PromptER: Prompt Contrastive Learning for Generalized Entity Resolution

Chaofan Dai, Qideng Tang, Wubin Ma, Yahui Wu, Haohao Zhou, and Huahua Ding

Science and Technology on Information Systems Engineering Laboratory, National University of Defense Technology, Changsha, China

Abstract. Entity resolution (ER), which aims to identify whether data records from various sources refer to the same real-world entity, is a crucial part of data integration systems. Traditional ER solutions assumes that data records are stored in relational tables with an aligned schema. However, in practical applications, it is common that data records to be matched may have different formats (e.g., relational, semi-structured, or textual types). In order to support ER for data records with varying formats, Generalized Entity Resolution has been proposed and has recently gained much attention. In this paper, we propose PromptER, a model based on pre-trained language models that offers an efficient and effective approach to accomplish Generalized Entity Resolution tasks. PromptER starts with a supervised contrastive learning process to train a Transformer encoder, which is afterward used for blocking and fine-tuned for matching. Specially, in the record embedding process, PromptER uses the proposed prompt embedding technique to better utilized the pre-trained language model layers and avoid embedding bias. Morever, we design a novel data augmentation method and an evaluation method to enhance the performance of the proposed model. We conduct experiments on the Generalized Entity Resolution dataset Machamp and the results show that PromptER significantly outperforms other state-of-art methods in the blocking and matching tasks.

Keywords: Entity resolution, data integration, deep learning, contrastive learning, prompt learning

1 Introduction

Entity resolution(ER), also known as entity matching, aims to identify records from multiple data sources that refer to the same entity in the real world, which is a fundamental problem in data integration and data cleaning [1]. ER has been widely applied in various fields such as constructing knowledge bases, building data warehouses, and e-commerce, thus attracting significant attention. In recent years, deep learning models, especially pre-trained language models (PLMs) have been proposed to address the ER problem and achieve the state-of-the art matching results among ER tasks [2–5].

Although great progress has been made in current entity resolution methods, there are still some problems with practical applications. Most entity resolution methods [2, 6, 7] assume that the records to be matched are stored in relational tables with an aligned schema. However, entities can be represented in various formats. As shown in Figure 1, for paper matching, paper metadata may be stored in relational tables or semi-structured JSON files, while paper descriptions (e.g., abstract) are textual data. When the data schema heterogeneity is severe, it is not practical to unify their schemas since we need a potentially expensive schema matching in the pre-processing step.

To support more application scenarios, based on the setting and datasets of Machamp [8], we focus on the research of generalized entity resolution between heterogeneous data. The existing methods designed for generalized entity resolution are mostly based on PLMs, and these methods need to be further improved in the following two aspects:

David C. Wyld et al. (Eds): CNSA, DMDB, NIAI, ICCIoT, AIAP -2024
pp. 33-48, 2024. CS & IT - CSCP 2024 DOI: 10.5121/csit.2024.140102

title	authors	journal	year
Joint entity resolution on multiple datasets	Steven Euijong Whang, Hector Garcia-Molina	The VLDB Journal	2013

Relational

{
"title": "Joint entity resolution on multiple datasets",
"author": ["Steven Euijong Whang", "Hector Garcia-Molina"],
"publication_info":{
"publication_date": "2013/2/15",
"volume": "22",
"number": "6"}
}

Semi-Structured

" …..Our approach also makes it possible to run ER on subsets of similar records at a time, important when the full data are too large to resolve together. We study the scheduling and coordination of the individual ER algorithms, in order to resolve the multiple datasets, and show the scalability of our approach….."

Textual

Fig. 1. An example of generalized entity resolution.

