/1704.02685.

[9] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, ISSN: 0899-7667. DOI: 10.1162/neco.1997.9.8.1735. eprint: https://direct.mit.edu/neco/article-pdf/9/8/1735/813796/neco.1997.9.8.1735.pdf. [Online]. Available: https://doi.org/10.1162/neco.1997.9.8.1735.

[10] W. S. Noble, "What is a support vector machine?" *Nature Biotechnology*, vol. 24, pp. 1565–1567, 2006.

[11] L. Breiman, "Random forests," *Machine learning*, vol. 45, pp. 5–32, 2001.

[12] J. Wei and K. Zou, "Eda: Easy data augmentation techniques for boosting performance on text classification tasks," *arXiv preprint arXiv:1901.11196*, 2019.

[13] R.-K. Sheu and M. S. Pardeshi, "A survey on medical explainable ai (xai): Recent progress, explainability approach, human interaction and scoring system," *Sensors*, vol. 22, no. 20, 2022, ISSN: 1424-8220. DOI: 10.3390/s22208068. [Online]. Available: https://www.mdpi.com/1424-8220/22/20/8068.

[14] A. Hanif, *Towards explainable artificial intelligence in banking and financial services*, 2021. DOI: 10.48550/ARXIV.2112.08441. [Online]. Available: https://arxiv.org/abs/2112.08441.

[15] L. Górski and S. Ramakrishna, "Explainable artificial intelligence, lawyer's perspective," New York, NY, USA: Association for Computing Machinery, 2021, ISBN: 9781450385268. DOI: 10.1145/3462757.3466145. [Online]. Available: https://doi.org/10.1145/3462757.3466145.

[16] B. Zhao, X. Le, and J. Xi, *A novel sdass descriptor for fully encoding the information of 3d local surface*, 2017. DOI: 10.48550/ARXIV.1711.05368. [Online]. Available: https://arxiv.org/abs/1711.05368.

[17] J. Dastin, *Amazon scraps secret ai recruiting tool that showed bias against women*, https://www.reuters.com/article/us-amazon-com-jobs-automation-insight/amazon-scraps-secret-ai-recruiting-tool-that-showed-bias-against-women-idUSKCN1MK08G, 2018.

[18] Wikipedia, *General data protection regulation*, https://en.wikipedia.org/wiki/General_Data_Protection_Regulation, 2020.

[19] Wikipedia, *Equal credit opportunity act*, https://en.wikipedia.org/wiki/Equal_Credit_Opportunity_Act, 2020.

[20] J. Doe, *DARPA xai BAA*, https://www.darpa.mil/attachments/DARPA-BAA-16-53.pdf, 2016.

[21] R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, D. Pedreschi, and F. Giannotti, *A survey of methods for explaining black box models*, 2018. DOI: 10.48550/ARXIV.1802.01933. [Online]. Available: https://arxiv.org/abs/1802.01933.

[22] J. A. Nelder and R. W. Wedderburn, "Generalized linear models," *Journal of the Royal Statistical Society: Series A (General)*, vol. 135, no. 3, pp. 370–384, 1972.

[23] J. R. Quinlan, *C4. 5: programs for machine learning.* Elsevier, 2014.

[24] S. Atakishiyev *et al.*, "A multi-component framework for the analysis and design of explainable artificial intelligence," *arXiv preprint arXiv:2005.01908*, 2020.

[25] A. Moore, Y. Cai, K. Jones, and V. Murdock, "Tree ensemble explainability," 2017.

[26] B. Pang, L. Lee, *et al.*, "Opinion mining and sentiment analysis," *Foundations and Trends® in information retrieval*, vol. 2, no. 1–2, pp. 1–135, 2008.

[27] R. Girshick, J. Donahue, T. Darrell, and J. Malik, *Rich feature hierarchies for accurate object detection and semantic segmentation*, 2013. DOI: 10.48550/ARXIV.1311.2524. [Online]. Available: https://arxiv.org/abs/1311.2524.

[28] D. Bahdanau, K. Cho, and Y. Bengio, *Neural machine translation by jointly learning to align and translate*, 2014. DOI: 10.48550/ARXIV.1409.0473. [Online]. Available: https://arxiv.org/abs/1409.0473.

