Prompt-tuning with Attribute Guidance for Low-resource Entity Matching

Lihui Liu[1⋆] and Carl Yang[2]

[1] Wayne State University, Detroit, MI, USA
[2] Emory University, Atlanta, GA, USA

Abstract. Entity Matching (EM) is a significant task involving the determination of the logical relationship between two entities, such as `Same`, `Different`, and `Undecidable`. Traditional approaches to entity matching (EM) heavily depend on supervised learning, which necessitates a vast collection of high-quality labeled data. This labeling process is both time-consuming and costly, limiting the practical application of these methods. Consequently, there is a pressing demand for low-resource EM solutions that can perform effectively with minimal labeled data. Recently, prompt tuning-based approaches have shown promising results on low-resource entity matching, but they tend to focus solely on entity-level matching, overlooking crucial attribute-level information. Moreover, they lack interpretability and explainability. To address this limitation, this paper introduces PROMPTATTRIB, a comprehensive solution that tackles entity matching challenges through attribute level prompt tuning and logical reasoning. PROMPTATTRIB leverages both entity-level and attribute-level prompts to enhance matching accuracy by incorporating valuable contextual information, and it induces the matching label by fuzzy logic formulas. By considering attributes, the model gains a deeper understanding of the entities, leading to improved matching results. Moreover, PROMPTATTRIB incorporates dropout-based contrastive learning on soft prompts, inspired by the SimCSE technique. This further improves the performance of entity matching. Extensive experiments on real-world datasets demonstrate the efficacy of PROMPTATTRIB.

Introduction

Entity Matching (EM) is a fundamental problem in data management that focuses on determining whether two entity records refer to the same real-world entity. This task carries significant importance in various domains such as question answering [17,10], dialog system [19,1,18], recommender system [23] and so on. In practical terms, consider a situation where two distinct knowledge databases store customer information from different sources. Due to differences in schema and data formats, effectively combining this information becomes a challenging task. However, by leveraging entity matching techniques, it becomes possible to identify matching customer records across knowledge databases, facilitating accurate data merging and linking. Thus, the application of entity matching plays a crucial role in improving data quality and enabling seamless information consolidation across diverse sources.

⋆ Lihui Liu is the first author and the corresponding author.

Conventional entity matching (EM) techniques rely extensively on abundant high-quality labeled data, which is often scarce or unavailable. Therefore, there is a growing need for low-resource EM methods that can achieve robust performance with minimal labeled data. Prompt tuning is a promising way for training and fine-tuning language models to tackle low-resource entity matching. By carefully crafting and adjusting the initial prompt given to the model, researchers and developers can guide the model's behavior and steer it towards desired outcomes. Existing prompt tuning based methods for low-resource entity matching, such as promptEM [22], commonly serialize each entity into a sentence, and a prompt template is used to generate the input for the language model. Despite these approaches have shown good performance in many scenarios, such prompt tuning based models function like a black-box and lack interpretability. Moreover, they consider only the entity-level information and overlook the crucial attribute information associated with each entity.

However, attribute information plays a significant role in accurately determining entity matches because it contains more fine-grained details. It can help us explain why two entities are the `Same`, `Different`, or `Ambiguous`. For example, Michael Jordan with the occupation "basketball player" is different from Michael Jordan with the occupation "computer scientist". Exploring innovative ways to incorporate attribute-level information holds promise for enhancing the robustness and reliability of entity matching systems.

In addition, existing prompt tuning methods typically assume that the input prompt embedding is well-trained during the training process, without explicitly making the representations of different inputs distinguishable from each other. However, high-quality entity/attribute representations are crucial for the success of entity matching. If two entities/attributes are different, their representations should be easily distinguishable. Taking inspiration from SimCSE, a technique that utilizes dropout-based contrastive learning on language models to enhance representation learning quality, we propose applying dropout-based contrastive learning on prompt tuning. This approach aims to increase the model's resilience to variations in input prompts by introducing dropout mechanisms during the training phase. By incorporating dropout-based contrastive learning on soft prompts, we seek to improve the overall performance of the entity matching model, ultimately leading to more accurate and consistent results.