The conflict between efficiency and effectiveness: The entity resolution models based on PLMs can be divided into two categories regarding representation learning: independent or interdependent representation [9]. Interdependent representation models [2, 3, 10, 11] have a deep interaction between pairs of records through attention mechanisms, resulting in better matching quality. Despite being effective, interdependent representation models come with a poor scalability for the quadratic searching space of record pairs, thus need additional blocking steps. Independent representation models [12–14] employ a representation-then-comparison scheme, wherein each record is first encoded into a representation, which is then compared using a learnable classifier. Independent representation models can integrate blocking techniques such as nearest neighbor search and local sensitive hash to effectively reduce the searching space. Albeit being able to reduce time cost, independent representation models suffer from substantial performance declines comparing to interdependent models, due to lack of fine-grained comparison between records. Ideally, we want a model can be used for blocking while achieving high matching performance.

The gap between pre-training and fine-tuning: Although methods based on PLMs have achieved impressive results in entity resolution tasks, recent research [15, 16] shows that the gap between the objectives of pre-training and fine-tuning, which restricts the full utilization of knowledge in pre-trained language models. Taking the fine-tuning process of PLMs on entity resolution tasks as an example, the pretraining is formalized as a fill-in-the-blanks task to predict mask words, while the fine-tuning is using the [CLS] embedding incorporating with classification layers to do match or mismatch decision. Further studies [17, 18] have also shown that using the [CLS] token embedding of PLMs is a suboptimal choice, as it suffers from embedding bias issues. They present that the [CLS] embedding of records in the feature space is very close, regardless of whether the records refer to the same entity or not. This is undesirable for entity resolution since it creates difficulties in distinguishing between matching and non-matching records.

To improve entity resolution in the above two aspects, we propose a novel model called PromptER, which using contrastive learning and prompt representation to enhance the independent representation model. Our model can be applied for both matching and blocking, and it solves the problem of poor matching performance in traditional independent representation methods. Our model starts with contrastive pre-training to minimize the distances between the representations of the same entity and maximizes the distances of different entity. The pretrained embedding model is then utilized to convert records into representations, which are used for downstream blocking and matching tasks. Motivated by the prompt-tuning techniques [15], which has reformulated NLP tasks as fill-in-the-blanks problems to bridge the gap of objective forms between pre-training and fine-tuning, we propose a prompt-based method by using the template to obtain the record representations. Prompt-based method can utilize the original PLM layers and avoid embedding bias. The experiments show that our proposed method achieves comparable performance to interdependent representation models on matching tasks, while also exhibiting excellent performance on blocking tasks.

Contribution: In summary, the contributions of this paper are:

- We present PromptER, a model that can address both matching and blocking tasks in generalized entity resolution and outperforms state-of-the-art models on both tasks.
- For records representation, we propose the prompt embedding method to tackle previous embedding bias problem. For contrastive pretraining, we introduce a prompt-based data augmentation method and a novel evaluation method. The prompt-based data augmentation method avoids distort to original input and further improves the performance. The evaluation method reduces extra pretraining time consumption and prevents overfitting.
- We conduct a series of ablation experiments to further demonstrate the effectiveness of prompt embedding on different PLMs, as well as its impact on records embedding similarity distribution and pretraining convergence speed.

2 RELATED WORK

Entity resolution: Entity resolution is a process that is typically divided into two steps: blocking and matching [19]. The blocking step groups similar records into clusters to reduce the computational cost of entity resolution, while the matching step involves comparing records within the same cluster to determine whether they match. In recent years, deep learning-based methods have achieved the best results in both blocking (e.g., [13, 20, 21]) and matching tasks (e.g., [2, 6, 22]). However, these methods only consider one of the two steps, requiring the design and training of separate models for each step in the entity resolution process, resulting in additional labor and training time. Although independent representation models can somewhat alleviate this issue, their performance is relatively poor due to the lack of detailed comparison between record pairs. Therefore, there is a need for a unified model that can handle both blocking and matching tasks simultaneously and achieve good performance on both.