[29] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[30] E. Choi, M. T. Bahadori, J. Sun, J. Kulas, A. Schuetz, and W. Stewart, "Retain: An interpretable predictive model for healthcare using reverse time attention mechanism," *Advances in neural information processing systems*, vol. 29, 2016.

[31] R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, and D. Pedreschi, "A survey of methods for explaining black box models," *ACM computing surveys (CSUR)*, vol. 51, no. 5, p. 93, 2018.

[32] A. Shahroudnejad, "A survey on understanding, visualizations, and explanation of deep neural networks," *preprint arXiv:2102.01792*, 2021.

[33] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 618–626.

[34] L. S. Whitmore, A. George, and C. M. Hudson, "Mapping chemical performance on molecular structures using locally interpretable explanations," *arXiv preprint arXiv:1611.07443*, 2016.

[35] S. Mishra, B. L. Sturm, and S. Dixon, "Local interpretable model-agnostic explanations for music content analysis.," in *ISMIR*, vol. 53, 2017, pp. 537–543.

[36] C. Modarres, M. Ibrahim, M. Louie, and J. Paisley, "Towards explainable deep learning for credit lending: A case study," *arXiv preprint arXiv:1811.06471*, 2018.

[37] S. H. Villarroya and Z. Akata, "Interpretability in sequence tagging models for named entity recognition by sofia herrero villarroya," 2018.

[38] A. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, "Learning word vectors for sentiment analysis," in *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, 2011, pp. 142–150.

[39] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," *Advances in neural information processing systems*, vol. 30, 2017.

[40] P. Hall, N. Gill, M. Kurka, and W. Phan, "Machine learning interpretability with h2o driverless ai," *H2O. ai*, 2017.

[41] L. Hu, J. Chen, V. N. Nair, and A. Sudjianto, "Locally interpretable models and effects based on supervised partitioning (lime-sup)," *arXiv preprint arXiv:1806.00663*, 2018.

[42] R. Guidotti, A. Monreale, S. Ruggieri, D. Pedreschi, F. Turini, and F. Giannotti, "Local rule-based explanations of black box decision systems," *arXiv preprint arXiv:1805.10820*, 2018.

[43] Y. Jia, J. Bailey, K. Ramamohanarao, C. Leckie, and X. Ma, "Exploiting patterns to explain individual predictions," *Knowledge and Information Systems*, pp. 1–24, 2019.

[44] J. Han, J. Pei, and Y. Yin, "Mining frequent patterns without candidate generation," in *ACM sigmod record*, vol. 29, 2000, pp. 1–12.

[45] E. Pastor and E. Baralis, "Explaining black box models by means of local rules," in *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*, 2019, pp. 510–517.

[46] L. S. Shapley, *A Value for N-Person Games*. Santa Monica, CA: RAND Corporation, 1952. DOI: 10.7249/P0295.

[47] Q. Xie, Z. Dai, E. Hovy, M.-T. Luong, and Q. V. Le, *Unsupervised data augmentation for consistency training*, 2019. DOI: 10.48550/ARXIV.1904.12848. [Online]. Available: https://arxiv.org/abs/1904.12848.

[48] E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, and Q. V. Le, "Autoaugment: Learning augmentation policies from data," *arXiv preprint arXiv:1805.09501*, 2018.

[49] O. Kolomiyets, S. Bethard, and M.-F. Moens, "Model-portability experiments for textual temporal analysis," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, Portland, Oregon, USA: Association for Computational Linguistics, Jun. 2011, pp. 271–276. [Online]. Available: https://aclanthology.org/P11-2047.

[50] Y. Li, T. Cohn, and T. Baldwin, "Robust training under linguistic adversity," in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, Valencia, Spain: Association for Computational Linguistics, Apr. 2017, pp. 21–27. [Online]. Available: https://aclanthology.org/E17-2004.

[51] A. Mosolova, V. Fomin, and I. Bondarenko, "Text augmentation for neural networks.," in *AIST (Supplement)*, 2018, pp. 104–109.