In this paper, we present PROMPTATTRIB, a comprehensive solution consisting of two main components. First, PROMPTATTRIB leverages both entity-level prompts and attribute-level prompts to effectively address the challenges related to low-resource entity matching. By incorporating fuzzy logic reasoning, the model predicts entity-level matches based on the logical relationships between different attribute information, making the prediction results more explainable. Second, PROMPTATTRIB employs dropout-based contrastive learning on soft prompts. This technique enhances the model's performance by encouraging distinguishable feature representations and reducing overfitting. Through dropout-based contrastive learning, the model learns to generalize better and captures more nuanced relationships between entities and attributes.

In summary, the main contributions of this paper are:

- **Algorithm**: We propose PROMPTATTRIB, which utilizes entity-level prompts, attribute-level prompts, fuzzy logic reasoning, and soft token contrastive learning to tackle low-resource entity matching.
- **Empirical Evaluations**: We conducted extensive experiments on several real-world datasets. The results of our experiments demonstrate the effectiveness of PROMPTATTRIB.

Problem Definition

Low-resource entity matching (EM) involves the identification of pairs of data entries from two collections that refer to the same real-world entity with limited labeled training datapoints. Formally, given two data sources E_A and E_B, we assign a binary label $y \in \{0, 1\}$ to each candidate pair $(e_a, e_b) \in E_A \times E_B$. Here, $y = 1$ indicates a true match, while $y = 0$ represents a mismatched pair.

Problem 1. Low-resource Entity Matching:

Given: (1) an entity e_a from data source E_A, (2) an entity e_b from data source E_B, (3) limited labeled training datapoints.

Output: a binary label $y \in \{0, 1\}$ for the entity pair (e_a, e_b)

name	position	address	postalCode
McKees Rocks Bridge	40.47708; -80.04827	Mckees Rocks Bridge; Route 65	15233

Fig. 1: Example.

Serializing: The matching problem can be effectively solved by formulating it as a sequence classification task. First, entity pairs are serialized to sequences, and then, a pre-trained LM is finetuned to solve the task. An entity with n attributes can be denoted as $e = \{\text{attr}_i, \text{val}_i\}_{i\in[1,n]}$, where attr_i is the attribute name and val_i is the corresponding attribute value. Then the serialization [22] is denoted as:

$$\text{serialize}(e) ::= [\text{COL}]\text{attr}_1[\text{VAL}]\text{val}_1 \ldots [\text{COL}]\text{attr}_n[\text{VAL}]\text{val}_n$$

where [COL] and [VAL] are two special tags indicating the start of attribute names and values, respectively. Taking the relational entity in Figure 1 as an example, we serialize it as:

$$[\text{COL}]\text{name}[\text{VAL}]\text{McKees Rocks Bridge} \ldots [\text{COL}]\text{postalCode}[\text{VAL}]15233$$

Prompt-based Tuning: Prompt-based tuning has been proposed to apply cloze-style tasks to tune LMs. Formally, we define a label word set $V_{\mathbf{y}} = \{w_1, \ldots, w_m\}$. $V_{\mathbf{y}}$ is a subset of the vocabulary V of the LM, i.e., $V_{\mathbf{y}} \subseteq V$. We get an overall dictionary V^* by taking the union of the dictionary corresponding to each label. Another primary

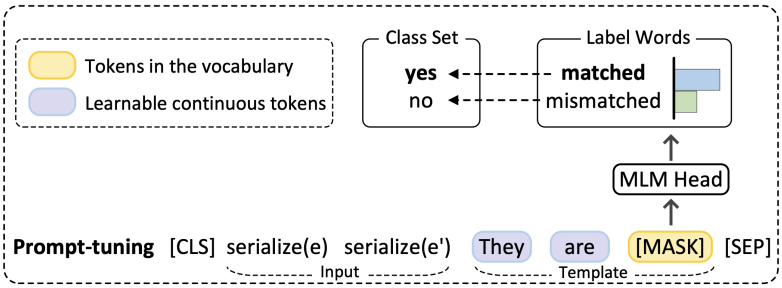

Fig. 2: The illustration of prompt-tuning. The blue rectangles in the figure are special prompt tokens, whose parameters are initialized and learnable during prompt-tuning.

component of prompt-based tuning is a prompt template $T(\cdot)$, which modifies the original input x into a prompt input $T(x)$ by adding a set of additional tokens in x. Generally, a token $[MASK]$ is added for LMs to predict the missing label word $w \in V_{\mathbf{y}}$ using $T(x)$. Thus, in prompt-tuning, a classification problem is transferred into a masked language modeling problem: $p(y \in Y|x) = p([MASK] = w \in V_{\mathbf{y}}|T(x))$, where Y is the label set.