Contrastive learning: Contrastive learning is a technique in deep learning that aims to learn representations of data by contrasting similar and dissimilar pairs [23]. The idea is to create a representation space in which similar examples are mapped to nearby points, while dissimilar examples are mapped to faraway points. The contrastive learning method has achieved great success in natural language processing tasks such as information retrieval and semantic matching. SupCon [12] first introduced contrastive pre-training in entity resolution, greatly improving the matching performance. Supervised contrastive

learning still faces some challenges, including embedding bias [17] and learned representation robustness. PromptER applies prompt embedding to reduce embedding bias and source-aware sampling to increase the robustness of the learned embeddings.

Prompt tuning: Prompt tuning is a new paradigm proposed in the field of natural language processing, which changes the traditional pretraining-fine-tuning mode to a pretraining-prompt tuning mode [15]. By constructing prompt templates, downstream tasks can be transformed into fill-in-the-blank forms of upstream tasks, which can more effectively utilize the original network structure of PLM and the prior knowledge obtained from pretraining. PromptEM [10] is the first work that applies prompt tuning for entity resolution tasks and performs well under low-resource and sufficient resource settings. Recently, some work also adopted the prompt-based method to eliminate embedding bias in PLMs [24], which can also be used to improve the embedding quality of records in entity resolution tasks.

3 PRELIMINARIES

This section provides a formal definition of generalized entity resolution problem and introduces the serializing method for this problem.

3.1 Problem Formulation

Entity resolution is to identify records across multiple data sources that correspond to the same real-world entity. In order to extend entity matching to more practical scenarios, Machamp [8] proposes a new research problem called generalized entity matching. Generalized entity matching can support matching between various data formats including relational, semi-structural, or textual types. In this paper, we formally define the blocking and matching tasks of generalized entity matching as follows:

Definition 1 (Blocking). Given two structured, semi-structured, or unstructured data sources $E_A \triangleq [x_1, \cdots, x_n]$ and $E_B \triangleq [x_1, \cdots, x_m]$ with n and m records, blocking outputs a subset of candidate pairs $C \subseteq [n] \times [m]$, such that for any $(r_a, r_b) \in C$, record r_a and r_b are likely to refer to same entity.

Definition 2 (Matching). Given the candidate set C to be matched, matching is to assign a binary label $y \in \{0, 1\}$ for each record pair $(r_a, r_b) \in C$, as 1 for match and 0 for mismatch.

3.2 Serializing

Since PLMs require token sequences as input, a key step is to convert records into token sequences. The existing serialization methods are primarily designed for structured homogeneous data, which are not suitable for generalized entity resolution. Following [8], we design a serialization method that transforms original structured or semi-structured data into token sequences while preserving the structural information.

Relational data: For relational data, each record take the form $r = \{\text{attr}_\text{i}, \text{val}_\text{i}\}$, where attr_i is the attribute name and val_i is the attribute value. The serialization method is illustrated as follows:

$$\textit{serialize}\,(r) : [\text{COL}]\,\text{attr}_1\,[\text{VAL}]\,\text{val}_1 \cdots [\text{COL}]\,\text{attr}_\text{n}\,[\text{VAL}]\,\text{val}_\text{n}$$

where [COL] and [VAL] are two special tokens indicating the start of attribute names and values respectively.

Semi-structured data: The semi-structured data is serialized in a similar way. Specially, (i) For nested attributes, we recursively add [COL] and [VAL] along with the attribute names and values for each level of nests. (ii) For attribute value containing a list, we concatenate the elements in the list into a space-separated string.

Textual data: Textual data is sequence originally, so it can be feed directly to PLMs.

4 Method

In this section, we introduce the proposed model, PromptER. We present the overall framework of the model in Section 4.1, followed by detailed explanations of the techniques used for prompt embedding and contrastive learning in Sections 4.2 and 4.3. In Sections 4.4 and 4.5, we describe the process of blocking and matching.

