

Product Entity Matching by Leveraging Tabular Data

by

Ali Naeim Abadi

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

Master of Science

Department of Computing Science
University of Alberta

© Ali Naeim Abadi, 2023

Abstract

Product Entity Matching (PEM) is a challenging subfield of record linkage that involves linking records referring to the same real-world product. Despite recent transformer models showing near-perfect performance scores on various datasets, they struggle the most when dealing with PEM datasets. In this thesis, we study PEM under the common setting where the information is spread over text and tables. To facilitate our research, we introduce a new dataset based on existing Amazon, Walmart and Google datasets, where each product contains a mix of both textual and tabular details. Our hypothesis is that leveraging detail tables alongside textual data can effectively tackle complex entity matching tasks where textual information alone falls short. However, existing models have proven to be inefficient and ineffective in utilizing such tabular data. We propose TATEM and TATTOO models, which offer an effective solution by harnessing pre-trained language models along with a novel serialization technique to encode tabular product data. Our models incorporate a novel attribute ranking module to make our model more data-efficient. Our experiments on both current benchmark datasets and our proposed datasets show significant improvements compared to state-of-the-art methods, including large language models in zero-shot and few-shot settings. Moreover, in out-of-domain and few-shot experiments, the TATTOO model showcases its superiority by outperforming strong baselines by a substantial margin.

Preface

The research presented in this thesis was undertaken in collaboration with Dr. Davood Rafiei, who served as my supervisor during my graduate studies, and Mr. Mir Tafseer Nayeem, my co-mentor. Both Dr. Rafiei and Mr. Nayeem contributed significantly by offering valuable guidance and constructive feedback throughout various stages of this study. Their expert input was particularly instrumental in refining the proposed methodology, experimental design, presentation, and written content, as I conducted experiments, analyzed data, and formulated conclusions. The cohesive efforts and support from these esteemed individuals significantly enriched the quality and rigor of this research endeavor.

*To my family and friends for their unwavering love and steadfast support throughout
this journey.*

Acknowledgements

I extend my heartfelt gratitude for the invaluable support and guidance offered by Dr. Davood Rafiei, my research supervisor, throughout the course of this research project. Undoubtedly, the successful completion of this work owes a great deal to his unwavering assistance and profound knowledge. I am pleased to acknowledge the enriching experience of honing and augmenting my research capabilities, facilitated by his generous provision of constructive feedback. Under his supervision, I was afforded the freedom to explore novel perspectives, fostering a conducive environment that allowed me to devise innovative solutions to complex problems.

Likewise, I wish to extend my appreciation to the esteemed members of the Database Research Group, with a special acknowledgment reserved for my valued colleague, Mir Tafseer Nayeem, whose exceptional intellect and innovative ideas have been a source of profound inspiration. The knowledge-sharing sessions conducted within this group have been instrumental in broadening my understanding and focus on the research problem at hand. I am deeply indebted to these interactions, as they have guided and fortified my research pursuits.

I express my heartfelt appreciation to my parents and family for their invaluable role as a continual source of inspiration and motivation throughout my life. I would like to express my deepest appreciation and gratitude to my beloved wife, Farzaneh Entezari, for her unwavering support, patience, and understanding throughout the challenging journey of completing this thesis. Her love, encouragement, and constant belief in my abilities have been my rock and my motivation.

Table of Contents

1	Introduction	1
1.1	Motivation	1
1.2	Thesis Objectives	5
1.3	Thesis Outline	7
2	Related Work	9
2.1	Probabilistic Entity Matching	9
2.2	Rule-based Entity Matching	10
2.3	Traditional Machine Learning Approaches for Entity Matching	11
2.4	Deep Learning Approaches	12
2.4.1	Entity Matching prior to Pre-trained Language Models	12
2.4.2	Pre-trained-language-models-based EM	14
2.5	EM Models Leveraging Domain Knowledge	15
2.6	Tabular Data as A Source of Knowledge	16
2.7	Gap in Research	17
3	Product Entity Matching via Tabular Data	18
3.1	Introduction	18
3.2	Dataset	20
3.3	TATEM Model	22
3.3.1	TATEM Serialization	23
3.3.2	Attribute Ranking Module (ARM)	24
3.4	EXPERIMENTAL Evaluation	25
3.4.1	Results	26
3.5	Summary	30
4	TATTOO: Product Entity Matching as a Topology Construction	31
4.1	Introduction	31
4.2	Dataset	35

4.3	TATTOO Model	37
4.3.1	TATTOO serialization	37
4.3.2	Attribute Ranking Module (ARM)	37
4.4	EXPERIMENTAL EVALUATION	39
4.4.1	Accuracy of PEM Models	39
4.4.2	Effects of ARM Modules	42
4.5	Evaluating Robustness	42
4.5.1	Few-Shot Setting	43
4.5.2	Out-of-domain Generalization	45
4.6	Summary	46
5	Conclusions, & Future Work	48
5.1	Conclusions	48
5.2	Future Work	49
	Bibliography	50

List of Tables

3.1	A few hard negative examples from Amazon-Google dataset [4]. Despite their highly similar titles, product pairs are not the same.	19
3.2	Stats. of Amazon-Google [4], Walmart-Amazon [11], Amazon*-Google (<i>ours</i>), and Walmart-Amazon* (<i>ours</i>).	21
3.3	Performance of our proposed model TATEM compared to different baselines.	28
4.1	Stats. of Amazon-Google [4], Walmart-Amazon [11], Amazon*-Google (<i>ours</i>) and Walmart*-Amazon*(<i>ours</i>). #Train & #Test denote the number of train and test samples for the datasets. #attrs denotes the number of attribute-value pairs for the tables. (N./P.) \Leftrightarrow (Num of Negative Pairs / Num of Positive Pairs).	36
4.2	Performance of TATTOO compared to different baselines. All reported results for TATTOO (<i>ours</i>) are statistically significant in paired t-test by taking DITTO (2020) as a reference with the confidence of 95% (p -value < 0.05).	41
4.3	EM results of TATTOO equipped with ARM, considering top n product table attributes from our enriched datasets. Only examples with tables are considered. Best results in each setting are marked bold	43
4.4	Out-of-domain results when a PLM-based EM model is fine-tuned on A and tested on B ($A \Rightarrow B$). Drop in performance compared to in-domain settings are presented in round brackets. Δ is the % change compared to baseline DITTO. GPT3 (k=0) in zero-shot setting is considered as out-of-domain. All training examples are considered. Best results in each setting are marked bold	46

List of Figures

1.1	A hard negative example between two Sony headphones. Despite their highly similar titles, the products are totally different.	3
1.2	A hard negative example disambiguated using an Amazon product detail table, that it is challenging to reject a match based only on the information given in titles because there are many overlapping words. An EM model can disambiguate this by forming a connection between the left product and the table (<i>if exists</i>). Here, the Product model number field helps the EM model to fill the information gap and reach a Non-Match decision.	6
3.1	A hard negative example disambiguated using an Amazon product detail table, showing that relying on the information given in titles alone is hard to vote against a match because of the large number of overlapping tokens. Our model TATEM disambiguates this by establishing a relationship between <i>e2</i> and <i>table1</i> (<i>if exists</i>). Here, the Model Number field helps TATEM to reach a Non-Match decision.	20
3.2	Our TATEM model coupled with ARM for PEM.	24
3.3	Detail tables for two products in semi-structured and structured formats.	27
4.1	A hard negative example disambiguated using product detail tables, indicating that depending only on the information provided in titles makes it challenging to confidently reject a match due to substantial missing key features and the presence of misleading and overlapping tokens. Our model TATTOO disambiguates this by establishing a relationship between entities and tables (<i>if exists</i>). Here, the highlighted fields help TATTOO to reach a Non-Match decision.	32
4.2	Topology Illustration.	33
4.3	Our TATTOO equipped with Plain ARM and Cross ARM to select top <i>n</i> attributes.	38
4.4	Few-Shot Setting Amazon*-Google.	44

4.5 Few-Shot Setting performance of TATTOO (*ours*) compared to DITTO [7] and ROBEM [9] on our enriched Walmart*-Amazon* [Quaternary (4-ary) Topology]. Bold-faced numbers on top of the bars indicate the F1 Score, while **green** and **red** colored numbers represent the percentage point increase or decrease, respectively, compared to the baseline DITTO [7]. 44

Abbreviations

A*-G Amazon*-Google dataset.

A-G Amazon-Google dataset.

ARM Attribute Ranking Module.

DK Domain Knowledge.

e.g. *exempli gratia* or for example.

EM Entity Matching.

i.e. *id est* or that is.

LLM Large Language Models.

PEM Product Entity Matching.

PLM Pre-trained Language Models.

TATEM TABLE & Text for Entity Matching model.

TATTOO TABLE & Text Entity Matching as TOPOLOGY CONSTRUCTION.

W-A* Walmart-Amazon* dataset.

W-A Walmart-Amazon dataset.

Chapter 1

Introduction

1.1 Motivation

Entity Matching (EM) refers to the procedure of recognizing and connecting identical entities from multiple databases into a unified representation [1]. EM entails the identification of records that pertain to the same real-world entity, like a person, organization, product, or location, and merging these records into unified representations, even if variations exist among different databases [2].

The process of linking entities from structured and unstructured sources is crucial in various fields and applications, such as healthcare, human resources and e-commerce [3]. For instance, accurately linking highly similar patients based on their textual medical case summaries and tabular diagnosis code data helps us create a well-informed, user-friendly clinical decision support system. In addition, the precise matching of resumes to particular job postings and the linkage of job seekers to matching organizations is essential for HR systems. Similarly, creating relationships between clients and companies, such hotels and restaurants, is of paramount importance in the world of web-based services.

An effective high-performance EM solution can benefit both business owners and their clients. From the consumer's perspective, making it easier to identify products listed for sale improves their ability to find items in niche markets. Moreover, it empowers systems that collect product information from diverse sources to provide re-

liable data, ultimately saving customers both time and money. For retailers, utilizing this product information allows them to elevate the customer experience by employing targeted advertisements, comprehensive product listings, improved content-based recommendation systems for newly launched items, and other enhancements. Comparing PEM with content-based recommendation systems, PEM models recognize if two entities are referring to the same real-world product; however, content-based recommendation systems aim to find and recommend items similar to the profiles of users. PEM and content-based recommendation systems are similar in the sense that both of them excel at finding highly similar items.

Moreover, the retail sector heavily depends on entity matching to achieve various goals efficiently. For instance, in online tools designed for facilitating price comparison, product matching is employed. This process involves comparing different product records to identify advertisements related to the same products offered by multiple vendors. Additionally, the compilation of product data from various sources allows the generation of comprehensive product catalogs and in-depth listings. Therefore, the main objective of this thesis is to develop a cost-efficient and highly effective solution for creating links between records in the domain of product matching.

The problem of entity matching has been extensively studied using various datasets [4–6], and recent techniques, such as injecting domain knowledge [7, 8], improving serialization [9], and utilizing LLMs [10], have been deployed. Here, we focus on Product Entity Matching (PEM) for two important reasons: Firstly, the PEM datasets, such as Amazon-Google [4] and Walmart-Amazon [11], remain challenging compared to many other datasets on which EM models have demonstrated near-perfect performance [10]. Secondly, enhanced PEM solutions can offer substantial benefits to numerous real-world e-commerce applications. However, we have identified some major challenges in product entity matching manifested in popular PEM benchmarks:

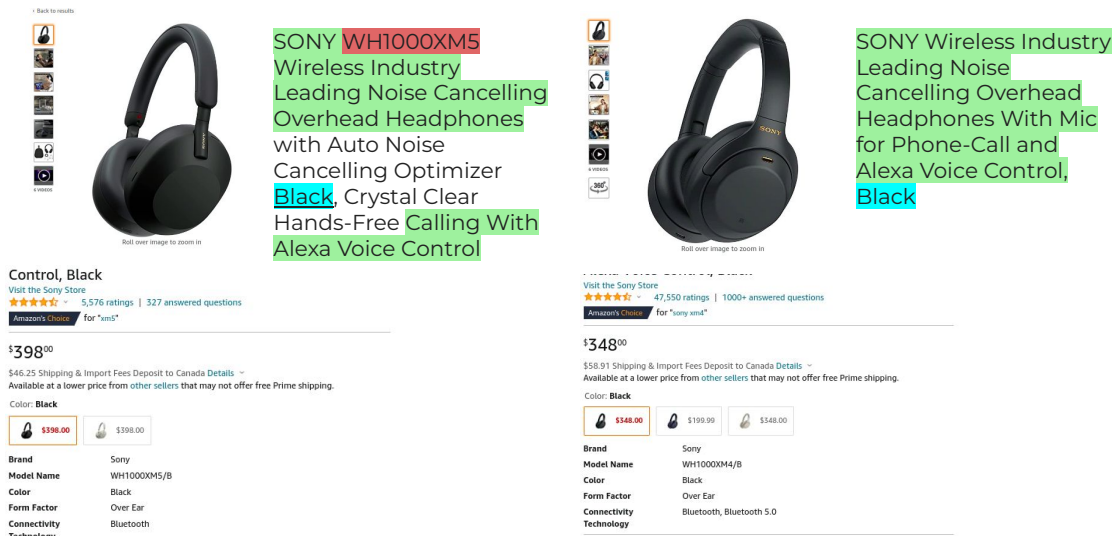


Figure 1.1: A hard negative example between two Sony headphones. Despite their highly similar titles, the products are totally different.

Hard Negative Examples Certain product titles share a high degree of similarity but are labeled as non-matching pairs, termed “hard negative examples”. Even human annotators face challenges in distinguishing these non-matching examples using the information available in the dataset. As a result, State-Of-The-Art (SOTA) models struggle to disambiguate them. Figure 1.1 illustrates one such hard negative example involving two Sony headphones listed on Amazon.com. Although these examples are considered non-matching pairs, all the words in the title of the product shown on the right are found in the title of the product shown on the left, making it intricate for the models to differentiate between them.

Attribute Identification Nowadays product titles play a pivotal role in encoding the distinctive attributes of products. Nevertheless, specific segments within a title, such as the model, year introduced, functionalities, etc., bear critical significance in the process of making matching decisions. The placement of this essential information may vary across different products. Therefore, the model may not have direct access to the encoded attributes. For instance, in the examples shown in Figure 1.1, the color “*Black*” appears in the middle of the left product title and at the end of the

right one. An EM model should be capable of recognizing the importance of such information and accurately recognizing the attribute it describes.

Fixed Attributes In all popular EM datasets [6, 12, 13], a fixed number of attributes is given for all samples (e.g., title, manufacturer, and price for the Amazon-Google dataset [4]). However, different products can have different number of characteristics and key attributes, Posing a limitation on the number of attributes is prone to neglecting pivotal product attributes.

Generalization to Unseen Domain Entities While current prominent EM models [7, 9, 14] achieve high performance for matching entities within the domain of the training data, our experiments involving entities from unseen domains—those not covered by the training data—indicate that the methods perform significantly worse for such entities. Our experiments in Section 4.5.2 confirm that all the existing EM models demonstrate a significant decline in performance when evaluated on an unseen domain. For example, new products frequently appear in e-commerce scenarios, and new books and articles are central to bibliographic data management.