Method

In this section, we detail how to utilize prompt-tuning to deal with EM. We first design EM-specific prompt templates and label words, and then, we describe the training and inference process.

Prompt Template

To cast the EM problem as a prompt-tuning one, we first design suitable prompt templates (i.e., hard-encoding templates and continuous templates) and label words set (to consider general binary relationship) following [22]. The hard-encoding templates and continuous templates are the same as those in [22], except that we use a different prompt template and label words set. Inspired by [22], given each candidate pair $x = (e, e')$, we construct the following templates[3]:

$$T_1(x) = \text{Are serialize}(e) \text{ and serialize}(e') \text{ the } [MASK]$$
$$T_2(x) = \text{serialize}(e) \text{ is } [MASK] \text{ to serialize}(e')$$

Since the goal of prompt construction is to optimize task performance rather than readability, prompts don't need to be human-interpretable. Continuous prompts, embedded directly in the model's space, address this need. We use P-tuning [22], where trainable prompt tokens are integrated into the embedding layer and processed by BiLSTM to capture their interactions. This approach allows the model to learn more effective prompts beyond its original vocabulary. An example is shown in Figure 2.

[3] Note that various prompts can be used in the experiments, we provide only two simple examples here.

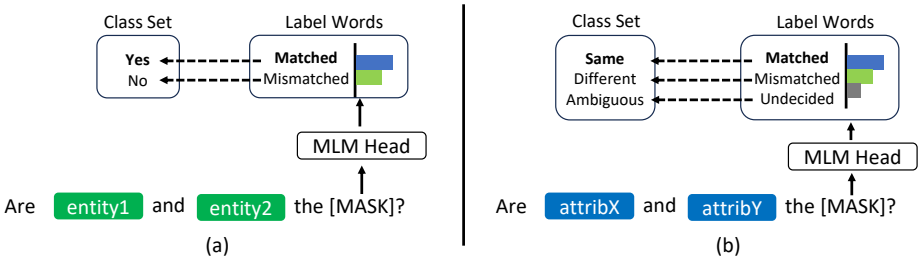

Fig. 3: The entity level prompt tuning and the attribute level prompt tuning.

Label Words Set. In addition to designing templates, another primary component is to design the set of label words. Given two entities e_1 and e_2, it is often more informative to classify the relationship between them rather than simply determining if they are identical. Specifically, we aim to determine if the entities are relevant to each other, which encompasses a broader range of relationships beyond mere matching. For a general binary relationship, we map the label $y =$ yes into a set $V_y = \{$matched, similar, relevant$\}$. Similarly, the label $y =$ no is mapped to a set $V_y = \{$mismatched, different, irrelevant$\}$. This approach allows for a more nuanced classification that captures various degrees of relevance and irrelevance between the entities.

Method Details

Entity Level Prompt-tuning Inspired by existing prompt tuning-based methods, such as PromptEM [22,11], we consider checking whether two entities are the same by prompt-tuning. More specifically, given two entities, we first serialize each entity into text using the method introduced in Subsection 1. Subsequently, these two serialized texts are used as input for the prompt template, as depicted in Figure 3(a). The resulting context is then fed as input to the large language model. The output embedding of the special token **[MASK]** is utilized by the **MLM Head** to predict the results. The label words set used in entity level prompt tuning is the same as the label words set introduced in Subsection 1. This means we map the label $y =$ yes to a set $V_y = \{$matched, similar, relevant$\}$. Similarly, the label $y =$ no is mapped to a set $V_y = \{$mismatched, different, irrelevant$\}$.