[52] M. Jungiewicz and A. Smywiński-Pohl, "Towards textual data augmentation for neural networks: Synonyms and maximum loss," *Computer Science*, vol. 20, 2019.

[53] X. Zuo, Y. Chen, K. Liu, and J. Zhao, "KnowDis: Knowledge enhanced data augmentation for event causality detection via distant supervision," in *Proceedings of the 28th International Conference on Computational Linguistics*, Barcelona, Spain (Online): International Committee on Computational Linguistics, Dec. 2020, pp. 1544–1550. DOI: 10.18653/v1/2020.coling-main.135. [Online]. Available: https://aclanthology.org/2020.coling-main.135.

[54] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," *Advances in neural information processing systems*, vol. 28, 2015.

[55] V. Marivate and T. Sefara, "Improving short text classification through global augmentation methods," in *International Cross-Domain Conference for Machine Learning and Knowledge Extraction*, Springer, 2020, pp. 385–399.

[56] K. Lu, P. Mardziel, F. Wu, P. Amancharla, and A. Datta, "Gender bias in neural natural language processing," in *Logic, Language, and Security*, Springer, 2020, pp. 189–202.

[57] C. Coulombe, "Text data augmentation made simple by leveraging nlp cloud apis," *arXiv preprint arXiv:1812.04718*, 2018.

[58] R. Xiang, E. Chersoni, Y. Long, Q. Lu, and C.-R. Huang, "Lexical data augmentation for text classification in deep learning," in *Canadian Conference on Artificial Intelligence*, Springer, 2020, pp. 521–527.

[59] W. Y. Wang and D. Yang, "That's so annoying!!!: A lexical and frame-semantic embedding based data augmentation approach to automatic categorization of annoying behaviors using# petpeeve tweets," in *Proceedings of the 2015 conference on empirical methods in natural language processing*, 2015, pp. 2557–2563.

[60] J. Li, S. Ji, T. Du, B. Li, and T. Wang, "Textbugger: Generating adversarial text against real-world applications," *arXiv preprint arXiv:1812.05271*, 2018.

[61] G. Rizos, K. Hemker, and B. Schuller, "Augment to prevent: Short-text data augmentation in deep learning for hate-speech classification," in *Proceedings of the 28th ACM international conference on information and knowledge management*, 2019, pp. 991–1000.

[62] X. Zhang and Y. LeCun, "Text understanding from scratch," *arXiv preprint arXiv:1502.01710*, 2015.

[63] T. H. Huong and V. T. Hoang, "A data augmentation technique based on text for vietnamese sentiment analysis," in *Proceedings of the 11th International Conference on Advances in Information Technology*, 2020, pp. 1–5.

[64] K. J. Madukwe, X. Gao, and B. Xue, "Token replacement-based data augmentation methods for hate speech detection," *World Wide Web*, vol. 25, no. 3, pp. 1129–1150, 2022.

[65] N. Mrkšić *et al.*, "Counter-fitting word vectors to linguistic constraints," in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, San Diego, California: Association for Computational Linguistics, Jun. 2016, pp. 142–148. DOI: 10.18653/v1/N16-1018. [Online]. Available: https://aclanthology.org/N16-1018.

[66] O. Levy and Y. Goldberg, "Dependency-based word embeddings," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, Baltimore, Maryland: Association for Computational Linguistics, Jun. 2014, pp. 302–308. DOI: 10.3115/v1/P14-2050. [Online]. Available: https://aclanthology.org/P14-2050.

[67] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95-international conference on neural networks*, IEEE, vol. 4, 1995, pp. 1942–1948.

[68] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic attribution for deep networks," in *International conference on machine learning*, PMLR, 2017, pp. 3319–3328.

[69] M. Alzantot, Y. Sharma, A. Elgohary, B.-J. Ho, M. Srivastava, and K.-W. Chang, "Generating natural language adversarial examples," *arXiv preprint arXiv:1804.07998*, 2018.

[70] C. Chelba *et al.*, "One billion word benchmark for measuring progress in statistical language modeling," *arXiv preprint arXiv:1312.3005*, 2013.