Low-Resource EM In order to meet the requirements of various applications, there is a need for a solution that can operate effectively under low-resource conditions [15], meaning it should be capable of functioning with only a limited number of labeled examples. Typically, current EM approaches [7, 9] employ supervised learning methods that make use of Pre-trained Language Models (PLM). However, these approaches heavily rely on having a substantial quantity of high-quality domain-specific labeled training data. Our experiments in Section 4.5.1 show that existing EM models encounter difficulties when challenged with low-resource conditions.

Lack of Domain Knowledge To implement an informed model for disambiguating hard negative examples, it is essential to have an additional source of product domain

knowledge to supplement the model. KAER [8] employs Wikidata as a knowledge graph for the purpose of integrating external knowledge at both the schema and entity levels. Nevertheless, this approach is inclined to exhibit reduced efficacy when dealing with challenging hard negative examples, since such instances generally pertain to the same category within the schema level. Furthermore, newly introduced and less widely recognized products are less likely to be discovered within the knowledge graph. DITTO [7] highlights the importance of extra domain knowledge for EM by adding Named Entity Recognition (NER) tags from spaCy [16] and rewriting text spans with developer-specified rules (e.g., replacing 5 % and 5.00 % with 5.0%). However, these measures are not domain specific, and the same technique is deployed for every domain (dataset).

1.2 Thesis Objectives

Provide Product Specific Domain Knowledge We want to enrich existing PEM benchmarks (e.g., from [12]) by introducing product detail tables as an additional source of product domain knowledge. These tables serve as crucial links, facilitating the connection between two entities of interest, thus potentially enhancing the accuracy of matching decisions. Within the datasets, each product exhibits a varying number of attributes. By integrating the detail table, we are able to directly capture all the characteristic features of products, contributing to a more comprehensive and informed matching process. Consider the example shown in Figure 1.2, where a model number is provided for the left product; however, this specific model number is not given in the title of the right product. The presence of the model number in the detail table serves as a connecting element that fills the gap between the two product descriptions.

However, including detail tables introduces new challenges to entity matching. For instance, as shown in our dataset analysis in Section 4.2, one of our enriched datasets includes 826 unique features and tables with up to 81 attributes. Therefore, it imposes

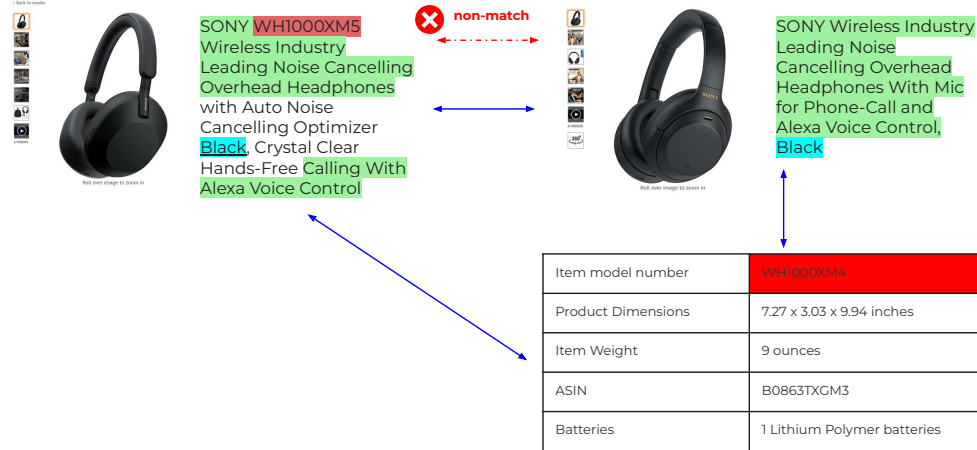


Figure 1.2: A hard negative example disambiguated using an Amazon product detail table, that it is challenging to reject a match based only on the information given in titles because there are many overlapping words. An EM model can disambiguate this by forming a connection between the left product and the table (*if exists*). Here, the **Product model number** field helps the EM model to fill the information gap and reach a **Non-Match** decision.

new challenges for table serialization using PLMs and a data-efficient solution for long product detail tables.

Design Attribute Ranking Module (ARM) Enriched datasets have many attributes per product, making it hard to determine key indicators for matches or non-matches. To address this challenge, we want to develop techniques dedicated to retrieving and ranking the most essential attribute-value pairs. These methods are vital for improving the matching process and increasing the accuracy of our entity matching system, especially in difficult scenarios.

Develop A Model for Semi-structured Data Prominent EM models [7, 9, 10, 12, 14] are implemented for existing EM benchmarks [4, 6, 11–13] which are structured with a preset number of features. We want to introduce an enriched dataset with a varying number of attributes and a varying schema for products. Our experimental findings (§3.4, §4.4.1) confirm that converting our semi-structured data to a structured format leads to a sparse and unnecessarily wide dataset. This

structured dataset suppresses the PLM-based model capabilities and the potential of the enriched data. Therefore, we need to devise an EM model and serialization technique specially devised for the semi-structured format of our dataset.

1.3 Thesis Outline

The thesis is organized as follows:

Chapter 2 reviews the background and related work about EM. Scientists have referred to entity linkage since 1959 [17], and have utilized different approaches for it. We discuss EM solutions by leveraging probabilistic, rule-based, traditional machine learning, active learning, deep learning, and PLMs.

Chapter 3 includes product entity matching via tabular data using our **TA**ble & **T**ext for **E**ntity **M**atching (**TATEM**) model. We introduce two enriched datasets (Amazon*-Google and Walmart-Amazon*) with Amazon product detail tables (§3.2). The PLM-based model employs a new serialization technique (§3.3.1) devised for our semi-structured enriched datasets. Additionally, to find the top n most essential detail table attributes, we develop an Attribute Ranking Module (ARM) (§3.3.2). In fact, ARM has the potential to substantially decrease the number of tokens sent to a PLM. A proficient ARM should bring matching pairs closer together and create greater separation among non-matching pairs using only a few attributes. Thanks to its effective serialization technique and informative product detail tables, TATEM obtains new SOTA results for both Amazon-Google and Walmart-Amazon datasets (§3.4).

Chapter 4 introduces **TATTOO** (**TA**ble & **T**ext Entity Matching as **TO**pology **CO**nstruction): product entity matching as a topology construction. We create Walmart*-Amazon* dataset benefiting from product detail tables for both Walmart

and Amazon items. TATTOO reaches new SOTA results for Walmart-Amazon and Amazon-Google datasets. Integrating ARM into TATTOO results in a high-performance, data-efficient entity matching system. We develop two versions of ARM (plain ARM and cross ARM) to optimize product table utilization on both sides (§4.3.2). TATTOO exhibits outstanding robust behavior when faced with reduced training size (§4.5.1). Trained on 5% of Walmart*-Amazon* dataset, TATTOO is leading with 39.9% enhancement over DITTO (77.14 vs 37.24). Additionally, our model exhibits promising out-of-domain robust behavior (§4.5.2) in zero-shot setting. In our experiment settings, TATTOO can outperform zero-shot GPT3 and few-shot GPT3(K=10) on Amazon-Google dataset, and our model achieves 31.60% improvement compared with DITTO.

Chapter 2

Related Work

Canadian researchers [17] in 1959 introduced the term *record linkage* in a paper published in the journal of *Science*. Since then, Entity Matching (EM) has been an important research problem with different solutions developed in the literature, such as rule-based (§2.2), traditional Machine Learning (ML) (§2.3), Deep Learning (DL) (§2.4), and Pre-trained Language Model (PLM) (§2.4.2). This chapter reviews the related work on entity matching with a focus on Product Entity Matching (PEM). Our review includes not only traditional and modern approaches based on machine learning but also some of the work that attempt to use Domain Knowledge (DK)(§2.5) as well as some of the models developed for tabular data (§2.6). Finally, we discuss the gap in EM research (§2.7) and suggest a source of product-specific DK to build a more effective solution for product entity matching.

2.1 Probabilistic Entity Matching

Newcombe, Kennedy, Axford, and James [17] were the first to identify record linkage as a Bayesian inference problem. Later, Fellegi and Sunter [18] formalized this intuition and introduced the notations that have been frequently used in the literature. Jaro suggests an expectation maximization algorithm [20] to compute the probabilities. Later, Winkler [21] propose a general expectation maximization algorithm to estimate conditional probabilities when conditional independence is not a reasonable

assumption.

Interestingly Fellegi and Sunter [18] introduce a “reject” class in addition to “match” (M) and “nonmatch” (U) classes for a pair of records. The reject class contains pairs of entities for which it is not possible to make a definitive decision and a “clerical review” is required by an expert.

Often, only focusing on the minimization of the probability of error is not the best metric for generating decision rules because the misclassification of each class may have different consequences. Tepping [22] was the first to suggest a solution for the cost of decisions. Later Verykios, Moustakides, and Elfeky [23] developed a formal framework to compute thresholds for the three decision areas, i.e. M, U, and Reject.

2.2 Rule-based Entity Matching

A well-known EM technique is defining a set of rules that determines the conditions under which two entities are considered matching. For instance, a rule may indicate that two records will be considered matching if they fully match the name field and the similarities between the fields of street name and city are higher than 0.8. This may be written as

$$(name, Jaccard, =, 1) \wedge (street_name, Jaccard, >, 0.8) \wedge (city, Jaccard, >, 0.8)$$

where the second argument in each condition gives the distance or similarity function being used. In this case, the similarity measure between the attributes is Jaccard index.

For early rule-based models [24, 25], an expert has to get into the intricate process of manually defining the rules and deciding on what attributes should be considered, what similarity function should be used between those attributes, what parameters and thresholds the similarity functions should run, which rules are of greater importance, and which configuration of rules works best for a specific domain. A new line of studies tries to address the above challenges and design an ML model to find the attributes, similarity functions, and rule configuration [26–29] using a given set of

examples. However, they study the challenges only in isolation.

Recently Paganelli, Sottovia, Guerra, and Velegrakis, 2019 [30] introduce a model which optimizes the similarity function, the variables and thresholds of the similarity function, and the weights of the rules, and obtains promising results.

2.3 Traditional Machine Learning Approaches for Entity Matching

Supervised Machine Learning (ML) approaches rely on training data in the form of pairs, labeled as matching or non-matching to learn an EM model. Cochinwala, Kurien, Lalk, and Shasha [31] utilize well-known Classification And Regression Tree (CART) [32] to create classification trees. However, they consider record pairs independently, similar to probabilistic approaches. Bilenko, Mooney, Cohen, Ravikumar, and Fienberg [33] employ an SVM classifier [34] to train a model for merging matching results for the individual attributes of records. The SVM model can beat other techniques that process the whole entity as one large attribute.

One group of studies constructs a graph with the entities as nodes and trains a model to cluster the matching nodes. Bansal, Blum, and Chawla [35] take advantage of a polynomial approximation algorithm to partition the graph, and find the clusters and the number of clusters in the dataset. Cohen and Richman [36] suggest a supervised clustering model which learns an adaptive distance function from a given set of training examples. Singla and Domingos [37] prove that by using attribute values as the nodes in the graph, you can propagate information across nodes, and this improves the performance of duplicate record detection.

The necessity for a large training set is a major problem for supervised ML models. To create a dataset for EM, it is straightforward to find a large number of training data that is clearly a match or non-match. However, it is very difficult to add ambiguous cases which enhance the model generalization to real-world situations. Some duplicate detection systems use active learning [38] to automatically identify such ambiguous

pairs. Unlike an “ordinary learner” that is trained by a static training set, an “active” learner actively selects subsets of unlabeled data which, when labeled, will give the model the most information gain.

Sarawagi and Bhamidipaty [39] introduce ALIAS, a duplicate detection model based on active learning, using the idea of “reject region” assigned to pairs with high uncertainty. The main idea behind ALIAS is that most duplicate and non-duplicate pairs are easy to distinguish. The model automatically categorizes such pairs into the U (Unmatched) and M (Matched) classes without the need for human manual labeling. However, ALIAS asks for manual labeling only when the uncertainty is high. The model can rapidly detect duplicates by leveraging only a small number of training data. Tejada, Knoblock, and Minton [40] implement Active Atlas, a record matching model, by a similar strategy using decision trees. They show that it is possible to identify ambiguous pairs by generating multiple classifiers that are trained using slightly different data or parameters.

2.4 Deep Learning Approaches

Deep Learning (DL) models and in particular Pretrained Language Models (PLMs) have revolutionized NLP and IR tasks including EM. Here deep learning models are divided into two sections: EM prior to PLM and PLM-based EM.

2.4.1 Entity Matching prior to Pre-trained Language Models

The first attempts and successes to apply DL to EM tasks are DeepER [41] and DeepMatcher [12]. DeepER utilizes an LSTM-based RNN with Siamese architecture [42], which contains two identical sub-networks with the same configuration, parameters, and weights. DeepER first tokenizes each entity, and then converts it to a vector representation using word embedding models, such as Glove [43] and fastText [44]. For each entity, the model generates a representation by aggregating token-level distributed representations. Given a pair of entity representations, they use a

dense fully-connected NN layer to calculate the similarity between the pair of entities, followed by an output layer to predict the matching (0/1).

DeepMatcher [12] formulates EM as a binary classification problem and experiments with different DL architectures from a simple one, similar to DeepER [41], to attention-based models. It is reported that the attention-based model outperforms other non-DL models. The attention-based networks allow DeepMatcher [12] to take into account the attribute similarity individually. Thus, DeepMatcher [12] is capable of capturing attribute-level similarity better than DeepER [41] because DeepER [41] only utilizes aggregate and coarser-grained entity representations to calculate the similarity between pairs of entities.

As one of the main limitations of DeepMatcher [12] and DeepER [41], both techniques treat an attribute as a sequence of tokens and use word embedding models; however, it is known that these models are not good at numerical comparisons. To address this issue, Fu, Han, Sun, Chen, Zhang, Wu, and Kong [45] introduce a multi-perspective matching model which considers multiple inputs individually (numeric, string, and text) and adaptively switches the similarity measures.

Another limitation of DeepMatcher [12] is that the attention mechanism only attends to the same attribute. It distracts the EM model when handling data from heterogeneous resources. To address this issue, Seq2SeqMatcher [46] extends the attention-based model of DeepMatcher [12] and lets it attend to other attributes. In other words, Seq2SeqMatcher [46] relaxes DeepMatcher’s [12] soft alignment condition (from the same attribute) to any attribute pairs. The more general attention model improves Seq2SeqMatcher [46] performance not only on heterogeneous but also on homogeneous ER-Magellan datasets [5]. To further enhance the performance on heterogeneous data, HierMatcher [47] designs three alignment layers for token-level, attribute-level, and entity-level similarity calculation.

In the era of DL, many EM models use RNN architectures and attention mechanisms. However, at a high level, how DL networks are designed differ in important

ways. Attribute comparator models apply cross attention only within the same attribute to make a matching decision, e.g., MPM [45], DeepMatcher [12], and Hi-EM [48] models. On the other hand, record comparator models use cross-attention at a record level considering all the attributes and the relation between them, e.g., Seq2SeqMatcher model [46].