Attribute Level Prompt-tuning

Existing prompt tuning-based methods primarily focus on considering only the entity-level information of two entities. However, attribute information is often of significant importance, as it can provide essential context to the system. To leverage this attribute information effectively, one potential approach is to integrate it into the prompt. In contrast to entity-level prompts that include only one **[MASK]** token, the attribute prompt contains both entity-level and attribute-level information, featuring multiple **[MASK]** tokens in the template. The first **[MASK]** is utilized to predict the entity-level information, while the subsequent **[MASK]** tokens are employed to predict the attribute-level information.

Despite the fact that the above idea is intuitive, certain entities may contain an extensive amount of information. For instance, the **Description** field of a product,

such as a "computer," can be very lengthy, even exceeding 512 words. This situation poses a challenge as the combined length of the enhanced prompt with attribute information would exceed the model's maximum input length, necessitating truncation of the text. To address this issue, we separate the entity-level prompt from the attribute-level prompt, as illustrated in Figure 3(b). Unlike entity-level prompt tuning, the label words set used in attribute-level prompt tuning contains three categories: `Same`, `Different`, and `Ambiguous`. More specifically, we map the label $y =$ Same to a set $V_y = \{$same, similar, positive$\}$, the label $y =$ Different is mapped to a set $V_y = \{$mismatched, different, irrelevant$\}$, and the label $y =$ Ambiguous is mapped to a set $V_y = \{$uncertain, unclear, neutral$\}$.

Fuzzy Logic Reasoning

Because each entity comprises multiple attributes, entity-level matching can be logically induced from attribute-level matching. Based on the definition of entity matching, we have developed the following induction rules.

Same Rule: It is evident that if two entities are the same, then all their attributes should also be the same, implying that all paired attributes should be labeled as `Same`. Let $\{(a_k^1, a_k^2)\}_{k=1}^K$ be all attribute pairs. Then, we induce a attribute-level `Same` score by

$$S(\texttt{Same}|e_1, e_2) = \left[\prod_{k=1}^{K} P(\texttt{Same}|a_k^1, a_k^2)\right]^{\frac{1}{K}}$$

This works in a fuzzy logic fashion [6], deciding whether the entity-level label should be `Same` by considering the average of attribute level predictions. Here, we use the geometric mean because it is biased towards low scores. In other words, if there exists one attribute pair with a low `Same` score, then the chance of the entity label being `Same` is also low.

Difference Rule: Two entities are `Different` if there exists (at least) one paired attribute labeled as `Different`. The fuzzy logic version of this induction rule is given by

$$S(\texttt{Different}|e_1, e_2) = \max_{k=1,\ldots,K} P(\texttt{Different}|a_k^1, a_k^2)$$

Here, the max operator is used in the induction because the `Different` rule is an existential statement, i.e., there exist(s) $\cdots$.

Ambiguous Rule: Two entities are `Ambiguous` if there exists (at least) one `Ambiguous` attribute pair, but there does not exist any `Different` attribute pair. The fuzzy logic formula is

$$S(\texttt{Ambiguous} \mid e_1, e_2) = \left[\max_{k=1\ldots K'} P(\texttt{Ambiguous} \mid a_k^1, a_k^2)\right] [1 - S(\texttt{Different} \mid e_1, e_2)]$$

The first factor determines whether there exists a `Ambiguous` attribute pair. The second factor evaluates the negation of "at least one different attribute" as suggested in the second clause of the Rule for `Ambiguous`.

Finally, we normalize the scores into probabilities by dividing the sum, since all the scores are already positive. This is given by $P(L|\cdot) = Z^{-1}S(L|\cdot)$, where $L \in \{\texttt{Same}, \texttt{Different}, \texttt{Ambiguous}\}$, and $Z = S(\texttt{Same}|\cdot)+S(\texttt{Different}|\cdot)+S(\texttt{Ambiguous}|\cdot)$ is the normalizing factor.

During training process, we use cross-entropy loss to train our model by minimizing $-\log P(t|\cdot)$ where $t \in \{\texttt{Same}, \texttt{Different}, \texttt{Ambiguous}\}$ is the ground truth entity-level label. During the experiment, we categorize `Ambiguous` based on various datasets as either `Same` or `Different`.