[71] F. Gao *et al.*, "Soft contextual data augmentation for neural machine translation," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 5539–5544.

[72] S. Kobayashi, "Contextual augmentation: Data augmentation by words with paradigmatic relations," *arXiv preprint arXiv:1805.06201*, 2018.

[73] M. Fadaee, A. Bisazza, and C. Monz, "Data augmentation for low-resource neural machine translation," *arXiv preprint arXiv:1705.00440*, 2017.

[74] Y. Belinkov and Y. Bisk, "Synthetic and natural noise both break neural machine translation," *arXiv preprint arXiv:1711.02173*, 2017.

[75] S. Y. Feng, V. Gangal, D. Kang, T. Mitamura, and E. Hovy, "Genaug: Data augmentation for finetuning text generators," *arXiv preprint arXiv:2010.01794*, 2020.

[76] A. Karimi, L. Rossi, and A. Prati, "Aeda: An easier data augmentation technique for text classification," *arXiv preprint arXiv:2108.13230*, 2021.

[77] J. Ebrahimi, A. Rao, D. Lowd, and D. Dou, "Hotflip: White-box adversarial examples for text classification," *arXiv preprint arXiv:1712.06751*, 2017.

[78] Z. Miao, Y. Li, X. Wang, and W.-C. Tan, "Snippext: Semi-supervised opinion mining with augmented data," in *Proceedings of The Web Conference 2020*, 2020, pp. 617–628.

[79] T. Niu and M. Bansal, "Adversarial over-sensitivity and over-stability strategies for dialogue models," *arXiv preprint arXiv:1809.02079*, 2018.

[80] M. Artetxe, G. Labaka, E. Agirre, and K. Cho, "Unsupervised neural machine translation," *arXiv preprint arXiv:1710.11041*, 2017.

[81] G. Lample, A. Conneau, L. Denoyer, and M. Ranzato, "Unsupervised machine translation using monolingual corpora only," *arXiv preprint arXiv:1711.00043*, 2017.

[82] J. Wang, H.-C. Chen, R. Radach, and A. Inhoff, *Reading Chinese script: A cognitive analysis*. Psychology Press, 1999.

[83] X. Dai and H. Adel, "An analysis of simple data augmentation for named entity recognition," *arXiv preprint arXiv:2010.11683*, 2020.

[84] M. Regina, M. Meyer, and S. Goutal, "Text data augmentation: Towards better detection of spear-phishing emails," *arXiv preprint arXiv:2007.02033*, 2020.

[85] Z. Xie *et al.*, "Data noising as smoothing in neural network language models," *arXiv preprint arXiv:1703.02573*, 2017.

[86] J. Wei, C. Huang, S. Vosoughi, Y. Cheng, and S. Xu, "Few-shot text classification with triplet networks, data augmentation, and curriculum learning," *arXiv preprint arXiv:2103.07552*, 2021.

[87] B. Peng, C. Zhu, M. Zeng, and J. Gao, "Data augmentation for spoken language understanding via pretrained language models," *arXiv preprint arXiv:2004.13952*, 2020.

[88] X. Song, L. Zang, and S. Hu, "Data augmentation for copy-mechanism in dialogue state tracking," in *International Conference on Computational Science*, Springer, 2021, pp. 736–749.

[89] G. Yan, Y. Li, S. Zhang, and Z. Chen, "Data augmentation for deep learning of judgment documents," in *International Conference on Intelligent Science and Big Data Engineering*, Springer, 2019, pp. 232–242.

[90] S. Yu, J. Yang, D. Liu, R. Li, Y. Zhang, and S. Zhao, "Hierarchical data augmentation and the application in text classification," *IEEE Access*, vol. 7, pp. 185 476–185 485, 2019.

[91] R. C. Holte, "Very simple classification rules perform well on most commonly used datasets," *Mach. Learn.*, vol. 11, no. 1, 63–90, 1993, ISSN: 0885-6125. DOI: 10.1023/A:1022631118932. [Online]. Available: https://doi.org/10.1023/A:1022631118932.