4.1 Framework

An illustration of the PomptER framework is shown in Figure 2. PromptER starts with step 1 and the prompt embedding method is applied to generate representation for each record. During step 2 of the contrastive pretraining process, the embedding model parameters are updated to enable the generated representation to capture the semantic similarity between records. In step 3, we use K nearest neighbor search (KNN) method for blocking to get the candidate matching set C. Finally, step 4 uses the labeled data to train the matcher, and the trained matcher is applied for inference.

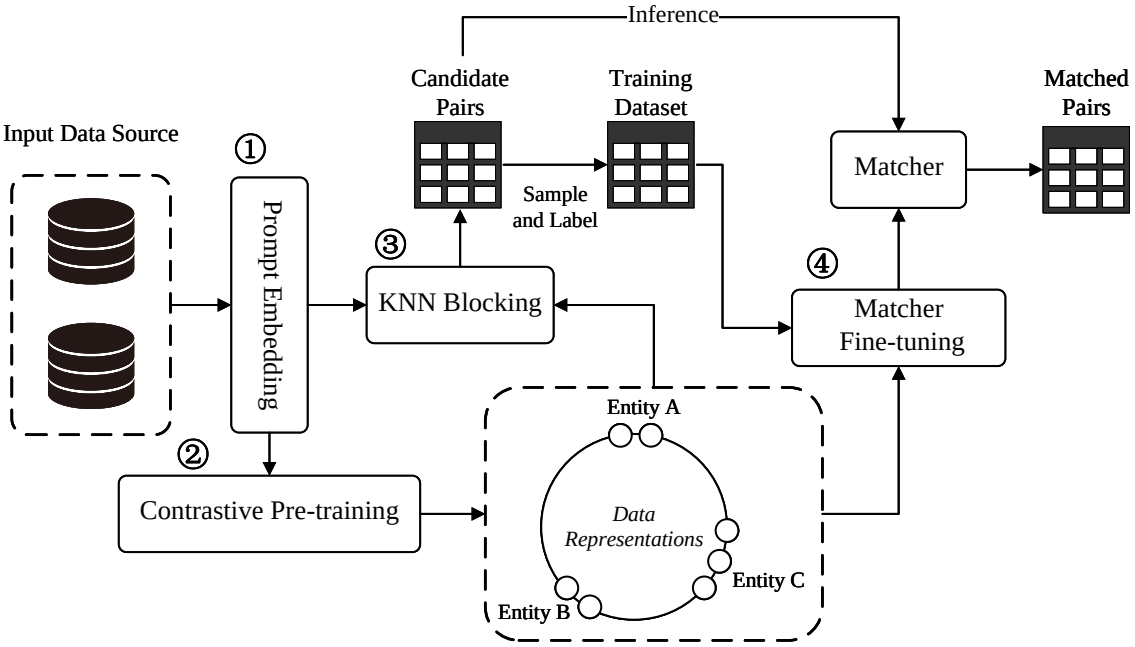

Fig. 2. The overall framework of PromptER.

4.2 Prompt Based Record Embedding

Motivated by [24], we propose a prompt based method to obtain record embedding. By reformulating the sentence embedding task as the mask language task, we can effectively

use original BERT layers by leveraging the large-scale knowledge and avoid [CLS] embedding bias. We explore the way to implement prompt based record embedding by solving the following two problem: 1) how to represent records with the prompt, and 2) how to find a proper prompt for record embedding.

Represent Record with the Prompt: By constructing prompt templates, we convert the original input $x_{\text{origin}} := \textit{serialize}(r)$ into $x_{\text{prompt}} := \textit{serialize}(r)\ \textit{is}$ [MASK]. Then x_{prompt} is feed to PLM to get the token sequence embedding:

$$PLM(x_{\text{prompt}}) = \{\mathrm{h}_{[\text{CLS}]}, \underbrace{\mathrm{h}_{w_1}, \ldots, \mathrm{h}_{w_n}}_{\textit{serialize}(r)}, \mathrm{h}_{is}, \mathrm{h}_{[\text{MASK}]}\} \tag{1}$$

The hidden vector of [MASK] token is chosen as record representation $\mathrm{h_r}$:

$$\mathrm{h_r} = \mathrm{h}_{[\text{MASK}]} \tag{2}$$

Prompt Search: The major challenge for prompt-based embedding is to find templates, because different templates have a huge impact on the performance of the model. To obtain proper template, we propose a two-stage template construction method. The first stage is manual template construction and the second stage is continuous tuning of the manual template.