Soft Token Contrastive Learning

Current methods for tuning prompts typically assume that the initial soft prompt is well-established during training and that the model's learned representation is sufficiently distinguishable. However, this assumption often does not hold, particularly in low-resource scenarios. High-quality representations of entities and attributes significantly influence performance. To tackle this issue, drawing inspiration from techniques such as SimCSE [5], we propose integrating contrastive learning into soft prompts to enhance representation learning.

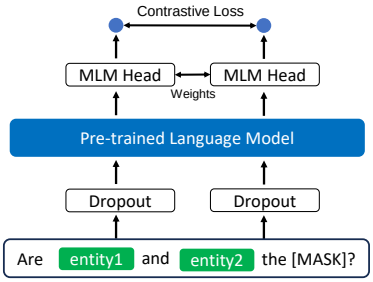

Fig. 4: Contrastive learning framework. Dropout is applied to the entire input embedding.

The idea of contrastive learning is to enhance representation learning by training a model to distinguish between similar and dissimilar instances in the data. By maximizing the similarity of representations from similar instances and minimizing that of dissimilar ones, contrastive learning fosters the creation of more meaningful and discriminative feature representations.

In PROMPTATTRIB, given an entity e_i, we construct its positive sampling data by randomly applying dropout masks to the soft prompt token inside serialize(e_i), as introduced in the previous sections, resulting in e_i^+. This approach has been utilized in various models; for instance, in the standard training of Transformers [21], dropout masks are applied to fully-connected layers and attention probabilities (typically $p = 0.1$) to enhance model robustness. After obtaining the positive data point, we pass it through the prompt-tuning model $z = f_\theta(e_i^+)$ to obtain its embedding z. This process is repeated twice using different dropout masks to generate two positive embeddings z_1 and z_2, and the training objective becomes: $L = \|z_1 - z_2\|_2$.

Note that unlike SimCSE, which applies dropout to the language model parameters, we keep the language model unchanged because modifying parameters in language

models can be time-consuming. Instead, we apply dropout to the input soft tokens within the prompt. This approach ensures that even if the input prompt context remains constant, applying dropout multiple times yields distinct inputs. This concept bears similarity to siamese networks [16], albeit focusing solely on positive pairs in the contrastive loss. The final framework of the model is shown in Figure 4

Experiment

In this research, we thoroughly evaluated our method using several datasets that represent diverse and realistic entity-matching scenarios. We used four main datasets. Geo-heter [22] contains 194,089 geospatial entities collected from platforms such as Yelp, Foursquare, and OpenStreetMap. Cameras [11] and Computers [11] are subsets of the large WDC product dataset, which includes millions of product offers from various e-commerce websites; these subsets focus specifically on camera-related and computer-related entities, including their models, brands, and specifications. Finally, the ISWC dataset [11] includes entities from the International Semantic Web Conference, such as papers, authors, venues, and publication years. Together, these datasets provide a broad and practical testbed for evaluating our approach.

Table 1: The entity matching performance of different methods. F means F1-score, P means average precision, A means Accuracy.

Dataset	geo-heter			cameras			computers			ISWC		
	F	P	A	F	P	A	F	P	A	F	P	A
SentenceBERT	58.8	53.8	64.8	38.3	21.4	21.3	39.4	35.6	57.5	66.2	68.0	75.5
DeepMatcher	43.8	28.9	72.1	39.2	22.4	23.0	42.8	37.1	58.9	69.1	70.6	76.3
Ditto	3.6	45.5	71.3	42.8	27.3	27.3	44.9	40.3	62.1	72.1	72.7	80.6
PromptEM	78.5	84.5	85.5	35.4	29.4	68.8	49.5	45.2	68.4	76.4	73.6	81.9
PROMPTATTRIB	81.1	85.4	88.9	45.5	45.6	72.2	47.7	49.5	72.1	77.5	79.6	82.5

The details of these datasets can be found in [22] and [11]. In our experiments, we use only 5% of the labeled training data to simulate a low-resource entity matching setting. To evaluate our approach, we compare it against several strong baseline methods. SentenceBERT [16] introduces a siamese architecture built on pretrained language models for sentence matching, which can also be applied to entity matching. DeepMatcher [24] is a traditional entity matching framework that uses recurrent neural networks to aggregate attribute values and align their representations. Ditto [11] is a state-of-the-art approach that fine-tunes a pretrained language model using domain knowledge, TF-IDF summarization, and data augmentation. Finally, PromptEM [22] adapts prompt-based tuning for entity matching by reformulating the task as a cloze-style masked language modeling problem. Together, these baselines provide a comprehensive comparison for assessing the effectiveness of our method.