Another important property of DL networks is the choice of independent or interdependent representation of entities. If a DL network sees a pair of entities to be compared, it generates interdependent representations of the entities, e.g., DeepMatcher [12], Hi-EM [48], and Seq2SeqMatcher [46] models. Otherwise, each entity is assigned a representation independent of other entities, e.g., DeepER [41] and AutoBlock [49] models.

2.4.2 Pre-trained-language-models-based EM

Recent high-performance models benefit from a fine-tuned PLM to tackle the EM problem. Transformer-based EM solutions employ interdependent representations, and they are categorized as record comparator models. DITTO [7], a prominent EM model, concatenates a pair of records to form a sequence, and fine-tunes a PLM to solve a sequence-pair classification problem. To improve its performance, DITTO has modules for Domain Knowledge (DK), data augmentation and summarization. About the summarization module, although the name implies a sophisticated neural network model, it just removes stop words in practice. In fact, it can be considered a pre-processing technique.

DITTO [7] serializes each data entry ($e = \{(attr_i, val_i)\}_{1 \leq i \leq k}$) as follows:

$serialize(e) ::= [COL]attr_1[VAL]val_1 \dots [COL]attr_k[VAL]val_k,$

where $[COL]$ and $[VAL]$ are special tokens for indicating the start of attribute names ($attr_i$) and attribute values (val_i), respectively. Brunner and Stockinger [50] introduce a similar transformer-based solution.

ROBEM [9] is inspired by DITTO, and it achieves promising results thanks to an

improved serialization technique, weighted cross-entropy loss function designed for imbalanced dataset, and a higher degree of non-linearity in the classification head. Structured data may be collected from heterogeneous resources and may lack attribute names. Thus, EM models are likely to become impaired when faced with circumstances where attribute name are not present, negatively affecting the model robustness. To alleviate this issue when serializing data entries, ROBEM does not consider attribute names and uses $[ATTR]$ as the special token between attribute values as follows:

$$serialize(e) ::= [ATTR]val_1\dots[ATTR]val_k,$$

Recently, SupCon [14] extends the idea of pre-trained transformers for EM using supervised contrastive learning [51], achieving SOTA results for Amazon-Google dataset. More recently, Narayan, Chami, Orr, and Ré [10] utilize LLMs such as **GPT3** [52] to push the SOTA results for EM benchmarks (e.g. Walmart-Amazon) in zero-shot and few-shot settings.

2.5 EM Models Leveraging Domain Knowledge

Although high-performance PLM-based EM models (e.g., DITTO [7], ROBEM [9], JointBERT [53]) reach near-perfect accuracy for most EM datasets (e.g., DBLP-Scholar and BeerAdvo-RateBeer [13]), these models are still lagging behind for product entity matching (PEM) datasets such as Amazon-Google [4] and Walmart-Amazon [11]. Some studies suggest using additional Domain Knowledge (DK) to improve EM model’s performance.

KAER [8] resorts to Wikidata as a knowledge graph to inject external DK at both schema and entity levels. However, this method is less likely to be effective for hard negative examples, as they typically belong to the same category at the schema level, and new and less popular products are less likely to be found in the knowledge graph. DITTO [7] highlights the importance of the existence of DK for EM and creates a

module for DK. However, what it does in practice is not consistent with what the name suggests and what the paper claims. The paper claims that it can inject DK into DITTO. However, in practice, (1) the authors use the Name Entity Recognition (NER) model of an off-the-shelf library (spaCy [16]) to assign tags to entities; (2) they rewrite text spans with developer-specified rules (e.g., replacing 5 % and 5.00 % with 5.0%). In fact, it is not domain-specific, they do the same for any domain, and the rewriting rules do not add any extra DK.

KAER [8] argues that external knowledge can help with the heterogeneity of data sources, but the reported results are only on the EM benchmarks that share the same schema. These recent studies have not focused on or addressed the aforementioned issues due to the lack of a relevant dataset.

2.6 Tabular Data as A Source of Knowledge

In the last section (§2.5), we discussed the EM solutions that resort to product Domain Knowledge (DK) to enhance EM model performance, although the suggested solutions weren't effective or product-specific. Product tables can be a reliable source of DK, and they are accessible from product pages on e-commerce platforms (e.g., Walmart, Amazon, BestBuy, etc.). Here we review studies that utilized tabular data as a source of DK. With the widespread use of electronic devices, tables have become the mainstream way to store facts, product characteristics, and enterprise data from various resources (e.g., webpages, databases, and spreadsheets), which represent the data as a grid-like format of rows and columns so that users can easily inquire about and discover patterns and insights from the data. Thus, tabular data is considered a rich source of knowledge and has been used for fact-based Question Answering (QA). The research on table QA has been increasing over the past few years, and the scholars have released different Text-to-SQL datasets such as Spider [54] and WikiSQL [55] and table QA datasets such as WTQ [56], SQA [57], and HiTab [58]. Recently, tabular and textual QA datasets have attracted the attention of the academic community,

and new datasets have been made available: TAT-QA [59], OTT-QA [60], HybridQA [61], FinQA [62], etc.

2.7 Gap in Research

Although Large Language Models (LLM) possess extensive language understanding capability which enables them to achieve near-perfect results for most entity matching datasets, the LLM-based models are still struggling with product entity matching (PEM) datasets (i.e., Amazon-Google [4] and Walmart-Amazon [11]). KAER [8] and DITTO [7] attempt to address this problem by providing additional Domain Knowledge (DK) information. However, their proposed solutions cannot help the model with PEM datasets and do not lead to new state-of-the-art results, when properly compared with strong baselines.

Tabular data has been proven to be an excellent source of DK for question answering models. However, it has not been applied for entity matching. Here we propose leveraging product-specific tabular data to improve the accuracy of entity matching models for PEM datasets.

Chapter 3

Product Entity Matching via Tabular Data

3.1 Introduction

In this chapter, we focus on improving Product Entity Matching (PEM) models by leveraging tabular data. Despite the importance of PEM for real-world e-commerce applications, the current prominent Pre-trained Language Model (PLM)-based Entity Matching (EM) models [7, 9, 10] are still lagging behind for these PEM datasets (§2.4.2, §2.5). As extensively discussed in Section 1.1, popular PEM datasets such as Amazon-Google [4] and Walmart-Amazon [11] suffer from intrinsic shortcomings: hard negative examples, lack of attribute identification, fixed attributes, and lack of domain knowledge. Table 3.1 shows three hard negative examples (highly similar titles but labeled as non-matching pairs) from the Amazon-Google dataset [4]. Certain parts of a title are more useful for reaching a matching decision (e.g., model, year introduced, functionalities, etc.), but those pieces of information may be located in different places for different products, and the model may not have direct access to the encoded attributes. For instance, in the first example in Table 3.1, the manufacturer “*mcafee*” is written at the beginning of the Amazon product title and in the middle of the Google one. An EM model should recognize if it is an important piece of information and the attribute it describes. In this chapter, we attempt to address these shortcomings of the PEM datasets by adding tabular data, a new serialization

Amazon Product Title	Google Product Title
“mcafee total protection 2007 3 users”	“mtp07emb3rua mcafee total protection 2007 complete package 3 users cd mini-box”
“britannica deluxe”	“britannica deluxe 2008”
“nero 7 ultra edition enhanced”	“70009 nero ultra edition enhanced v.7 complete package 1 user cd win”

Table 3.1: A few hard negative examples from Amazon-Google dataset [4]. Despite their highly similar titles, product pairs are not the same.

technique, and a new EM model.

We enrich PEM benchmarks from [12] to offer a product detail table as a source of additional knowledge for every Amazon product, serving as a bridge to connect two entities of interest and potentially improving the accuracy of matching decisions. The datasets include a varying number of attributes for each product, and the introduction of the detail table allows all characteristic features to be captured. The supplementary data is anticipated to reveal distinguishing features that may not be present in the product title, which can help the model to disambiguate hard negative examples. Consider the example shown in Figure 3.1, where a model number is given for the Google product but this model number is not mentioned in the title of the Amazon product. The presence of the model number in the detail table provides this missing link between the two product descriptions.

We also present **TA**ble & **TE**x for **E**ntity **M**atching (**TATEM**), an entity matching approach based on PLMs. TATEM reaches new SOTA results for PEM benchmarks by incorporating the complementary tabular data. We further introduce an **A**tribute **R**anking **M**odule (**ARM**) to rank the Amazon table attributes based on their relevance to Google products. Our evaluation shows that ARM is capable of finding the most effective attributes for EM.

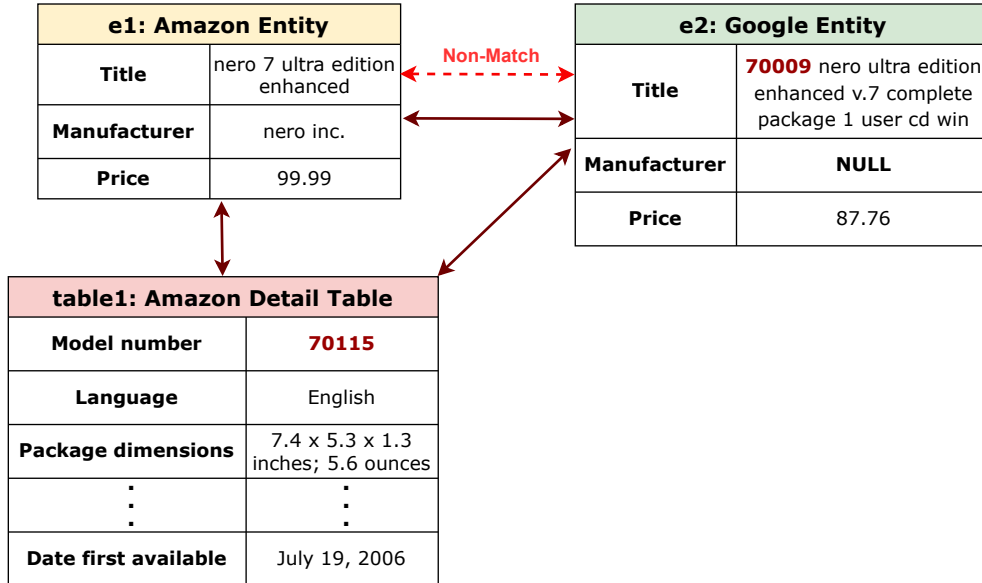


Figure 3.1: A hard negative example disambiguated using an Amazon product detail table, showing that relying on the information given in titles alone is hard to vote against a match because of the large number of overlapping tokens. Our model TATEM disambiguates this by establishing a relationship between *e2* and *table1* (*if exists*). Here, the **Model Number** field helps TATEM to reach a **Non-Match** decision.

Our contributions are summarized as follows: **(1)** We enrich popular and challenging PEM benchmarks [12] with complementary product tables. **(2)** We propose a new serialization technique to encode semi-structured tables in our PEM datasets for PLM-based models. **(3)** We develop TATEM, a model which employs both tabular and textual information for EM, reaching a new SOTA for challenging PEM benchmarks. **(4)** We design ARM to select important product-specific attributes and to make the model data-efficient.

3.2 Dataset

Many structured EM datasets have a fixed schema for their records, with a predetermined number of attributes for each sample. For instance, every product in the original Amazon-Google dataset introduced by Köpcke, Thor, and Rahm [4] has four attributes: title, manufacturer, price, and description. Based on the original Amazon-Google [4] and Walmart-Amazon datasets [11], Mudgal, Li, Rekatsinas, Doan, Park,

	Amazon-Google	Walmart-Amazon
	[4]	[11]
#Train samples (N./P.)	6175/699	5568/579
#Test samples (N./P.)	2059/234	1856/193
#Attrs (<i>fixed</i>)	3	5
	Amazon*-Google	Walmart-Amazon*
	(<i>ours</i>)	(<i>ours</i>)
#Tables (Amazon)	909	16264
Table coverage	66%	73%
#Unique attrs	84	695
Avg. #attrs	10.2	19.97
Max #attrs	28	81

Table 3.2: Stats. of Amazon-Google [4], Walmart-Amazon [11], Amazon*-Google (*ours*), and Walmart-Amazon* (*ours*).

Krishnan, Deep, Arcaute, and Raghavendra [12] released structured Amazon-Google and Walmart-Amazon datasets with only three and five attributes, respectively. Currently, high-performance EM models utilize the later version of the datasets. Our proposed enriched Amazon*-Google and Walmart-Amazon* datasets are based on the original datasets. To improve the effectiveness of EM models in disambiguating hard negative examples (see Figure 3.1) and to provide them with new challenges, we have added product-specific tables with varying numbers of attributes and many distinct schemas. Those detail tables are acquired in a two-step process: First, retrieving product pages using ASIN [63] of Amazon products, given in the original datasets; Second, extracting the relevant tabular data sections using the HTML structure of product webpages (e.g., tags, ids, class names, etc.).

Our enriched Amazon*-Google and Walmart-Amazon* datasets capture all the key features of Amazon products, such as model number, language, compatible OS, genre, and more. However, detail tables are product specific, and the schema and attributes

vary between products. Table 3.2 provides some statistics of the structured Amazon-Google [4], structured Walmart-Amazon [11], our Amazon*-Google and our Walmart-Amazon* datasets. The Amazon*-Google dataset provides product detail tables for 909 Amazon products through 84 unique attributes. Interestingly, our Walmart-Amazon* dataset includes more intricate product tables for a total of 16,264 Amazon items using 695 different unique attributes. Consequently, the EM task becomes challenging in **(1)** serialization of tables using PLMs (§3.3.1) and **(2)** data-efficient solutions for long tables (§3.3.2).

Our extended datasets (Amazon*-Google and Walmart*-Amazon) carry the same labels as the original datasets (Amazon-Google [4] and Walmart-Amazon [11]). The original Walmart-Amazon dataset [11] was created via hands-off crowd-sourcing, using only a crowd of ordinary workers (such as those on Amazon Mechanical Turk). The labeling was done in 3 stages: (1) blocking to reduce the sets of pairs, (2) selecting a limited number of rules for a matching decision by the crowd, and (3) evaluating the matching decision by the crowd using the product title and product page. For the original Amazon-Google dataset [4], two attributes (title and product description) were taken into account for each product. In order to recognize perfect match products, they utilized the UPC (Universal Product Code) [64] which allowed a unique identification of a product.

3.3 TATEM Model

We build our model TATEM atop ROBEM [9], a high-performance EM model that uses PLMs for recognizing matching product pairs. To this end, we modify the serialization technique to be compatible with the dataset structure, as discussed in the following section (§3.3.1), and we also develop ARM (§3.3.2) to make our model more data-efficient.