In the first stage, as show in Figure 3, we divide the template into two parts: 1) relationship tokens, which indicates the relationship between the serialized record x_{origin} and [MASK] token and 2) prefix tokens, which modifies x_{origin}. Then, we greedily search for the combination of relational and prefix tokens, which achieve the highest F1-score on the validation set, as the final template.

Number	Relationship tokens
R1	serialize(e) [MASK]
R2	serialize(e) is [MASK]
R3	serialize(e) means [MASK]

Number	Prefix tokens
P1	This serialize(e)
P2	This record serialize(e)
P3	This tuple serialize(e)

Combination	Final prompt templates	F1-score
P1,R1	This serialize(e) [MASK]	0.78
P2,R2	This record serialize(e) is [MASK]	0.93
P3,R3	This record serialize(e) means [MASK]	0.91
…	…	…

Fig. 3. Greedy searching for prompt templates combination. The F1-score is the result on the validation set of the SEMI-REL dataset.

In the second stage, following the setting of OptiPrompt [25], the continuous prompt is initialized with the manually constructed prompt, where the static embeddings of the continuous prompt are equivalent to that of the manual template. In the next step, we keep all other parameters of the model fixed and optimize the continuous template through contrastive pre-training. In this process, only the static embedding of the continuous template is updated. Then the trained continuous template is applied for downstream tasks.

4.3 Contrastive Pre-training

The contrastive pre-training is to learn the similarity-aware record representations of all entity entries from the two input tables. When the model already captures entity similarity, it can be used for downstream blocking and matching tasks. The overall architecture of contrastive pre-training is shown in Figure 4.

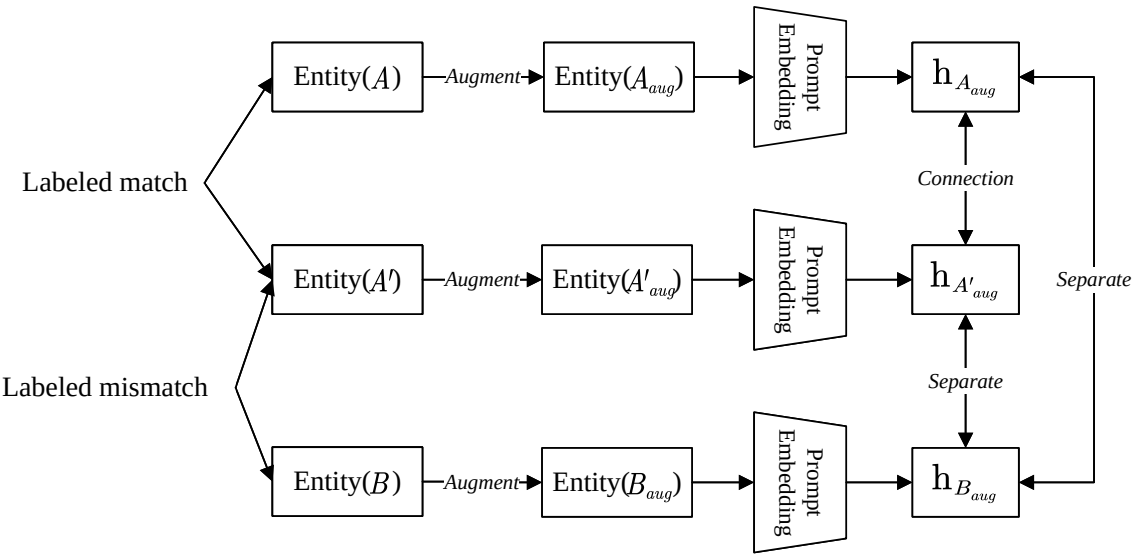

Fig. 4. Contrastive pre-training learns similarity-aware representations by connecting matching records (e.g., $\mathbf{h}_{A_{aug}}$ and $\mathbf{h}_{A\prime_{aug}}$) and separating mismatching records (e.g., $\mathbf{h}_{A_{aug}}$ and $\mathbf{h}_{B_{aug}}$).