All experiments are conducted on a machine with an Intel(R) Xeon(R) Gold 6240R CPU, 1510 GB memory, and an NVIDIA-SMI Tesla V100-SXM2 GPU.

Entity Matching Performance

In this subsection, we test the performance of different baseline methods on entity matching task. We use metrics: F1-score, Average Precision and Accuracy to measure the performances of all the baselines.

Table 1 shows the results. As we can see, for the geo-heter dataset, PROMPTATTRIB outperforms both Ditto and PromptEM, achieving the highest F1-score (81.1%), Average Precision (85.3%), and Accuracy (88.9%). PromptEM also performs well with an F1-score of 78.5%, while Ditto has the lowest F1-score of 3.6%.

In the cameras dataset, PROMPTATTRIB has the highest F1-score (45.5%), Average Precision (45.6%) and Accuracy (72.2%), while PromptEM achieves the second highest Accuracy of 0.688 compared to other methods. For the computers dataset, PROMPTATTRIB again outperforms the other methods, achieving the highest F1-score (47.7%), Average Precision (49.5%) and Accuracy (72.1%). PromptEM follows closely with an F1-score of 49.5% and an Average Precision of 45.2%. Ditto lags behind with an F1-score of 44.9%. In the ISWC dataset, PROMPTATTRIB also performs the best, obtaining the highest F1-score (77.5%), Average Precision (79.6%) and Accuracy (82.5%). PromptEM closely follows with an F1-score of 76.4% and, Average Precision of 73.6% and Accuracy 81.9%. Ditto achieves an F1-score of 72.1%. Overall, the results indicate that PROMPTATTRIB consistently outperforms other baselines across the different datasets, showcasing its effectiveness in entity matching tasks.

Performance of Different Language Models

We also conducted experiments using various large language models as the backbone of the prompt tuning model. Specifically, we evaluated six models, including Bert [3], Roberta [13], Albert [8], GPT2 [15], Roberta-large, and Albert-large. The performance results are presented in Table 2.

Table 2: The entity matching performance of different language models as backbone. Dataset is geo-heter.

	F	P	A
BERT	77.2	73.5	83.4
Roberta	79.5	78.2	86.4
Albert	78.7	80.2	86.6
GPT2	64.7	68.4	79.7
Roberta-large	**81.1**	**85.4**	**88.9**
Albert-large	79.9	83.1	86.2

As we can see, the results show that among the tested language models, Roberta-large achieved the highest F1-score (81.1%), Average Precision (85.4%) and Accuracy (88.9%), closely followed by Albert-large (F1-score: 79.9%, Average Precision: 83.1%, and Accuracy: 86.2%). Roberta also performs well with an F1-score of 79.5% and an Average Precision of 78.2%. BERT obtained an F1-score of 77.2% and an Average Precision of 73.5%. GPT2 achieved a lower F1-score of 64.7% and an Average Precision

of 68.4% compared to the other language models. In terms of Accuracy, Roberta-large again stands out with a value of 88.9%, followed closely by Albert (Accuracy: 86.6%) and Albert-large (Accuracy: 86.2%). Overall, the results demonstrate that language models, such as Roberta-large and Albert-large, tend to perform better than smaller ones like BERT and GPT2, indicating the importance of model size and capacity in improving entity matching performance.

Ablation Study

In this subsection, we examine the effectiveness of integrating contrastive learning with attribute-level prompt tuning. We evaluate how this combination improves model's overall performance, particularly in low-resource scenarios. By analyzing various metrics and comparing results under different scenarios, we aim to demonstrate their advantages.