3.3.1 TATEM Serialization

High-performance PEM models utilize PLMs to generate a dense high-dimensional representations encoding structured data about product entities. However, these PLMs are designed to encode textual content. Thus, we need serialization techniques to convert this structured data into a proper format for the PLMs. DITTO [7] and ROBEM [9] utilize serialization techniques for structured datasets with a fixed number of attributes. However, TATEM especially designs and utilizes a serialization technique for semi-structured product-specific data. Although ROBEM and DITTO serialization techniques have the potential to be applied to semi-structured data, they are not as effective as TATEM serialization as it is shown in Table 3.3. Here TATEM serialization is explained for Amazon*-Google although the same procedure is applied to Walmart-Amazon*. For every example of Amazon*-Google dataset, there exist three fixed attributes: title, manufacturer, and price; and k' additional attributes from the product detail table, with k' varied for each product:

$$e = (title, val_{title}), (manufac, val_{manufac}), (price, val_{price}), \\ \{(attr_i, val_i)\}_{1 \leq i \leq k'}.$$

To serialize an entity e , for the first three attributes, only the attribute value is considered because they always appear in the same position, but for the other k' attributes, the attribute name is concatenated with the attribute value because the attributes are different for every product. Similar to ROBEM, a special token appears between the attributes:

$$serialize(e) ::= val_{title} [ATTR] val_{manufac} [ATTR] val_{price} [ATTR] \\ (attr_i, val_i) \dots [ATTR] (attr_{k'}, val_{k'}).$$

For example, the Amazon entity given in Figure 3.1 is serialized as: `nero 7 ultra edition enhanced [ATTR] nero inc. [ATTR] 99.99 [ATTR] model number`

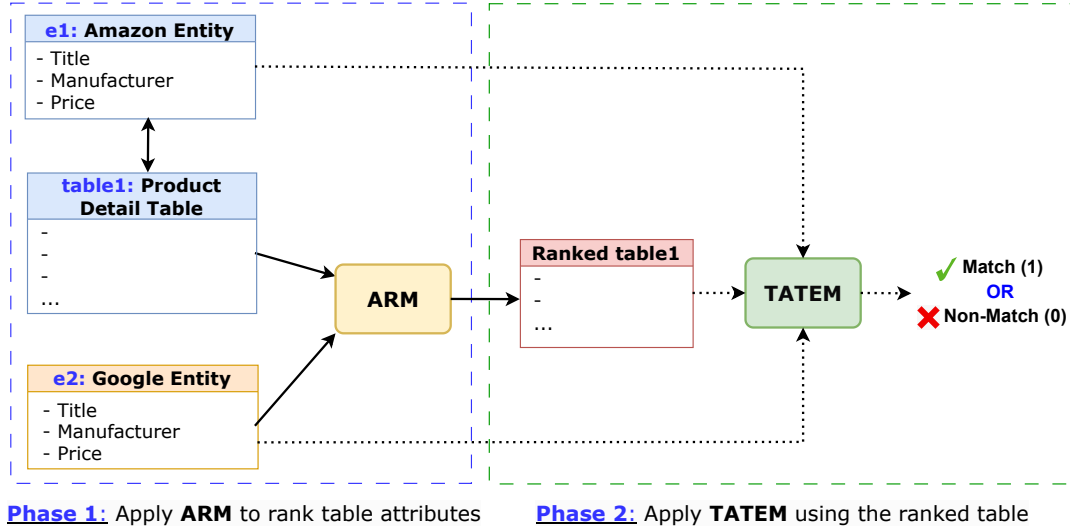


Figure 3.2: Our TATEM model coupled with ARM for PEM.

70115 [ATTR] ...[ATTR] date first available July 19, 2006.

3.3.2 Attribute Ranking Module (ARM)

In both of our datasets, each product entity is associated with a large number of attributes, making it challenging to determine which attributes are the most indicative of a true match or non-match. The goal of ARM is to generate the top n attribute-value pairs for a given product entity (e.g., from Amazon) in response to a pair of entities (e.g., Amazon-Google) for EM. This is important for two main reasons. Firstly, transformer-based PLMs [65, 66] have a limitation on the maximum length of input sequences they can handle [67]. The most common solution to this problem is to trim the input sequences to a certain length (e.g., 512). However, trimming a long input sequence is tricky for EM in general because important information for matching decisions may be located towards the bottom of the tables. Secondly, employing TATEM equipped with ARM can improve the overall efficiency and effectiveness of the EM task. For instance, reducing the number of input tokens can save computational resources, quicken the inference time, and save financial resources in particular when using pay-as-you-go services such as GPT PLMs [68].

Here ARM is described for Amazon*-Google although the same procedure is applied to Walmart-Amazon* dataset to select the most influential attributes. Inspired by unsupervised text ranking [69], ARM calculates the relevance of Amazon detail table attributes, $attr_i$, with respect to a Google product context, and it returns the top n attributes based on these estimates of relevance.

$$P(\text{Relevance} = 1 \mid attr_i, cntx) \triangleq \phi(\eta_{attr}(attr_i), \eta_{cntx}(cntx))$$

where ϕ is a comparison function and η_{cntx} and η_{attr} give encodings of a context from one source and a table attribute from another source, respectively. Our design choice for both encoders is Sentence-BERT (SBERT) [70], and we utilize cosine similarity as the comparison function and title as the Google product context. Important Amazon attributes for EM should contain information about features that are mentioned in a Google record, but not in an Amazon record. Consequently, an effective ARM should make matching records closer and non-matching records farther by adding just a few high-ranking attributes without a big drop in performance.

Figure 3.2 illustrates the operations of TATEM on an Amazon entity (t1) with 3 attributes (`title`, `manufacturer`, `price`) and a product detail table, compared to a Google product (t2). In phase 1, ARM outputs a ranked list of attribute-value pairs. Interestingly, the ranked table attributes are different for each pair of records. In phase 2, TATEM is applied to the Amazon-Google pair enriched with the ranked Amazon attributes and it outputs a matching decision (0, 1).

3.4 EXPERIMENTAL Evaluation

We implemented our models using PyTorch [71] and Huggingface transformers library [72]. We utilized RoBERTa_{base} [66] as our PLM because it has been found effective for the EM tasks [7, 73]. For training the models, we followed the parameter configuration of DITTO [7].

3.4.1 Results

Here, we aim to evaluate the importance of auxiliary tabular data about product key features, the effectiveness of TATEM serialization methodology, and the effect of ARM on the model’s data efficiency. As a base for comparison, Table 3.3 shows the performance, in terms of F1-score, of DeepMatcher+ (DM+), DITTO, ROBEM, KAER, SupCon, and GPT3 on structured Amazon-Google dataset [4]. To demonstrate how competitive models (DITTO, ROBEM, and SupCon) perform if they have access to our enriched Amazon*-Google dataset, we design two sets of experiments: **(1)** We convert our semi-structured Amazon*-Google and Walmart-Amazon* datasets into a structured format with a column allocated to each unique attribute. The results are denoted as Amazon*-Google and Walmart-Amazon* (*ours*) [structured] in Table 3.3. **(2)** We modify the implementation of DITTO and ROBEM to be compatible with our semi-structured data using their respective serializations described in Section 3.3.1. For the modified models, denoted as DITTO-m and ROBEM-m, we take the attribute name and attribute value directly from our Amazon*-Google and Walmart-Amazon* datasets to serialize each entry.

Figure 3.3 shows product detail tables for two sample products, containing different features of a laptop. It includes tables in the original semi-structured (a) and structured format (b). ROBEM [9] and DITTO [7] are designed for structured datasets with fixed schemas. To use our product detail tables for not modified ROBEM [9] and DITTO [7], we need to convert them into a structured format before serialization. We serialize Table1 for DITTO as:

```
serialize(Table1): [COL] Screen Type [VAL] IPS [COL] Screen Size [VAL]
15.6 [COL] Refresh Rate [VAL] 144Hz [COL] Processor Brand [VAL] ‘ ‘ [COL]
Processor Model [VAL] ‘ ‘ [COL] Processor Model Number [VAL] ‘ ‘ [COL]
Hard Drive [VAL] ‘ ‘ ,
```

Here we serialize Table1 for ROBEM in structured format as:

(a) Product Detail Tables

Table1	
Screen Type	IPS
Screen Size	15.6
Refresh Rate	144Hz

Table2	
Processor Brand	Intel
Processor Model	Intel 12th Generation Core i5
Processor Model Number	Intel 12th Generation Core i5-12500H
Hard Drive	512 SSD

(b) Product Detail Tables [Structured]

Table1	
Screen Type	IPS
Screen Size	15.6
Refresh Rate	144Hz
Processor Brand	
Processor Model	
Processor Model Number	
Hard Drive	

Table2	
Screen Type	
Screen Size	
Refresh Rate	
Processor Brand	Intel
Processor Model	Intel 12th Generation Core i5
Processor Model Number	Intel 12th Generation Core i5-12500H
Hard Drive	512 SSD

Figure 3.3: Detail tables for two products in semi-structured and structured formats.

```
serialize(Table1): [ATTR] IPS [ATTR] 15.6 [ATTR] 144Hz [ATTR] ‘ ‘ ’ [ATTR]
‘ ‘ ’ [ATTR] ‘ ‘ ’ [ATTR] ‘ ‘ ’ [ATTR],
```

DITTO-m and ROBEM-m models can directly serialize the semi-structured detail tables. We serialize Table1 for DITTO-m as:

```
serialize(Table1): [COL] Screen Type [VAL] IPS [COL] Screen Size [VAL]
15.6 [COL] Refresh Rate [VAL] 144Hz,
```

Similarly, we serialize Table1 for ROBEM-m as:

```
serialize(Table1): [ATTR] IPS [ATTR] 15.6 [ATTR] 144Hz [ATTR],
```

Obviously, DITTO-m and ROBEM-m are significantly more efficient for serializing our semi-structured detail tables and avoiding PLM confusion.

As reported in Table 3.3, once DITTO has access to the structured format of Amazon-Goole-Tab dataset, it outperforms the current SOTA model (SupCon) and presents a better performance than ROBEM, underscoring the importance of having additional product knowledge for EM. On the contrary, when DITTO-m and ROBEM-m are utilized for Amazon-Goole-Tab dataset, ROBEM-m outperforms DITTO-m. This disparity in performance is rooted in the serialization techniques employed by each model, highlighting the need for a serialization technique that is tailored to the structure of data. Adding complementary information to Walmart-Amazon

Models	F1 Score	
	Amazon-Google [4]	Walmart-Amazon [11]
DM+ (2018)	70.7	73.6
DITTO (2020)	75.58	86.76
KAER (2023)	76.52	-
ROBEM (2022)	79.06	86.68
SupCon (2022)	79.28	-
GPT3(k=0) (2022)	54.3	60.6
GPT3(k=10) (2022)	63.5	87.0
	Amazon*-Google (ours) [structured]	Walmart-Amazon* (ours) [structured]
DITTO	80.56	86.85
ROBEM	78.50	85.74
SupCon	78.58	-
	Amazon*-Google (ours)	Walmart-Amazon* (ours)
DITTO-m	79.35	86.42
ROBEM-m	80.92	88.31
TATEM (ours)		
+ w/ all tuples	82.2	90.56
+ w/ ARM (n=1)	80.12	88.52
+ w/ ARM (n=3)	81.28	89.24
+ w/ ARM (n=5)	81.83	89.77

Table 3.3: Performance of our proposed model **TATEM** compared to different baselines.

in a structured format doesn't help and even leads to a decrease in F1 score. This unintentional behavior is rooted in the difference between our two enriched datasets. In fact, Walmart-Amazon* includes far more unique features (695 compared to 84 unique features) as noted in Table 3.2, and this leads to wider, more sparse tables in a structured format, which confuses the PLM-based EM. In contrast, employing Walmart-Amazon* in a semi-structured format for DITTO-m and ROBEM-m significantly enhances the performance and outperforms the last SOTA results. It emphasizes the advantages of directly serializing semi-structured data, particularly for lengthy, complex product tables.

Our proposed model, TATEM, reaches new SOTA results (F1 score of 82.2 and 90.56 for Amazon*-Google and Walmart-Amazon*, respectively) as it benefits from a serialization technique that is specially designed for the product-specific tabular structure of our enriched datasets. Based on our findings, the best serialization for the three *fixed* attributes (i.e., title, manufacturer, price) is to exclude the attribute names. On the other hand, for k' varying attributes from the product detail table, including both the attribute name and value is the best strategy as it provides the EM model with information about the attribute type. The KAER model [8], despite having access to additional entity information from WikiData, fails to outperform ROBEM and reaches an F1 score of 76.25 on Amazon-Google dataset [4], demonstrating that accessing extra information does not essentially guarantee an increase in performance.

In our dataset, a typical entity can have up to 28 attributes. However, ARM can significantly reduce the number of tokens fed to a PLM. An effective ARM should identify the most important Amazon attributes for EM to make matching pairs closer and non-matching pairs further apart with just a few attributes. To evaluate the effect of ARM on EM results, ARM is applied to Amazon*-Google dataset, and from the ranked list, only the top n attributes are selected. Just adding the top one attribute can beat the current SOTA results (see Table 3.3), and the top five attributes are

responsible for most of the performance gain. Here it is demonstrated the effectiveness of ARM for improving the PEM model performance when only a few attributes are added. Additionally, as a baseline ARM can be compared with a dataset when n random table attributes are added.

3.5 Summary

In this paper, we introduced two new datasets, the Amazon*-Google and Walmart-Amazon* datasets, and developed a new solution called TATEM that uses Pre-trained Language Models (PLMs) with a novel serialization technique to encode semi-structured tables for Product Entity Matching (PEM). The experiments conducted on both existing benchmark datasets and the proposed datasets show significant improvements compared to current state-of-the-art methods. Additionally, we have designed an unsupervised attribute ranking module that enhances the model’s data-efficiency and cost-effectiveness for commercial services. For future work, we will test the robustness of TATEM model against distribution shift and input perturbation.

Chapter 4

TATTOO: Product Entity Matching as a Topology Construction

4.1 Introduction

In this chapter, our primary attention is directed toward enhancing Entity Matching (EM) model using tabular data. In Chapter 3, we proposed using product detail tables as a reliable source of product domain knowledge and introduced two new enriched datasets (Walmart-Amazon* and Amazon*-Google) in which Amazon detail tables were given for the Amazon items (refer to §3.2 for the extended analysis). However, in this chapter, we release a new enriched Walmart*-Amazon* as a Product Entity Matching (PEM) dataset in which there are product-specific tabular data for both Walmart and Amazon products.

As extensively discussed in Section 1.1 and exemplified in Figure 1.2 and Figure 3.1, popular PEM datasets such as Amazon-Google [4] and Walmart-Amazon [11] exhibit inherent limitations: hard negative examples, lack of attribute identification, fixed attributes, and lack of domain knowledge. To address these limitations in Chapter 3, we suggested leveraging tabular data (§3.2) as a crucial link between two entities, and our experimental results (§3.4) showed that this approach was successful in enhancing Pre-trained Language Models (PLM)-based EM models, and it reached new State-


Table1		Entity1		Entity2		Table2	
Screen Type	IPS	Title1		Title2		Screen Type	IPS
Screen Size	15.6	Acer - Nitro 5 15.6" Gaming Laptop FHD-Intel 12th Gen Core i5-AMD Ryzen 7 6800H 8-Core-NVIDIA GeForce RTX3050 Ti- 16GB DDR4-512GB SSD		Acer Nitro 5 15.6" FHD Gaming Laptop Black NVIDIA GeForce RTX 3050 Ti Intel Core i5 16GB DDR4 512GB Gen 4 SSD		Screen Size	15.6
Refresh Rate	144Hz	Manufacturer1		Manufacturer2		Refresh Rate	144Hz
Processor Brand	Intel	Acer		Acer		Processor Brand	Intel
Processor Model	Intel 12th Generation Core i5	Price1		Price2		Processor Model	Intel 12th Generation Core i5
Processor Model Number	Intel 12th Generation Core i5-12450H (8-core)	949.99		999.99		Processor Model Number	Intel 12th Generation Core i5-12500H
Hard Drive	512 SSD					Hard Drive	512 SSD
Graphic Coprocessor	NVIDIA GeForce RTX3050-Ti					Graphic Coprocessor	NVIDIA GeForce RTX3050-Ti
Model Number	AN515-58-57QW					Model Number	AN515-58-5046
Year of release	2023					Year of release	2022
System Memory (RAM)	16 GB					System Memory (RAM)	16 GB
Type of Memory	DDR4					Type of Memory	DDR4
Keyboard	Red Backlit					Keyboard	4-Zone RGB Backlit

Figure 4.1: A hard negative example disambiguated using product detail tables, indicating that depending only on the information provided in titles makes it challenging to confidently reject a match due to substantial missing key features and the presence of misleading and overlapping tokens. Our model TATTOO disambiguates this by establishing a relationship between entities and tables (*if exists*). Here, the highlighted fields help TATTOO to reach a **Non-Match** decision.