In the first step, we use the sampling strategy depicted in Section 4.3 to select a batch B_i of record pairs $(r_i, r_{i\prime})$, where $i \in [1, |B_i|]$ and $|B_i|$ is the batch size. Following this, we apply the data augmentation method proposed in Section 4.3 to each record pair in batch B_i. The prompt-based embedding method maps each augmented record A_{aug} to a representation: $\mathrm{h}_{aug} = prompt\,(A_{aug})$. Afterward, the embedding model parameters are updated by optimizing the contrastive loss:

$$\mathcal{L}_{contrast} = \frac{1}{|B_i|} \sum_{i \in B_i} \frac{1}{|P_i|} \sum_{p \in P_i} -\log \frac{\exp\left(sim\left(h_i \cdot h_p\right)/\tau\right)}{\sum_{b \in B_i \setminus h_i} \exp\left(sim\left(h_i \cdot h_b\right)/\tau\right)} \tag{3}$$

Here, sim is cosine similarity function. τ is the temperature hyperparameter in the range $(0, 1]$. For each record, the numerator calculates the similarity between the current record and its matching record, while the denominator computes the similarity between the current record and all other records in the same batch. The contrastive loss is the sum of all records calculation results within this batch.

Sampling Strategy: Supervised contrastive learning requires that all records that refer to the same real-world entity share the same identifier. Typically, entity matching datasets do not provide identifiers for records, but rather label a subset of record pairs from different sources as matching or mismatching. For contrastive pre-training, we need to label such identifier for each record. For this labelling, we build a correspondence graph similar with SupCon [12]. The vertices of this graph are the records and the edges connect matching records. We then label a unique identifier to each connected component of the vertices in the graph so that matching records share the same identifier.

The labelling approach mentioned above can result in inter-label noise, as we only have knowledge of a subset of matches between sources and some records that are actually matches may be assigned different labels. In the training process, this will result in treating these unlabeled matching records as non-matching records if they appear in the same batch. This inter-source noise can seriously affect the quality of the learned record representations as the matched records are not embedded into a close-by location.

To alleviate this problem, source-aware sampling is applied [12]. Rather than generating a single training dataset that includes all the records and their labels, we create a separate training dataset for each input data source A or B. Each training dataset contains all records from the corresponding source and only records from the other source which share the same identifier with a record in the current source. To illustrate, the training dataset A includes all the records from dataset A and all the records from dataset B that share an identifier with a record in dataset A.

Once the training dataset for each data source is built, in each sampling step, we randomly choose a dataset to sample records into a batch. Within each batch, records with the same identifier form matching pairs, while records with different identifiers form mismatching pairs. When data sources A and B themselves do not contain any duplicates, this sampling process can completely eliminate inter-source noise.

Prompt Based Data Augmentation: In contrastive learning, data augmentation can provide diverse representations of the same entity, thereby enhancing the model's robustness. Traditional data augmentation methods, such as word deletion, reordering, and substitution, distort the original content of the record, which make the assigned label no longer correct. For example, dropping the "ti" from the string "RTX 3090 ti", changes the original entity to a different one and the raw label may be incorrect.