Table 3: The entity matching performance with contrastive learning under different dropout ratios. Dataset is cameras.

Model	F	P	A
Ditto	42.9	27.3	27.3
PromptEM	42.3	31.7	71.3
PROMPTATTRIB No	40.4	39.7	71.7
PROMPTATTRIB 0.35	45.5	45.6	72.3
PROMPTATTRIB 0.4	37.6	39.0	70.9
PROMPTATTRIB 0.45	39.1	39.5	73.1

Dropout ratio. Table 3 shows the entity matching performance with contrastive learning under different dropout ratios, where "PROMPTATTRIB 0.35" means the dropout ratio is 0.35 and "PROMPTATTRIB No" means without dropout. As we can see, PROMPTATTRIB with a dropout ratio of 0.35 achieves the highest F-score (45.5%) and average precision (45.6%), indicating that this ratio provides the best balance between regularization and information retention. Comparing "PROMPTATTRIB No" and PROMPTATTRIB with different dropout ratios, it is evident that applying dropout generally improves performance. For instance, "PROMPTATTRIB No" achieves an F-score of 40.4%, while "PROMPTATTRIB 0.35" achieves a significantly higher F-score of 45.5%. This improvement suggests that dropout helps in preventing overfitting and enhances the model's ability to generalize from the training data. Interestingly, PROMPTATTRIB with a dropout ratio of 0.4 and 0.45 shows a decrease in F-score (37.6% and 39.1%, respectively) compared to "PROMPTATTRIB 0.35", indicating that too much dropout can degrade performance. However, the accuracy (A) metric is highest for "PROMPTATTRIB 0.45" at 73.1%, suggesting that while the F-score and average precision metrics are sensitive to dropout ratio, while accuracy still benefit from higher dropout ratio.

Related Work

Entity Matching: Entity Matching (EM) is a critical task in data management, essential for ensuring data quality and consistency. Various methods have been explored, including

rule-based approaches, crowdsourcing techniques, and traditional machine learning (ML) models. Recently, deep learning (DL) has shown significant promise in EM. For instance, DeepER [4] employs neural networks to extract features and treats EM as a classification problem. Similarly, DeepMatcher [24] provides a comprehensive DL framework for EM. However, DL approaches often require substantial labeled training data, which is costly and impractical. To address this, Ditto [11] leverages pre-trained language models and incorporates data augmentation to improve EM performance with less training data. Additionally, Rotom [14] enhances EM tasks by integrating multiple data augmentation operators, while DADER [20] advances EM through domain adaptation techniques. Other strategies, such as information fusion, active learning, and transfer learning, have also been investigated to boost EM performance.

Prompt Tuning: Despite the success of fine-tuning pre-trained language models (LMs) [13,3,8], the gap between pre-training and fine-tuning objectives limits the full potential of pre-trained knowledge. The introduction of GPT-3 [2] sparked interest in prompt-tuning [9], which uses hand-crafted prompts to achieve strong performance on various tasks, particularly in low-resource settings. Building on GPT-3, many hand-encoded prompts have been explored. Recently, methods like continuous prompts [12,7] have emerged, reducing the need for manual prompt design and improving prompt expressiveness. Prompt-tuning has led to advancements in natural language inference and entity typing.

Conclusion

In this paper, we propose PromptAttrib, a comprehensive solution with two main components for effective entity matching. The first part leverages entity-level prompts and attribute-level prompts to address challenges in entity matching. By incorporating these prompts, the model gains valuable contextual information, enhancing accuracy. In the second part, PromptAttrib employs dropout-based contrastive learning on soft prompts, promoting robust feature representations and reducing overfitting. This approach enables better generalization and captures nuanced entity-attribute relationships. Experimental results demonstrate the efficacy of PromptAttrib, showcasing significant improvements in entity matching performance.

Acknowledge

This research was partially supported by the the internal funds and GPU servers provided by the Computer Science Department of Emory University, the US National Science Foundation under Award Numbers 2442172, 2312502, 2319449, and the US National Institutes of Health under Award Numbers K25DK135913, RF1NS139325, R01DK143456 and U18DP006922.