Of-The-Art (SOTA) results for both Amazon-Google and Walmart-Amazon datasets (refer to Table 3.3). In this Chapter, we extend this idea, release a new enriched PEM dataset (Walmart*-Amazon*) which provides detail tables for both entities.

Figure 4.1 shows an example in which you face the above shortcomings in the real world: Key product features are located in titles without any order; The titles are missing some key attributes; Different separators are utilized between features; Some misleading information is included in the titles to improve search visibility (e.g., *AMD Ryzon 7 6800 8-Core* in *Title1*); Only attribute values are included, and the PLM should figure out which attribute name they belong to. In these complicated examples, only product detail tables can help the EM model to disambiguate and correctly vote for a **Non-Match** decision. In Figure 4.1, the fields of the tables that contribute to the decision are highlighted.

Utilizing these tabular data for entity matching presents several challenges, which

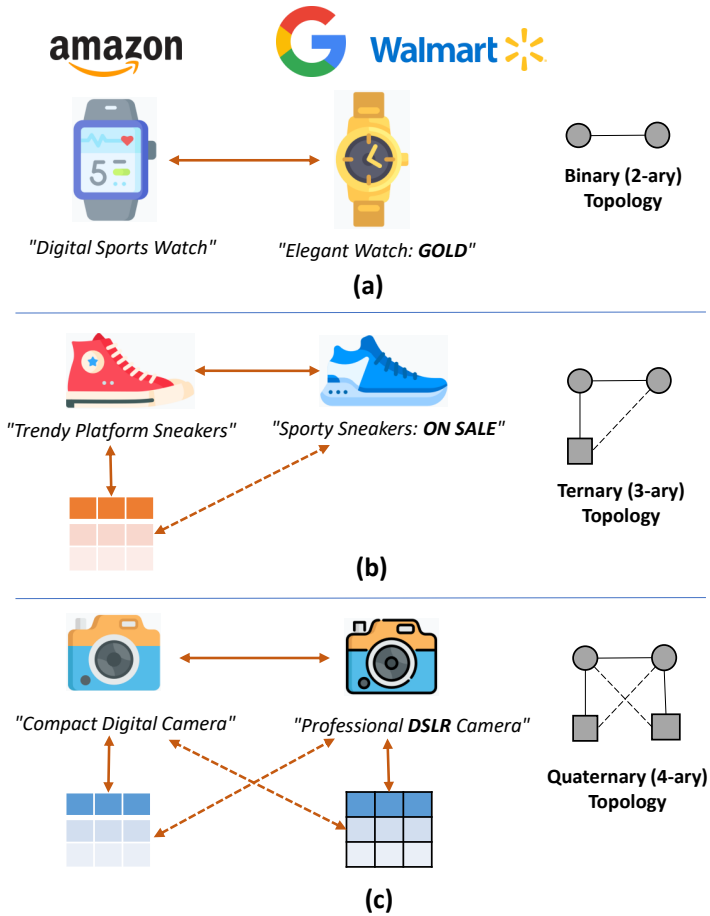


Figure 4.2: Topology Illustration.

we elaborate on below:

Binary (2-ary) Topology In most of the current entity matching models, the setup involves using title information to perform entity matching (Figure 4.2 (a)). However, product titles often pose challenges as they are often messy, SEO-optimized, incomplete, and sometimes encoded as a single string without proper separation or clear labeling of fields such as the product name, brand, price, etc. These complexities introduce significant challenges for existing product entity matching models, particularly when dealing with ambiguous negative examples (refer to the example in Figure 4.1).

Ternary (3-ary) Topology In the context of products that are new to the market or lack a sales history, product matching serves as a valuable tool to identify similar products. As the number of both new product releases and online retail vendors continue to grow, product records are becoming increasingly abundant and complex. However, the level of detail in these records can vary significantly among retailers, ranging from highly detailed information to more concise descriptions. These quantitative and qualitative differences in product information prompt us to explore various linkage techniques that are best suited for specific matching tasks. On one end of the spectrum, we encounter new or unpopular items, commonly referred to as “cold-start items.” On the other end, we have items with more comprehensive information and detailed records (refer to Figure 4.2 (b)).

Quaternary (4-ary) Topology In our latest setup, we dive into product entities that have a substantial amount of information, ranging from textual data (e.g., product titles) to semi-structured data (e.g., attribute-value pairs). This presents a challenge in determining the most crucial aspects of the information that signify a genuine match or non-match (refer to Figure 4.2 (c)). To address this challenge, we have developed techniques dedicated to retrieving essential attribute-value pairs. These techniques play a crucial role in refining the matching process and enhancing the accuracy of our entity matching system under some settings.

We approach PEM as a topology construction problem and propose a novel technique called **TATTOO** (**T**able & **T**ext Entity Matching as **T**opology **C**onstruction). Our contributions are summarized as follows:

- We enhance well-established PEM benchmarks [12] by incorporating complementary product detail tables for both items, which opens up new research avenues for PEM in general. In Chapter 3, we explained the effect of 3-ary datasets (Amazon*-Google and Walmart-Amazon*) on the performance of PLM-based EM models. In this chapter, we introduce a 4-ary PEM dataset with detail

tables for both entities.

- We develop TATTOO, a model which employs both tabular and textual information for EM, reaching a new SOTA for challenging PEM benchmarks. TATTOO is applied for 3-ary and 4-ary datasets.
- We devise Attribute Ranking Modules (ARM) to identify the most essential attributes of product-specific tables within a given topology. These modules facilitate an efficient utilization of commercial services. In this chapter, we implement two versions of this module for 4-ary datasets: Plain ARM and Cross ARM.
- To assess the robustness of TATTOO in few-shot training and out-of-domain generalization scenarios, we conduct comparative evaluation against popular baselines. The results reveal that our model exhibits greater robustness in the face of these changes.

4.2 Dataset

We explained our 3-ary enriched datasets (Amazon*-Google and Walmart-Amazon*) in Section 3.2. In this chapter, we aim to introduce our 4-ary enriched dataset (Walmart*-Amazon*). To enhance the ability of EM models, disambiguate hard negative examples, address the limitations of PEM datasets (§1.1) and provide them with new challenges, we present Walmart*-Amazon* dataset with detail tables for both Amazon and Walmart entities. Our proposed enriched Walmart*-Amazon* dataset is derived from the original Walmart-Amazon dataset. The product-specific detail tables are acquired following the same two-step process explained in Section 3.2. Although a large number of EM datasets are structured [4, 6, 11–13] with a pre-defined schema and a fixed number of attributes, our enriched PEM datasets are semi-structured with varying number of attributes for each product.

Original Dataset	Amazon-Google	Walmart-Amazon	
#Train samples (N./P.)	6175/699	5568/579	
#Test samples (N./P.)	2059/234	1856/193	
#Attrs (<i>fixed</i>)	3	5	
Enriched Dataset	Amazon*-Google	Walmart*-Amazon*	
	<i>(ours, 3-ary)</i>	<i>(ours, 4-ary)</i>	
		[Walmart]	[Amazon]
#Tables	909	1,863	16,264
Table coverage	66%	73%	73%
#Unique attrs	84	131	695
Avg. #attrs	10.2	12.31	19.97
Max #attrs	28	28	81

Table 4.1: Stats. of Amazon-Google [4], Walmart-Amazon [11], Amazon*-Google (*ours*) and Walmart*-Amazon*(*ours*). #Train & #Test denote the number of train and test samples for the datasets. #attrs denotes the number of attribute-value pairs for the tables. (N./P.) \Leftrightarrow (Num of Negative Pairs / Num of Positive Pairs).

Table 4.1 provides some statistics of the structured Amazon-Google [4], structured Walmart-Amazon [11], our enriched Amazon*-Google and Walmart*-Amazon* datasets. The Amazon*-Google dataset provides product detail tables for 909 Amazon products through 84 unique attributes. Interestingly, our Walmart*-Amazon* dataset includes more intricate tabular product data for a total of 16,264 tables with 695 different unique attributes and 1863 tables with 131 unique attributes for Amazon and Walmart items, respectively. Consequently, the EM task becomes more challenging in **(1)** serialization of tables using PLM-based models (§4.3) and **(2)** data-efficient solutions for long tables (§4.3.2).

4.3 TATTOO Model

4.3.1 TATTOO serialization

Both DITTO [7] and ROBEM [9] rely on serialization techniques for managing structured datasets featuring a fixed number of attributes. In contrast, TATTOO is specifically designed for handling semi-structured product-specific data with varying number of attributes. TATTOO serialization technique is adapted from our TATEM serialization method explained in Section 3.3.1.

4.3.2 Attribute Ranking Module (ARM)

It is difficult to identify which features of detail tables are the most indicative of a true match or non-match because each product entity is linked to a large number of attributes. In Section 3.3.2, we extensively studied the concept of ARM as an unsupervised module to identify the top n most decisive attributes for a matching decision. Our module in Chapter 3 was applied to 3-ary datasets and assumed Google product titles as the context to select the most informative attributes from Amazon detail tables of Amazon*-Google dataset. Here, in this chapter, we extend the idea of ARM for our 4-ary datasets.

Nowadays product characteristic features are encoded into product titles to enhance product discoverability (e.g., “Acer - Nitro 5 15.6" Gaming Laptop FHD - Intel 12th Gen Core i5 - NVIDIA GeForce RTX3050 Ti - 16GB DDR4 - 512GB SSD”). Therefore, here we utilize those informative titles to filter out table attributes. For matching entities A and B in our 4-ary setting when the model has access to detail tables, we design two variants of ARM: Plain ARM and Cross ARM. Figure 4.3 shows applying Plain ARM and Cross ARM to product detail tables to generate ranked key-value pairs for each table. This is done in Plain ARM by using the title of Entity A to select the most relevant attributes from the detail table of Entity A and returning a ranked list of those attributes. In practice, this filters out those at-

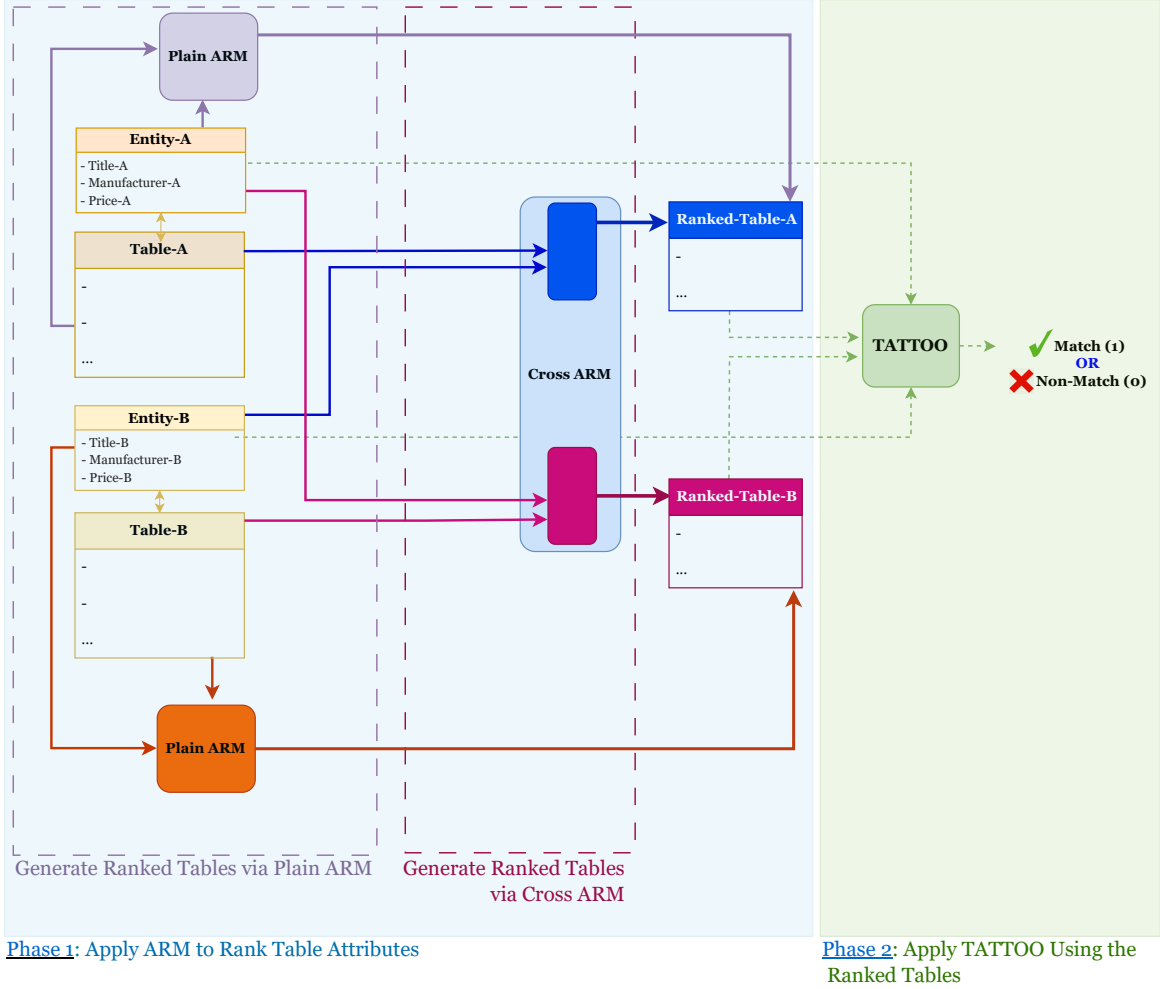


Figure 4.3: Our TATTOO equipped with Plain ARM and Cross ARM to select top n attributes.

tributes that are very different from the title of Entity A, and focuses more on the attributes similar to the features included in the title. In fact, Plain ARM makes product descriptions more complete independent of the entity on the other side that is being compared.

In contrast, Cross ARM looks at the other side to choose the top n attributes (refer to Figure 4.3). It extracts attributes from the detail table of Entity A based on their relevance to the title of Entity B. These extracted attributes may contain features not mentioned in the title of Entity A although there are corresponding features encoded in the title of Entity B. Thus, these top n attributes can play an important role in

identifying the differences and similarities between the records.