To address this issue, we propose a simple yet effective data augmentation method. We use different templates to represent the description of the same entity as different points of view without changing the record itself. For example, one record in a pair is wrapped using template [this entity: $serialize(r)$ is [MASK]] to obtain representation $\mathrm{h_r} = \mathrm{h_{[MASK]}}$, while the other record is wrapped using template [information of this entity is $serialize(r')$, this entity is [MASK]] to obtain representation $\mathrm{h_{r'}} = \mathrm{h_{[MASK]}}$. Then, during the training process, the two representations h_r and $\mathrm{h}_{r'}$ are pulling closer in the embedding space.

Evaluation Method: The contrastive pre-training continuous pulling together representations of the same entity and pushing apart representations of different entities. However, it is unknown to us what distance is appropriate for downstream matching and blocking tasks. In other words, we do not know when to stop contrastive pre-training. SupCon [12] and Block-SCL [26] treat the contrastive pre-training epochs as a hyperparameter, and determine it through manual setting or grid search. This approach requires additional searching and training time and is susceptible to overfitting. To solve this problem, we propose a validation method to determine the optimal number of training steps. A detailed explanation of this method is provided below:

When training the embedding model for the predetermined number of steps, we use it to convert the labeled record pairs in the train set into representations: (h_1, h_2), and combine them as $(h_1, h_2, |h_1 - h_2|, h_1 * h_2)$. Afterwards, the combined representations are applied to train a binary classifier using cross-entropy loss. Next, we follow the same procedures as described above to encode all record pairs in the validation set and utilize the trained classifier to make matching and mis-matching decisions. The F1-score on the

validation set is selected as the evaluation metric. Finally, an early stopping strategy is employed, where pre-training is stopped when the F1-score on the validation set ceases to improve.

4.4 Entity Blocking and Entity Matching

Entity Blocking: After contrastive pre-training, the embedding model is able to capture similarity among records. The K-nearest neighbor (KNN) search method can then be utilized to generate a candidate set of matching record pairs. We use the embedding model to convert all records in data sources $A = \{a_1, a_2, \cdots a_n\}$ and $B = \{b_1, b_2, \cdots b_n\}$ to representations set $H_A = \{h_{a_1}, h_{a_2}, \cdots h_{a_n}\}$ and $H_B = \{h_{b_1}, h_{b_2}, \cdots h_{b_n}\}$. For each record representation h_{a_i} in H_A, we calculate the cosine similarity with all representations in H_B. Then, we select the top K records in data source B with the highest cosine similarity to form candidate matching pairs. Those K pairs (a_i, b_j) where $a_i \in A, b_j \in B$ will be included in the candidate set.

Entity Matching: In the matching stage, we add a classification head on top of the embedding model to return a binary label indicating whether the pair of records is a match or a mismatch. The classification head consists of a dropout layer and a linear layer, and takes the combination of the records pair representations $(h_1, h_2, |h_1 - h_2|, h_1 * h_2)$ as input. During training, the parameters of the embedding model are frozen, and the classification head is tuned using the binary cross-entropy loss.

5 Experiments

In this section, we evaluate the performance of PromptER on generalized entity resolution dataset Machamp [8]. Compared to the state-of-the-art ER methods, PromptER achieves significant performance improvements on both the blocking and matching tasks using a single model. We first discuss the experiment setup in Section 5.1, then present the blocking and matching results in Section 5.2. Finally, we present an ablation study to demonstrate the effectiveness of the proposed prompt-based embedding in several aspects in Section 5.3.

5.1 Experimental Setup

Datasets: We use seven real-world benchmark datasets with different structures from Machamp [8]. The statistics of datasets are summarized in Table 1. Each dataset contains two data sources, which can be in one of the three formats: relational data (REL), semi-structured data (SEMI) and textual data (TEXT). When they are of the same format, they can have a homogeneous (HOMO) or heterogeneous (HETER) schema. The labeled ground truth dataset provides matching and mismatching record pairs from two data sources, and is divided into training, validation, and testing sets in a ratio of 3:1:1.