References

1. Acharya, A., Adhikari, S.: Alexa conversations: An extensible data-driven approach for building task-oriented dialogue systems (2021)

2. Brown, T.B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P.: Language models are few-shot learners (2020)
3. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding (2019)
4. Ebraheem, M., Thirumuruganathan, S., Joty, S., Ouzzani, M., Tang, N.: Distributed representations of tuples for entity resolution. Proc. VLDB Endow. **11**(11), 1454–1467 (jul 2018). https://doi.org/10.14778/3236187.3236198, https://doi.org/10.14778/3236187.3236198
5. Gao, T., Yao, X., Chen, D.: SimCSE: Simple contrastive learning of sentence embeddings. In: Empirical Methods in Natural Language Processing (EMNLP) (2021)
6. Hájek, P.: Metamathematics of fuzzy logic, vol. 4. Springer Science & Business Media (2013)
7. Han, X., Zhao, W., Ding, N., Liu, Z., Sun, M.: Ptr: Prompt tuning with rules for text classification (2021)
8. Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., Soricut, R.: Albert: A lite bert for self-supervised learning of language representations (2020)
9. Lester, B., Al-Rfou, R., Constant, N.: The power of scale for parameter-efficient prompt tuning (2021)
10. Lewis, M., Liu, Y., Goyal, N.: Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension (2019)
11. Li, Y., Li, J., Suhara, Y., Doan, A., Tan, W.C.: Deep entity matching with pre-trained language models. VLDB (2020)
12. Liu, X., Zheng, Y., Du, Z., Ding, M., Qian, Y., Yang, Z., Tang, J.: Gpt understands, too (2023)
13. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V.: Roberta: A robustly optimized bert pretraining approach (2019)
14. Miao, Z., Li, Y., Wang, X.: Rotom: A meta-learned data augmentation framework for entity matching, data cleaning, text classification, and beyond. In: Proceedings of the 2021 International Conference on Management of Data. p. 1303–1316. SIGMOD '21, Association for Computing Machinery, New York, NY, USA (2021). https://doi.org/10.1145/3448016.3457258, https://doi.org/10.1145/3448016.3457258
15. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al.: Language models are unsupervised multitask learners. OpenAI blog **1**(8), 9 (2019)
16. Reimers, N., Gurevych, I.: Sentence-bert: Sentence embeddings using siamese bert-networks. arXiv preprint arXiv:1908.10084 (2019)
17. Tapaswi, M., Zhu, Y., Stiefelhagen, R., Torralba, A., Urtasun, R., Fidler, S.: Movieqa: Understanding stories in movies through question-answering (2016)
18. Thoppilan, R., Freitas, D.D., Hall, J.: Lamda: Language models for dialog applications. arXiv (2022). https://doi.org/10.48550/ARXIV.2201.08239, https://arxiv.org/abs/2201.08239
19. Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., et al.: Llama 2: Open foundation and fine-tuned chat models. arXiv preprint arXiv:2307.09288 (2023)
20. Tu, J., Fan, J., Tang, N., Wang, P., Chai, C., Li, G., Fan, R., Du, X.: Domain adaptation for deep entity resolution. In: Proceedings of the 2022 International Conference on Management of Data. p. 443–457. SIGMOD '22, Association for Computing Machinery, New York, NY, USA (2022). https://doi.org/10.1145/3514221.3517870, https://doi.org/10.1145/3514221.3517870
21. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need (2017)
22. Wang, P., Zeng, X., Chen, L., Ye, F., Mao, Y., Zhu, J., Gao, Y.: Promptem: prompt-tuning for low-resource generalized entity matching. VLDB (2022)

23. Wang, Y., Zhao, Y., Zhang, Y., Derr, T.: Collaboration-aware graph convolutional network for recommender systems. In: Proceedings of the ACM Web Conference 2023. p. 91–101. WWW '23, Association for Computing Machinery, New York, NY, USA (2023). https://doi.org/10.1145/3543507.3583229, https://doi.org/10.1145/3543507.3583229
24. Xie, T., Dai, K., Wang, K., Li, R., Zhao, L.: Deepmatcher: a deep transformer-based network for robust and accurate local feature matching. Expert Systems with Applications **237**, 121361 (2024)