[92] J. R. Quinlan, "Learning logical definitions from relations," *Mach. Learn.*, vol. 5, no. 3, 239–266, 1990, ISSN: 0885-6125. DOI: 10.1023/A:1022699322624. [Online]. Available: https://doi.org/10.1023/A:1022699322624.

[93] W. W. Cohen, "Fast effective rule induction," in *Machine learning proceedings 1995*, Elsevier, 1995, pp. 115–123.

[94] R. Agrawal, R. Srikant, *et al.*, "Fast algorithms for mining association rules," in *Proc. 20th int. conf. very large data bases, VLDB*, Santiago, Chile, vol. 1215, 1994, pp. 487–499.

[95] J. Han, J. Pei, and Y. Yin, "Mining frequent patterns without candidate generation," *SIGMOD Rec.*, vol. 29, no. 2, 1–12, 2000, ISSN: 0163-5808. DOI: 10.1145/335191.335372. [Online]. Available: https://doi.org/10.1145/335191.335372.

[96] B. Liu, W. Hsu, Y. Ma, *et al.*, "Integrating classification and association rule mining.," in *Kdd*, vol. 98, 1998, pp. 80–86.

[97] W. Li, J. Han, and J. Pei, "Cmar: Accurate and efficient classification based on multiple class-association rules," in *Proceedings 2001 IEEE international conference on data mining*, IEEE, 2001, pp. 369–376.

[98] X. Yin and J. Han, "Cpar: Classification based on predictive association rules," in *Proceedings of the 2003 SIAM international conference on data mining*, SIAM, 2003, pp. 331–335.

[99] W. Hamalainen, "Efficient discovery of the top-k optimal dependency rules with fisher's exact test of significance," in *2010 IEEE International Conference on Data Mining*, 2010, pp. 196–205. DOI: 10.1109/ICDM.2010.143.

[100] M. Wilcox, S. Schuermans, and C. Voskoglou, "Developer economics, state of the developer nation," VisionMobile Ltd, Tech. Rep. 2016.

[101] O. R. Zaïane and M.-L. Antonie, "On pruning and tuning rules for associative classifiers," in *Knowledge-Based Intelligent Information and Engineering Systems*, R. Khosla, R. J. Howlett, and L. C. Jain, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 966–973, ISBN: 978-3-540-31990-0.

[102] T. Miller, "Explanation in artificial intelligence: Insights from the social sciences," *Artificial Intelligence*, 2018.

[103] A. Gulli, *AG's News Topic Classification Dataset*, http://groups.di.unipi.it/~gulli/AG_corpus_of_news_articles.html, 2005.

[104] F. Sebastiani, "Machine learning in automated text categorization," *ACM computing surveys (CSUR)*, vol. 34, no. 1, pp. 1–47, 2002.

[105] J. Ramos *et al.*, "Using tf-idf to determine word relevance in document queries," in *Proceedings of the first instructional conference on machine learning*, Citeseer, vol. 242, 2003, pp. 29–48.

[106] D. Slack, S. Hilgard, E. Jia, S. Singh, and H. Lakkaraju, *Fooling lime and shap: Adversarial attacks on post hoc explanation methods*, 2020. arXiv: 1911.02508 [cs.LG].

[107] M. Collins, "Head-driven statistical models for natural language parsing," *Computational linguistics*, vol. 29, no. 4, pp. 589–637, 2003.

[108] J. J. Purtilo and J. R. Callahan, "Parse tree annotations," *Communications of the ACM*, vol. 32, no. 12, pp. 1467–1477, 1989.

[109] C. D. Manning, M. Surdeanu, J. Bauer, J. R. Finkel, S. Bethard, and D. McClosky, “The stanford corenlp natural language processing toolkit,” in *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, 2014, pp. 55–60.

[110] C. L. Odgers and M. R. Jensen, “Annual research review: Adolescent mental health in the digital age: Facts, fears, and future directions,” *Journal of Child Psychology and Psychiatry*, vol. 61, no. 3, pp. 336–348, 2020.

[111] Thirusangu and Senthil, “Cyber disorders on adolescence depression and suicide mental health concern for the new millennium,” *International Journal of Advances in Nursing Management*, vol. 7, no. 3, pp. 281–284, 2019.