Figure 4.3 illustrates the operations of TATTOO on Entity-A with 3 attributes (`title`, `manufacturer`, `price`) accompanied by a product detail table (Table-A), compared to a Entity-B and Table-B. In Phase 1, ARM outputs ranked tables comprising ranked attribute-value pairs. Interestingly, the ranked table attributes are different for each pair of records. In Phase 2, TATTOO is applied to the pair of Entity-A and Entity-B enriched with the ranked tables, and it outputs a matching decision (i.e. either 0 or 1).

4.4 EXPERIMENTAL EVALUATION

We implemented our models using Huggingface transformers library [72]. We utilized RoBERTa_{base} [66] as our PLM because it has been found effective for the EM tasks [7, 73]. For training the models, we followed the parameter configuration of DITTO [7].

4.4.1 Accuracy of PEM Models

We examined the impact of our 3-ary enriched datasets on the performance of PEM tasks in Section 3.4. Here, we attempt to evaluate the effectiveness of our complementary tabular data in 4-ary settings for PEM tasks. As a base for comparison, Table 4.2 shows the performance, in terms of F1-score, of DeepMatcher+ (DM+), DITTO, ROBEM, KAER, SupCon, and GPT3 on structured PEM datasets [12]. To evaluate the performance of TATTOO against strong baseline models (DITTO, ROBEM, and SupCon) when they have access to our enriched datasets, we design two sets of experiments (the same design as the Experimental Section of Chapter 3 (§3.4)): **(1)** We convert our semi-structured enriched datasets (Amazon*-Google, Walmart-Amazon*) into a structured format with a column assigned to each unique feature. The results are denoted as Amazon*-Google (*ours*) [structured] and Walmart-Amazon* (*ours*) [structured] in Table 4.2. **(2)** We modify the implementa-

tion of DITTO and ROBEM to be compatible with our semi-structured data using their respective serializations (§4.3.1). For the modified models, denoted as DITTO-m and ROBEM-m, we use the attribute name and attribute value directly from our enriched datasets to serialize each entry.

As reported in Table 4.2, upon obtaining access to the structured format of the Amazon*-Google dataset, DITTO demonstrates superior performance compared to the current SOTA model (SupCon) and exhibits enhanced efficacy when compared to ROBEM. This highlights the significance of incorporating supplementary product domain knowledge into the EM process. In contrast, the utilization of DITTO-m and ROBEM-m on the Amazon*-Google dataset reveals superior performance by ROBEM-m over DITTO-m. This difference in performance stems from the serialization methods employed by each model, underscoring the necessity for a serialization approach tailored to the specific dataset structure.

Adding complementary tabular data to Walmart-Amazon in a structured format (Walmart-Amazon* (*ours*) [structured]) does not help and even leads to a decrease in F1 score. The difference between our two enriched datasets is the source of this unintended behavior. In fact, as seen in Table 4.1, Walmart*-Amazon has significantly more unique features than Amazon*-Google (695 vs. 84), resulting in broader, more sparse tables in a structured format, confusing the PLM-based EM. This finding underscores the significance of a serialization method devised specifically for semi-structured data. For DITTO-m and ROBEM-m, however, using Walmart-Amazon* in a semi-structured format avoids sparse structured representation, significantly boosts performance, and outperforms the last SOTA results. In, fact, ROBEM-m outperforms GPT3 (K=10) (latest SOTA) (88.81 vs. 87.00). This highlights the benefits of directly serializing semi-structured data, especially for lengthy, complex product tables.

Our proposed model, TATTOO, achieves new SOTA results (F1 score of 92.41 for Walmart*-Amazon*). The performance gain of TATTOO can be attributed to

two factors: (1) complementary product tables acting as effective product domain knowledge which help the model with hard negative examples; (2) the optimized serialization technique that is specifically designed for the structure of our enriched datasets. Based on our findings in this chapter and Chapter 3.4, the best serialization for the *fixed* attributes (e.g., title, manufacturer, price for Amazon*-Google) is to exclude the attribute names. On the other hand, for k' varying attributes from the product detail table, including both the attribute name and value is the best strategy as it provides the EM model with information about the attribute type. TATTOO achieves new SOTA with a large improvement (refer to Table 4.2), and we aim to compare TATTOO with DITTO [7] as a popular and strong baseline. Here, we use a paired t-test to find out if there is a significant difference between the two models.

Utilizing product domain knowledge to enhance EM accuracy has been referred to in the literature before (refer to Section 2.5 for more details). The KAER model [8] and DITTO [7] implementation, despite their claims, are not successful in empowering EM models with domain knowledge, and their solutions do not lead to an improvement. This study proves that product detail tables are a reliable source of domain knowledge, and they are specific for each product.

4.4.2 Effects of ARM Modules

A typical entity from the Walmart*-Amazon* and Amazon*-Google datasets can have up to 836 and 81 attributes (§4.2), respectively, using product detail tables as the source of domain knowledge for PEM. However, ARM can dramatically lower the number of tokens sent to a PLM. An efficient ARM for EM should determine the most essential Amazon properties that bring together matching pairs and push apart non-matching pairs with just a few attributes. ARM is applied on our enriched datasets in order to evaluate the impact of attribute ranking on PEM results. Only the top n attributes are then chosen from the ranked list. Table 4.3 summarizes the performance of our TATTOO model when equipped with ARM. Highlighting the effect of

Models	F1 Score		
	Amazon-Google (2-ary)	Walmart-Amazon (2-ary)	—
DM+ (2018)	70.70	73.60	—
DITTO (2020)	75.58	86.76	—
KAER (2023)	76.25	—	—
ROBEM (2022)	79.06	86.68	—
SUPCON (2022)	79.28	—	—
GPT3 ($k=0$) (2022)	54.30	60.60	—
GPT3 ($k=10$) (2022)	63.50	87.00	—
	Amazon*-Google (ours, 3-ary) [structured]	Walmart-Amazon* (ours, 3-ary) [structured]	Walmart*-Amazon* (ours, 4-ary) [structured]
DITTO	80.56	86.85	81.24
ROBEM	78.50	85.74	84.34
SUPCON	78.58	—	—
	Amazon*-Google (ours, 3-ary)	Walmart-Amazon* (ours, 3-ary)	Walmart*-Amazon* (ours, 4-ary)
DITTO-m	79.35	86.42	87.68
ROBEM-m	80.92	88.32	88.81
TATTOO (ours)	82.20	90.56	92.41

Table 4.2: Performance of **TATTOO** compared to different baselines. All reported results for **TATTOO** (*ours*) are statistically significant in paired t-test by taking DITTO (2020) as a reference with the confidence of 95% (p -value < 0.05).

complementary detail tables, only entities with tables are selected. Interestingly, just adding the top one attribute can beat the current SOTA results, and the top five attributes are responsible for most of the performance gain. Our additional experiments show adding more than 5 attributes leads to fluctuating results and decreases the performance for $n = 7$.

We design two variants of ARM: Plain ARM and Cross ARM (refer to Figure 4.3) in two modes: 3-ary and 4-ary topology (refer to Figure 4.2). Plain ARM finds top- n attributes of Amazon product detail tables by considering the title of the Amazon entities as the context. In contrast, Cross ARM identifies the top- n attributes of Amazon product detail tables by taking into account the title of Walmart entities. Unanimously, Cross ARM leads to higher performance because it is more powerful to identify the missing links between entities, helping our TATTOO model recognize similarities/differences between pairs of products.

Models	F1 Score	
	W-A (2-ary)	A-G (2-ary)
DITTO (2020)	84.41	73.24
ROBEM (2022)	83.32	78.25
	W-A* (<i>ours</i> , 3-ary)	A*-G (<i>ours</i> , 3-ary)
TATTOO (<i>ours</i>)		
+ w/ <i>all tuples</i>	92.51	86.27
+ w/ ARM ($n=1$)	90.17	80.34
+ w/ ARM ($n=3$)	91.36	81.21
+ w/ ARM ($n=5$)	92.11	82.12
	W*-A* (<i>ours</i> , 4-ary)	—
TATTOO (<i>ours</i>)		
+ w/ <i>all tuples</i>	93.70	—
+ w/ ARM ($n=1$)	90.76	—
+ w/ ARM ($n=3$)	92.05	—
+ w/ ARM ($n=5$)	92.56	—

Table 4.3: EM results of TATTOO equipped with ARM, considering top n product table attributes from our enriched datasets. Only examples with tables are considered. Best results in each setting are marked **bold**.

4.5 Evaluating Robustness

Since PEM data is typically sparse and noisy, it is challenging to train DL-based EM models using large, high-quality training data. Thus, there is a growing demand for robust models that excel in few-shot and out-of-domain settings.

4.5.1 Few-Shot Setting

We examine the data generalization of EM models to unseen test data from the same domain as the training data with limited access to the training dataset. For our in-domain generalization experiments, we compare TATTOO with two prominent PLM-based EM models: DITTO [7] and ROBEM [9] when they are trained only on a

portion (50%, 20%, 10%, 5%) of the training dataset and tested on the whole test set. Figure 4.4 and Figure 4.5 illustrate the results for the reduced training dataset size setting of Amazon*-Google and Walmart*-Amazon* datasets, respectively. Focused on the effect of product-specific tabular data, we only run the experiments on products with tables. This boosts the performance and increases F1 score from 92.41 to 93.7 and from 82.2 to 86.27 for Walmart*-Amazon* and Amazon*-Google, respectively. As we expect, F1 score of EM models drops when they are trained on a smaller dataset set. TATTOO exhibits incredibly robust behavior for the Amazon*-Google dataset. The F1 score of TATTOO is 95.15% larger than that of DITTO (66.84 vs. 34.25), provided only 5% of the training data. The TATTOO model trained on 50% of training data still outperforms the current SOTA model for Amazon*-Google.

Similarly, TATTOO trained on Walmart*-Amazon* reveals to be significantly more robust compared to the baselines. TATTOO trained on 5% of the training set achieves an F1 score of 76.43 which is 118.12% (+41.39) better than DITTO. As a result, the TATTOO model which employs product detail tables as the source of product domain knowledge is significantly more robust to training set size.

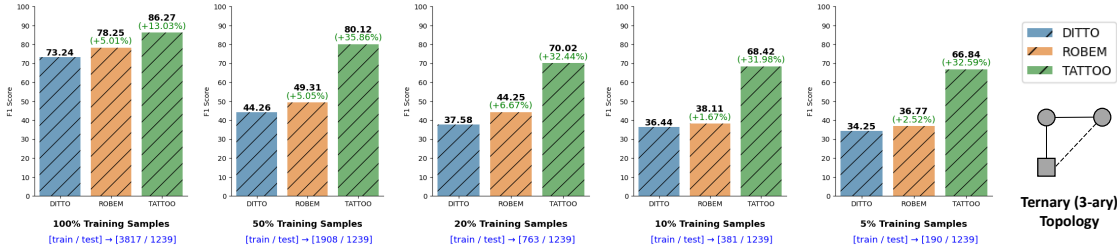


Figure 4.4: Few-Shot Setting Amazon*-Google.

4.5.2 Out-of-domain Generalization

We set up Out-Of-Domain (OOD) experiments between DITTO [7], ROBEM [9], GPT3 [10], and our TATTOO in which a fine-tuned model is confronted with an OOD dataset at test time. For example, we fine-tune a model on Walmart-Amazon* dataset, then repurpose the model for Amazon*-Google dataset ($W - A^* \Rightarrow A^* - G$)

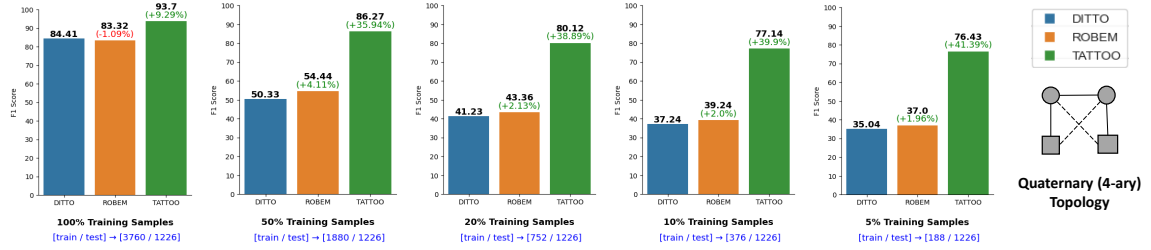


Figure 4.5: Few-Shot Setting performance of TATTOO (*ours*) compared to DITTO [7] and ROBEM [9] on our enriched Walmart*-Amazon* [Quaternary (4-ary) Topology]. Bold-faced numbers on top of the bars indicate the F1 Score, while green and red colored numbers represent the percentage point increase or decrease, respectively, compared to the baseline DITTO [7].

at zero-shot setting. Table 4.4 summarizes the performance of the baselines and our TATTOO model in two different domain shift settings. Then, to evaluate the drop in performance in a domain-shift setting, a comparison is made between in-domain and OOD results, summarized in Table 4.4. In $(W - A^* \Rightarrow A^* - G)$ setting, our TATTOO model displays far more robust OOD behavior (F1=66.12) than other baselines, it even beats GPT3(K=0) and GPT3(K=10) on Amazon-Google dataset. In fact, the drop in performance of TATTOO is way smaller than DITTO (16.08% vs. 46.04%). The difference between TATTOO and the baselines becomes more pronounced in OOD, and TATTOO achieves 31.60% higher F1 compared to DITTO. The promising OOD behavior of TATTOO is rooted in its serialization technique in which our model faces a varying number of distinct features for every product during training. Thus, TATTOO is inherently trained to generalize to products with different features, and it enables the model to be adaptive to a new domain with unseen attributes. In contrast, ROBEM and DITTO using structured datasets encounter the same features for all the samples during training, and they don't expect to see a new feature. Thus, testing the model on a test set with a different set of unseen attributes drops the performance.

Similarly for the $(A^* - G \Rightarrow W - A^*)$ setting, TATTOO is leading in OOD tests by F1 of 60.27 in comparison with DITTO and ROBEM (32.21 and 34.67, respectively). This OOD robust performance is accomplished with a considerably smaller drop in

performance when compared with DITTO (30.29% vs. 54.64%) (see Table 4.4).

Amazon*-Google (<i>ours</i> , 3-ary)			
Models	F1 Score		Δ
	In-domain (A*-G \Rightarrow A*-G)	Out-of-domain (W-A* \Rightarrow A*-G)	
DITTO	80.56	34.52 (\downarrow 46.04%)	[\uparrow 0.00%]
ROBEM	78.50	36.76 (\downarrow 41.74%)	[\uparrow 2.24%]
GPT3 (k=0)	—	54.30 (\downarrow 00.00%)	[\uparrow 19.78%]
TATTOO (ours)	82.20	66.12 (\downarrow 16.08%)	[\uparrow 31.60%]
Walmart-Amazon* (<i>ours</i> , 3-ary)			
Models	F1 Score		Δ
	In-domain (W-A* \Rightarrow W-A*)	Out-of-domain (A*-G \Rightarrow W-A*)	
DITTO	86.85	32.21 (\downarrow 54.64%)	[\uparrow 0.00%]
ROBEM	85.74	34.67 (\downarrow 51.07%)	[\uparrow 2.46%]
GPT3 (k=0)	—	60.60 (\downarrow 00.00%)	[\uparrow 28.06%]
TATTOO (ours)	90.56	60.27 (\downarrow 30.29%)	[\uparrow 27.36%]

Table 4.4: Out-of-domain results when a PLM-based EM model is fine-tuned on A and tested on B ($A \Rightarrow B$). Drop in performance compared to in-domain settings are presented in round brackets. Δ is the % change compared to baseline DITTO. GPT3 (**k=0**) in zero-shot setting is considered as out-of-domain. All training examples are considered. Best results in each setting are marked **bold**.