[112] S. Hossen, “Nature and aftermath of cyberbullying with female university students in bangladesh,”

[113] Abdhullah-Al-Mamun and S. Akhter, “Social media bullying detection using machine learning on bangla text,” in *2018 10th International Conference on Electrical and Computer Engineering (ICECE)*, 2018, pp. 385–388. DOI: 10.1109/ICECE.2018.8636797.

[114] E. Boiy and M.-F. Moens, “A machine learning approach to sentiment analysis in multilingual web texts,” *Information retrieval*, vol. 12, no. 5, pp. 526–558, 2009.

[115] M Ikonomakis, S. Kotsiantis, and V Tampakas, “Text classification using machine learning techniques.,” *WSEAS transactions on computers*, vol. 4, no. 8, pp. 966–974, 2005.

[116] G. A. León-Paredes *et al.*, “Presumptive detection of cyberbullying on twitter through natural language processing and machine learning in the spanish language,” in *2019 IEEE CHILEAN Conference on Electrical, Electronics Engineering, Information and Communication Technologies (CHILECON)*, IEEE, 2019, pp. 1–7.

[117] E. Raisi and B. Huang, “Cyberbullying detection with weakly supervised machine learning,” in *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2017*, 2017, pp. 409–416.

[118] S. Agrawal and A. Awekar, “Deep learning for detecting cyberbullying across multiple social media platforms,” in *European conference on information retrieval*, Springer, 2018, pp. 141–153.

[119] M. Gogoi and S. K. Sarma, “Document classification of assamese text using naïve bayes approach,” *International Journal of Computer Trends and Technology*, vol. 30, pp. 182–186, 2015.

[120] B. Haidar, M. Chamoun, and A. Serhrouchni, “A multilingual system for cyberbullying detection: Arabic content detection using machine learning,” *Advances in Science, Technology and Engineering Systems Journal*, vol. 2, no. 6, pp. 275–284, 2017.

[121] N. Krail and V. Gupta, "Domain based classification of punjabi text documents using ontology and hybrid based approach," in *WSSANLP@COLING*, 2012.

[122] S. Bhattacharya, M. Choudhury, and A. Basu, "Inflectional morphology synthesis for bengali noun, pronoun and verb systems," 2005.

[123] H. A. Chowdhury, M. A. H. Imon, S. M. Hasnayeen, and M. S. Islam, "Authorship attribution in bengali literature using convolutional neural networks with fasttext's word embedding model," in *2019 1st International Conference on Advances in Science, Engineering and Robotics Technology (ICASERT)*, IEEE, 2019, pp. 1–5.

[124] M. F. Ahmed, Z. Mahmud, Z. T. Biash, A. A. N. Ryen, A. Hossain, and F. B. Ashraf, *Cyberbullying detection using deep neural network from social media comments in bangla language*, 2021. arXiv: 2106.04506 `[cs.CL]`.

[125] S. Qaiser and R. Ali, "Text mining: Use of tf-idf to examine the relevance of words to documents," *International Journal of Computer Applications*, vol. 181, no. 1, pp. 25–29, 2018.

[126] A. M. Kibriya, E. Frank, B. Pfahringer, and G. Holmes, "Multinomial naive bayes for text categorization revisited," in *Australasian Joint Conference on Artificial Intelligence*, Springer, 2004, pp. 488–499.

[127] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[128] M. Pal, "Random forest classifier for remote sensing classification," *International journal of remote sensing*, vol. 26, no. 1, pp. 217–222, 2005.

[129] 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, pp. 785–794.

[130] K. O'Shea and R. Nash, "An introduction to convolutional neural networks," *arXiv preprint arXiv:1511.08458*, 2015.

[131] T. Mikolov, K. Chen, G. Corrado, and J. Dean, *Efficient estimation of word representations in vector space*, 2013.

[132] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, *Enriching word vectors with subword information*, 2016.

[133] M. L. McHugh, "Interrater reliability: The kappa statistic," *Biochemia medica*, vol. 22, no. 3, pp. 276–282, 2012.