4.6 Summary

We have introduced a new enriched dataset, Walmart*-Amazon*, with unique product detail tables for the task of product entity matching (PEM), and a new effective PEM solution that uses Pre-trained Language Models (PLM) with a novel serialization technique to encode semi-structured product-specific tables. The experiments performed on the proposed datasets and the existing benchmarks demonstrate significant enhancements over SOTA techniques. To enhance the model’s data-efficiency

and cost-effectiveness, we devised an unsupervised Attribute Ranking Module (ARM) in two versions (Plain ARM and Cross ARM) to select the top n most informative product attributes. Our TATTOO model equipped with ARM can outperform the SOTA models even with one attribute for both Amazon-Google and Walmart-Amazon datasets.

TATTOO displays much more robust in-domain generalization behavior when it has access only to a small portion of the training set (5%-50%). Trained on 5% of Walmart*-Amazon* dataset, TATTOO is leading with 39.9% improvement over DITTO (77.14 vs 37.24). Additionally, our model demonstrates notable out-of-domain robustness. In $(W - A^* \Rightarrow A^* - G)$ setting, TATTOO can outperform GPT3(K=0) and GPT3(K=10) on Amazon-Google dataset, and our model achieves 31.60% enhancement compared with DITTO.

Chapter 5

Conclusions, & Future Work

5.1 Conclusions

Our goal is to enhance Entity Matching (EM) approaches that are based on Pre-trained Language Models (PLM) with product domain knowledge. To achieve this, we utilize product detail tables extracted from product web pages as a reliable source of domain knowledge to enrich two Product Entity Matching (PEM) datasets: Amazon-Google and Walmart-Amazon.

We introduce our enriched Amazon*-Google and Walmart-Amazon* datasets containing Amazon product detail tables, and we develop TATEM, a high-performance Table & Text for Entity Matching model using PLM and tabular data, to effectively serialize our semi-structured enriched datasets. Our experimental results on both the existing benchmark datasets and the newly created datasets indicate considerable enhancements in comparison to the current State-Of-The-Art (SOTA) techniques. TATEM improves the SOTA results from 87.0 to 90.56 and from 79.28 to 82.2 for Walmart-Amazon and Amazon-Google datasets, respectively. Additionally, we design an unsupervised Attribute Ranking Module (ARM) to identify the most decisive table attributes. Adding even top *one* attribute outperforms the current SOTA model for both Amazon-Google and Walmart-Amazon datasets.

We create Walmart*-Amazon* dataset benefiting from product-specific detail tables for both Walmart and Amazon items. Our enriched Walmart*-Amazon* includes

779 unique attributes and tables with up to 81 attributes. To take advantage of the product detail tables, we especially devise TATTOO, TABLE & Text Entity Matching as TOPOLOGY CONSTRUCTION). TATTOO beats current SOTA models and achieves an F1 score of 92.41 for Walmart*-Amazon* by leveraging PLMs, effective serialization techniques, and product detail tables. Here we develop two versions of ARM: Cross ARM and Plain ARM. We evaluate the robustness of TATTOO when faced low-resource scenarios and trained only on a portion of the training set. Trained on 5% of Walmart*-Amazon* dataset, TATTOO displays 41.39% improvement over DITTO (76.43 vs. 35.04). Additionally, our TATTOO shows a robust out-of-domain generalization behavior. For example, in $(W - A^* \Rightarrow A^* - G)$ setting, TATTOO outperforms both zero-shot GPT3 and few-shot GPT3(K=10) on Amazon-Google dataset, achieving a remarkable 31.60% enhancement compared to DITTO.

5.2 Future Work

In this thesis, we built unsupervised Attribute Ranking Module (ARM) modules to recognize the top n most decisive attributes, although we observed inconsistent behavior when adding more than 5 attributes. Designing a supervised ARM module coupled with the PEM model trained on our enriched dataset based on one loss function is a possible future direction. This loss function helps us fine-tune the PLM, classification head of PEM model, and the ARM module. Additionally, ARM is capable of detecting product attributes that lead to a match or non-match decision. Thus, ARM may be repurposed to add a layer of explainability to our PEM model. Exploring this is a possible future direction.

We plan to investigate the performance of TATTOO equipped with ARM when tested on recent products with longer detail tables which is more challenging. Also, we plan to examine EM model robustness against unit perturbation by comparing enriched/traditional EM datasets. Lastly, we plan to identify the most important piece of information for EM decisions using explainable AI tools.

Bibliography

- [1] Ivan P. Fellegi and Alan B. Sunter, “A theory for record linkage,” *Journal of the American Statistical Association*, vol. 64, no. 328, pp. 1183–1210, 1969. DOI: 10.1080/01621459.1969.10501049. [Online]. Available: <https://www.tandfonline.com/doi/abs/10.1080/01621459.1969.10501049>.
- [2] Ralph Peeters, Reng Chiz Der, and Christian Bizer, *Wdc products: A multi-dimensional entity matching benchmark*, 2023. DOI: 10.48550/ARXIV.2301.09521. [Online]. Available: <https://arxiv.org/abs/2301.09521>.
- [3] Yuliang Li, Jinfeng Li, Yoshihiko Suhara, Jin Wang, Wataru Hirota, and Wang-Chiew Tan, “Deep entity matching: Challenges and opportunities,” *J. Data and Information Quality*, vol. 13, no. 1, 2021, ISSN: 1936-1955. DOI: 10.1145/3431816. [Online]. Available: <https://doi.org/10.1145/3431816>.
- [4] Hanna Köpcke, Andreas Thor, and Erhard Rahm, “Evaluation of entity resolution approaches on real-world match problems,” *Proc. VLDB Endow.*, vol. 3, no. 1–2, 484–493, 2010, ISSN: 2150-8097. DOI: 10.14778/1920841.1920904. [Online]. Available: <https://doi.org/10.14778/1920841.1920904>.
- [5] Pradap Konda, Sanjib Das, Paul Suganthan G. C., AnHai Doan, Adel Ardalan, Jeffrey R. Ballard, Han Li, Fatemah Panahi, Haojun Zhang, Jeff Naughton, Shishir Prasad, Ganesh Krishnan, Rohit Deep, and Vijay Raghavendra, “Magellan: Toward building entity matching management systems over data science stacks,” *Proc. VLDB Endow.*, vol. 9, no. 13, 1581–1584, 2016, ISSN: 2150-8097. DOI: 10.14778/3007263.3007314. [Online]. Available: <https://doi.org/10.14778/3007263.3007314>.
- [6] Anna Primpeli, Ralph Peeters, and Christian Bizer, “The wdc training dataset and gold standard for large-scale product matching,” in *Companion Proceedings of The 2019 World Wide Web Conference*, ser. WWW ’19, San Francisco, USA: Association for Computing Machinery, 2019, 381–386, ISBN: 9781450366755. DOI: 10.1145/3308560.3316609. [Online]. Available: <https://doi.org/10.1145/3308560.3316609>.
- [7] Yuliang Li, Jinfeng Li, Yoshihiko Suhara, AnHai Doan, and Wang-Chiew Tan, “Deep entity matching with pre-trained language models,” *Proc. VLDB Endow.*, vol. 14, no. 1, 50–60, 2020, ISSN: 2150-8097. DOI: 10.14778/3421424.3421431. [Online]. Available: <https://doi.org/10.14778/3421424.3421431>.

- [8] Liri Fang, Lan Li, Yiren Liu, Vetle I. Torvik, and Bertram Ludäscher, *Kaer: A knowledge augmented pre-trained language model for entity resolution*, 2023. DOI: 10.48550/ARXIV.2301.04770. [Online]. Available: <https://arxiv.org/abs/2301.04770>.
- [9] Mehdi Akbarian Rastaghi, Ehsan Kamaloo, and Davood Rafiei, “Probing the robustness of pre-trained language models for entity matching,” in *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, ser. CIKM ’22, Atlanta, GA, USA: Association for Computing Machinery, 2022, 3786–3790, ISBN: 9781450392365. DOI: 10.1145/3511808.3557673. [Online]. Available: <https://doi.org/10.1145/3511808.3557673>.
- [10] Avanika Narayan, Ines Chami, Laurel Orr, and Christopher Ré, “Can foundation models wrangle your data?” *Proc. VLDB Endow.*, vol. 16, no. 4, 738–746, 2022, ISSN: 2150-8097. DOI: 10.14778/3574245.3574258. [Online]. Available: <https://doi.org/10.14778/3574245.3574258>.
- [11] Chaitanya Gokhale, Sanjib Das, AnHai Doan, Jeffrey F. Naughton, Narasimhan Rampalli, Jude Shavlik, and Xiaojin Zhu, “Corleone: Hands-off crowdsourcing for entity matching,” in *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD ’14, Snowbird, Utah, USA: Association for Computing Machinery, 2014, 601–612, ISBN: 9781450323765. DOI: 10.1145/2588555.2588576. [Online]. Available: <https://doi.org/10.1145/2588555.2588576>.
- [12] Sidharth Mudgal, Han Li, Theodoros Rekatsinas, AnHai Doan, Youngchoon Park, Ganesh Krishnan, Rohit Deep, Esteban Arcaute, and Vijay Raghavendra, “Deep learning for entity matching: A design space exploration,” in *Proceedings of the 2018 International Conference on Management of Data*, ser. SIGMOD ’18, Houston, TX, USA: Association for Computing Machinery, 2018, 19–34, ISBN: 9781450347037. DOI: 10.1145/3183713.3196926. [Online]. Available: <https://doi.org/10.1145/3183713.3196926>.
- [13] Sanjib Das, AnHai Doan, Paul Suganthan G. C., Chaitanya Gokhale, Pradap Konda, Yash Govind, and Derek Paulsen, *The magellan data repository*, <https://sites.google.com/site/anhaidgroup/projects/data>.
- [14] Ralph Peeters and Christian Bizer, “Supervised contrastive learning for product matching,” in *Companion Proceedings of the Web Conference 2022*, ser. WWW ’22, Virtual Event, Lyon, France: Association for Computing Machinery, 2022, 248–251, ISBN: 9781450391306. DOI: 10.1145/3487553.3524254. [Online]. Available: <https://doi.org/10.1145/3487553.3524254>.
- [15] Pengfei Wang, Xiaocan Zeng, Lu Chen, Fan Ye, Yuren Mao, Junhao Zhu, and Yunjun Gao, “Promptem: Prompt-tuning for low-resource generalized entity matching,” *Proc. VLDB Endow.*, vol. 16, no. 2, 369–378, 2022, ISSN: 2150-8097. DOI: 10.14778/3565816.3565836. [Online]. Available: <https://doi.org/10.14778/3565816.3565836>.

- [16] spaCy, *Https://spacy.io/api/entityrecognizer*. [Online]. Available: <https://spacy.io/api/entityrecognizer>.
- [17] Howard B Newcombe, James M Kennedy, SJ Axford, and Allison P James, “Automatic linkage of vital records: Computers can be used to extract” follow-up” statistics of families from files of routine records,” *Science*, vol. 130, no. 3381, pp. 954–959, 1959.
- [18] Ivan P. Fellegi and Alan B. Sunter, “A theory for record linkage,” *Journal of the American Statistical Association*, vol. 64, no. 328, pp. 1183–1210, 1969. DOI: 10.1080/01621459.1969.10501049. eprint: <https://www.tandfonline.com/doi/pdf/10.1080/01621459.1969.10501049>. [Online]. Available: <https://www.tandfonline.com/doi/abs/10.1080/01621459.1969.10501049>.
- [19] Matthew A. Jaro, “Advances in record-linkage methodology as applied to matching the 1985 census of tampa, florida,” *Journal of the American Statistical Association*, vol. 84, no. 406, pp. 414–420, 1989. DOI: 10.1080/01621459.1989.10478785. eprint: <https://www.tandfonline.com/doi/pdf/10.1080/01621459.1989.10478785>. [Online]. Available: <https://www.tandfonline.com/doi/abs/10.1080/01621459.1989.10478785>.
- [20] A. P. Dempster, N. M. Laird, and D. B. Rubin, “Maximum likelihood from incomplete data via the em algorithm,” *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 39, no. 1, pp. 1–22, 1977. DOI: <https://doi.org/10.1111/j.2517-6161.1977.tb01600.x>. eprint: <https://rss.onlinelibrary.wiley.com/doi/pdf/10.1111/j.2517-6161.1977.tb01600.x>. [Online]. Available: <https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/j.2517-6161.1977.tb01600.x>.
- [21] William E Winkler, *Improved decision rules in the fellegi-sunter model of record linkage*. Bureau of the Census Washington, DC, 1993, vol. 56.
- [22] Benjamin J. Tepping, “A model for optimum linkage of records,” *Journal of the American Statistical Association*, vol. 63, no. 324, pp. 1321–1332, 1968. DOI: 10.1080/01621459.1968.10480930. eprint: <https://www.tandfonline.com/doi/pdf/10.1080/01621459.1968.10480930>. [Online]. Available: <https://www.tandfonline.com/doi/abs/10.1080/01621459.1968.10480930>.
- [23] Vassilios S Verykios, George V Moustakides, and Mohamed G Elfeky, “A bayesian decision model for cost optimal record matching,” *The VLDB Journal*, vol. 12, pp. 28–40, 2003.
- [24] Wenfei Fan, Xibei Jia, Jianzhong Li, and Shuai Ma, “Reasoning about record matching rules,” *Proc. VLDB Endow.*, vol. 2, no. 1, 407–418, 2009, ISSN: 2150-8097. DOI: 10.14778/1687627.1687674. [Online]. Available: <https://doi.org/10.14778/1687627.1687674>.
- [25] Rohit Singh, Vamsi Meduri, Ahmed Elmagarmid, Samuel Madden, Paolo Papotti, Jorge-Arnulfo Quiané-Ruiz, Armando Solar-Lezama, and Tang, “Generating concise entity matching rules,” in *Proceedings of the 2017 ACM International Conference on Management of Data*, ser. SIGMOD ’17, Chicago, Illinois, USA: Association for Computing Machinery, 2017, 1635–1638, ISBN:

9781450341974. DOI: 10.1145/3035918.3058739. [Online]. Available: <https://doi.org/10.1145/3035918.3058739>.
- [26] Surajit Chaudhuri, Bee-Chung Chen, Venkatesh Ganti, and Raghav Kaushik, “Example-driven design of efficient record matching queries,” in *Proceedings of the 33rd International Conference on Very Large Data Bases*, ser. VLDB ’07, Vienna, Austria: VLDB Endowment, 2007, 327–338, ISBN: 9781595936493.
- [27] Wenfei Fan, Xibei Jia, Jianzhong Li, and Shuai Ma, “Reasoning about record matching rules,” *Proc. VLDB Endow.*, vol. 2, no. 1, 407–418, 2009, ISSN: 2150-8097. DOI: 10.14778/1687627.1687674. [Online]. Available: <https://doi.org/10.14778/1687627.1687674>.
- [28] Rohit Singh, Venkata Vamsikrishna Meduri, Ahmed Elmagarmid, Samuel Madden, Paolo Papotti, Jorge-Arnulfo Quiané-Ruiz, Armando Solar-Lezama, and Nan Tang, “Synthesizing entity matching rules by examples,” *Proc. VLDB Endow.*, vol. 11, no. 2, 189–202, 2017, ISSN: 2150-8097. DOI: 10.14778/3149193.3149199. [Online]. Available: <https://doi.org/10.14778/3149193.3149199>.
- [29] Jiannan Wang, Guoliang Li, Jeffrey Xu Yu, and Jianhua Feng, “Entity matching: How similar is similar,” *Proc. VLDB Endow.*, vol. 4, no. 10, 622–633, 2011, ISSN: 2150-8097. DOI: 10.14778/2021017.2021020. [Online]. Available: <https://doi.org/10.14778/2021017.2021020>.
- [30] Matteo Paganelli, Paolo Sottovia, Francesco Guerra, and Yannis Velegrakis, “Tuner: Fine tuning of rule-based entity matchers,” in *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, ser. CIKM ’19, Beijing, China: Association for Computing Machinery, 2019, 2945–2948, ISBN: 9781450369763. DOI: 10.1145/3357384.3357854. [Online]. Available: <https://doi.org/10.1145/3357384.3357854>.
- [31] Munir Cochinwala, Verghese Kurien, Gail Lalk, and Dennis Shasha, “Efficient data reconciliation,” *Information Sciences*, vol. 137, no. 1, pp. 1–15, 2001, ISSN: 0020-0255. DOI: [https://doi.org/10.1016/S0020-0255\(00\)00070-0](https://doi.org/10.1016/S0020-0255(00)00070-0). [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0020025500000700>.
- [32] Wei-Yin Loh, “Classification and regression trees,” *WIREs Data Mining and Knowledge Discovery*, vol. 1, no. 1, pp. 14–23, 2011. DOI: <https://doi.org/10.1002/widm.8>. eprint: <https://wires.onlinelibrary.wiley.com/doi/pdf/10.1002/widm.8>. [Online]. Available: <https://wires.onlinelibrary.wiley.com/doi/abs/10.1002/widm.8>.
- [33] M. Bilenko, R. Mooney, W. Cohen, P. Ravikumar, and S. Fienberg, “Adaptive name matching in information integration,” *IEEE Intelligent Systems*, vol. 18, no. 5, pp. 16–23, 2003. DOI: 10.1109/MIS.2003.1234765.
- [34] Thorsten Joachims, “Making large-scale svm learning practical,” Technical report, Tech. Rep., 1998.
- [35] Nikhil Bansal, Avrim Blum, and Shuchi Chawla, “Correlation clustering,” *Machine learning*, vol. 56, pp. 89–113, 2004.

- [36] William W. Cohen and Jacob Richman, “Learning to match and cluster large high-dimensional data sets for data integration,” in *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD ’02, Edmonton, Alberta, Canada: Association for Computing Machinery, 2002, 475–480, ISBN: 158113567X. DOI: 10.1145/775047.775116. [Online]. Available: <https://doi.org/10.1145/775047.775116>.
- [37] Parag Singla and Pedro Domingos, “Multi-relational record linkage,” in *Proc. KDD-2004 Workshop Multi-Relational Data Mining*, 2004, pp. 31–48.
- [38] David Cohn, Les Atlas, and Richard Ladner, “Improving generalization with active learning,” *Machine learning*, vol. 15, pp. 201–221, 1994.
- [39] Sunita Sarawagi and Anuradha Bhamidipaty, “Interactive deduplication using active learning,” in *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD ’02, Edmonton, Alberta, Canada: Association for Computing Machinery, 2002, 269–278, ISBN: 158113567X. DOI: 10.1145/775047.775087. [Online]. Available: <https://doi.org/10.1145/775047.775087>.
- [40] Sheila Tejada, Craig A. Knoblock, and Steven Minton, “Learning domain-independent string transformation weights for high accuracy object identification,” in *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD ’02, Edmonton, Alberta, Canada: Association for Computing Machinery, 2002, 350–359, ISBN: 158113567X. DOI: 10.1145/775047.775099. [Online]. Available: <https://doi.org/10.1145/775047.775099>.
- [41] Muhammad Ebraheem, Saravanan Thirumuruganathan, Shafiq Joty, Mourad Ouzzani, and Nan Tang, “Distributed representations of tuples for entity resolution,” *Proc. VLDB Endow.*, vol. 11, no. 11, 1454–1467, 2018, ISSN: 2150-8097. DOI: 10.14778/3236187.3236198. [Online]. Available: <https://doi.org/10.14778/3236187.3236198>.
- [42] Jonas Mueller and Aditya Thyagarajan, “Siamese recurrent architectures for learning sentence similarity,” *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 30, no. 1, 2016. DOI: 10.1609/aaai.v30i1.10350. [Online]. Available: <https://ojs.aaai.org/index.php/AAAI/article/view/10350>.
- [43] Jeffrey Pennington, Richard Socher, and Christopher D Manning, “Glove: Global vectors for word representation,” in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.
- [44] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov, “Enriching Word Vectors with Subword Information,” *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 135–146, Jun. 2017, ISSN: 2307-387X. DOI: 10.1162/tacl.a.00051. eprint: <https://direct.mit.edu/tacl/article-pdf/doi/10.1162/tacl.a.00051/1567442/tacl.a.00051.pdf>. [Online]. Available: <https://doi.org/10.1162/tacl.a.00051>.

- [45] Cheng Fu, Xianpei Han, Le Sun, Bo Chen, Wei Zhang, Suhui Wu, and Hao Kong, “End-to-end multi-perspective matching for entity resolution,” in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, International Joint Conferences on Artificial Intelligence Organization, Jul. 2019, pp. 4961–4967. DOI: 10.24963/ijcai.2019/689. [Online]. Available: <https://doi.org/10.24963/ijcai.2019/689>.
- [46] Hao Nie, Xianpei Han, Ben He, Le Sun, Bo Chen, Wei Zhang, Suhui Wu, and Hao Kong, “Deep sequence-to-sequence entity matching for heterogeneous entity resolution,” in *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, ser. CIKM ’19, Beijing, China: Association for Computing Machinery, 2019, 629–638, ISBN: 9781450369763. DOI: 10.1145/3357384.3358018. [Online]. Available: <https://doi.org/10.1145/3357384.3358018>.
- [47] Cheng Fu, Xianpei Han, Jiaming He, and Le Sun, “Hierarchical matching network for heterogeneous entity resolution,” in *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, 2021, pp. 3665–3671.
- [48] Chen Zhao and Yeye He, “Auto-em: End-to-end fuzzy entity-matching using pre-trained deep models and transfer learning,” in *The World Wide Web Conference*, ser. WWW ’19, San Francisco, CA, USA: Association for Computing Machinery, 2019, 2413–2424, ISBN: 9781450366748. DOI: 10.1145/3308558.3313578. [Online]. Available: <https://doi.org/10.1145/3308558.3313578>.
- [49] Wei Zhang, Hao Wei, Bunyamin Sisman, Xin Luna Dong, Christos Faloutsos, and Davd Page, “Autoblock: A hands-off blocking framework for entity matching,” in *Proceedings of the 13th International Conference on Web Search and Data Mining*, ser. WSDM ’20, Houston, TX, USA: Association for Computing Machinery, 2020, 744–752, ISBN: 9781450368223. DOI: 10.1145/3336191.3371813. [Online]. Available: <https://doi.org/10.1145/3336191.3371813>.
- [50] Ursin Brunner and Kurt Stockinger, “Entity matching with transformer architectures - a step forward in data integration,” en, in *Proceedings of EDBT 2020*, 23rd International Conference on Extending Database Technology, Copenhagen, 30 March - 2 April 2020, OpenProceedings, 2020, ISBN: 978-3-89318-083-7. DOI: 10.21256/zhaw-19637. [Online]. Available: <https://digitalcollection.zhaw.ch/handle/11475/19637>.
- [51] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan, “Supervised contrastive learning,” in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, Eds., vol. 33, Curran Associates, Inc., 2020, pp. 18 661–18 673. [Online]. Available: <https://proceedings.neurips.cc/paper/2020/file/d89a66c7c80a29b1bdbab0f2a1a94af8-Paper.pdf>.

- [52] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei, “Language models are few-shot learners,” in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, Eds., vol. 33, Curran Associates, Inc., 2020, pp. 1877–1901. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfc4967418bfb8ac142f64a-Paper.pdf.
- [53] Qian Chen, Zhu Zhuo, and Wen Wang, “BERT for joint intent classification and slot filling,” *CoRR*, vol. abs/1902.10909, 2019. arXiv: 1902.10909. [Online]. Available: <http://arxiv.org/abs/1902.10909>.
- [54] Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir Radev, *Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task*, 2019. arXiv: 1809.08887 [cs.CL].
- [55] Victor Zhong, Caiming Xiong, and Richard Socher, “Seq2sql: Generating structured queries from natural language using reinforcement learning,” *CoRR*, vol. abs/1709.00103, 2017. arXiv: 1709.00103. [Online]. Available: <http://arxiv.org/abs/1709.00103>.
- [56] Panupong Pasupat and Percy Liang, “Compositional semantic parsing on semi-structured tables,” *CoRR*, vol. abs/1508.00305, 2015. arXiv: 1508.00305. [Online]. Available: <http://arxiv.org/abs/1508.00305>.
- [57] Mohit Iyyer, Wen-tau Yih, and Ming-Wei Chang, “Search-based neural structured learning for sequential question answering,” in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vancouver, Canada: Association for Computational Linguistics, Jul. 2017, pp. 1821–1831. DOI: 10.18653/v1/P17-1167. [Online]. Available: <https://aclanthology.org/P17-1167>.
- [58] Zhoujun Cheng, Haoyu Dong, Zhiruo Wang, Ran Jia, Jiaqi Guo, Yan Gao, Shi Han, Jian-Guang Lou, and Dongmei Zhang, “Hitab: A hierarchical table dataset for question answering and natural language generation,” *CoRR*, vol. abs/2108.06712, 2021. arXiv: 2108.06712. [Online]. Available: <https://arxiv.org/abs/2108.06712>.
- [59] Fengbin Zhu, Wenqiang Lei, Youcheng Huang, Chao Wang, Shuo Zhang, Jiancheng Lv, Fuli Feng, and Tat-Seng Chua, “TAT-QA: A question answering benchmark on a hybrid of tabular and textual content in finance,” *CoRR*, vol. abs/2105.07624, 2021. arXiv: 2105.07624. [Online]. Available: <https://arxiv.org/abs/2105.07624>.
- [60] Wenhui Chen, Ming-Wei Chang, Eva Schlinger, William Yang Wang, and William W. Cohen, “Open question answering over tables and text,” *CoRR*, vol. abs/2010.10439, 2020. arXiv: 2010.10439. [Online]. Available: <https://arxiv.org/abs/2010.10439>.

- [61] Wenhui Chen, Hanwen Zha, Zhiyu Chen, Wenhan Xiong, Hong Wang, and William Yang Wang, “Hybridqa: A dataset of multi-hop question answering over tabular and textual data,” *CoRR*, vol. abs/2004.07347, 2020. arXiv: 2004.07347. [Online]. Available: <https://arxiv.org/abs/2004.07347>.
- [62] Zhiyu Chen, Wenhui Chen, Charese Smiley, Sameena Shah, Iana Borova, Dylan Langdon, Reema Moussa, Matt Beane, Ting-Hao Huang, Bryan R. Routledge, and William Yang Wang, “Finqa: A dataset of numerical reasoning over financial data,” *CoRR*, vol. abs/2109.00122, 2021. arXiv: 2109.00122. [Online]. Available: <https://arxiv.org/abs/2109.00122>.
- [63] Rebecca Allen, *Amazon standard identification number (asin)*, Accessed: February 20, 2023, 1996. [Online]. Available: https://en.wikipedia.org/wiki/Amazon\._Standard\._Identification\._Number.
- [64] Alan Haberman, *Universal product code (upc or upc code)*.
- [65] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova, “BERT: Pre-training of deep bidirectional transformers for language understanding,” in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 4171–4186. DOI: 10.18653/v1/N19-1423. [Online]. Available: <https://aclanthology.org/N19-1423>.
- [66] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov, “Roberta: A robustly optimized bert pretraining approach,” *arXiv preprint arXiv:1907.11692*, 2019. [Online]. Available: <https://arxiv.org/abs/1907.11692>.
- [67] Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang, “How to fine-tune bert for text classification?” In *Chinese Computational Linguistics: 18th China National Conference, CCL 2019, Kunming, China, October 18–20, 2019, Proceedings*, Kunming, China: Springer-Verlag, 2019, 194–206, ISBN: 978-3-030-32380-6. DOI: 10.1007/978-3-030-32381-3_16. [Online]. Available: https://doi.org/10.1007/978-3-030-32381-3_16.
- [68] OpenAI, *Pricing: Simple and flexible. only pay for what you use*. Accessed: May 20, 2023, 2023. [Online]. Available: <https://openai.com/pricing>.
- [69] Andrew Yates, Rodrigo Nogueira, and Jimmy Lin, “Pretrained transformers for text ranking: Bert and beyond,” in *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, ser. WSDM ’21, Virtual Event, Israel: Association for Computing Machinery, 2021, 1154–1156, ISBN: 9781450382977. DOI: 10.1145/3437963.3441667. [Online]. Available: <https://doi.org/10.1145/3437963.3441667>.

- [70] Nils Reimers and Iryna Gurevych, “Sentence-BERT: Sentence embeddings using Siamese BERT-networks,” in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 3982–3992. DOI: 10.18653/v1/D19-1410. [Online]. Available: <https://aclanthology.org/D19-1410>.
- [71] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala, “Pytorch: An imperative style, high-performance deep learning library,” in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32, Curran Associates, Inc., 2019. [Online]. Available: <https://proceedings.neurips.cc/paper/2019/file/bdbca288fee7f92f2bfa9f7012727740-Paper.pdf>.
- [72] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush, “Transformers: State-of-the-art natural language processing,” in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, Online: Association for Computational Linguistics, Oct. 2020, pp. 38–45. DOI: 10.18653/v1/2020.emnlp-demos.6. [Online]. Available: <https://aclanthology.org/2020.emnlp-demos.6>.
- [73] Yuliang Li, Jinfeng Li, Yoshihiko Suhara, Jin Wang, Wataru Hirota, and Wang-Chiew Tan, “Deep entity matching: Challenges and opportunities,” *J. Data and Information Quality*, vol. 13, no. 1, 2021, ISSN: 1936-1955. DOI: 10.1145/3431816. [Online]. Available: <https://doi.org/10.1145/3431816>.