Baseline Methods: We compare PromptER with six SOTA deep learning based ER methods. The models used for comparison can be divided into two categories. One is independent representation models, including DeepER, SentenceBert and SupCon. The other is interdependent representation models, including DeepMatcher and Ditto. The independent representation models can be used for blocking and matching at the same

Table 1. Statistics of the Machamp Datasets. "#row" denotes the number of records, and "#attr" denotes the average attribute number. "%POS" is the matching record pairs rate in the train, valid and test dataset.

Datasets	Left Table		Right Table		Labeled Examples			
	#row	#attr	#row	#attr	Train	Valid	Test	%POS
REL-HETER	534	6.00	332	7.00	567	190	189	11.63%
SEMI-HOMO	2616	8.65	64263	7.34	17223	5742	5742	18.63%
SEMI-HETER	22133	12.28	23264	12.03	1240	414	414	38.20%
SEMI-REL	29180	8.00	32823	13.81	1309	437	437	41.64%
SEMI-TEXT-w	9234	10.00	20897	1.00	12538	4180	4179	11.80%
SEMI-TEXT-c	20897	10.00	20897	1.00	12538	4180	4179	14.07%
REL-TEXT	2616	1.00	2295	6.00	7417	2473	2473	17.96%

time, while interdependent representation models require an additional blocking step. The details of the baseline models are described as follows:

DeepER [22] first uses a bi-directional Long Short-Term Memory (LSTM) model to convert records into representations. Then, it calculates the similarity vectors between representations using methods such as cosine similarity, vector difference (subtracting), and Hadamard product (multiplying). Finally, a multi-layer neural network is used to perform binary classification based on the similarity vectors.

SentenceBert [27] proposes a siamese architecture for PLMs to perform sentence matching tasks. It first encodes two sentences separately using the same encoder and then concatenates the two representations and a vector generated from an element-wise operation between them as the output for prediction.

SupCon [12] also leverages PLMs for record representations and incorporates a contrastive pre-training process. SupCon has demonstrated the effectiveness of contrastive pre-training in entity resolution tasks and has achieved the best results among many entity resolution datasets. Therefore, we consider it a strong baseline for comparison.

DeepMatcher [6] utilizes Siamese recurrent neural (RNN) networks as the basic structure to aggregate the attribute values, and then uses the attention mechanism to compare the embeddings of the attributes. After that, it uses a two layer fully-connected ReLU HighwayNet followed by a softmax layer to implement the classifier.

Ditto [2] combines the PLMs with data augmentation techniques for entity matching. By leveraging the power of PLMs and data augmentation methods, it has achieved leading performance in many entity resolution benchmark tasks.

Implementation Details: We implement PromptER in PyTorch and the Transformers library. We use Roberta-base as the backbone structure for matching and blocking result reports. In all experiments for PromptER, the max sequence length is set to 256, the learning rate is set to 1e-5, and the batch size is set to 24. For contrastive pre-training, we train using AdamW optimizer for 20 epochs, while applying the proposed evaluation strategy at each epoch to determine when to stop the pre-training process. In the training of classifier, the epoch is set to 10 and the model with the highest F1-score in validation dataset is selected for evaluation. Each model is trained three times and we report the average results.

Evaluation Metrics: For entity macthing evaluation, we employ three widely-used classification metrics, namely, precision (P), recall (R), F1-score (F), and report the final result on the test dataset.

For entity blocking evaluation, following [21], we use the recall (R) and candidate set size ratio (CSSR) metrics. Assuming C be the candidate set as the output of blocking on two tables A and B, and let G be the set of true matches between A and B. Then recall is measured as $|G \cap C|/|G|$, and candidate set size ratio is measured as $|C|/|A \times B|$. As we use the K nearest neighbors blocking method, the CSSR is proportional to K. For convenience, we report the value of K instead of CSSR. Ideally, we want high recall and small K value.

5.2 Entity Matching Results

The results of comparing with state-of-the-art methods are shown in Table 2. In 5 of 7 datasets, our method achieves the best results. On the other 2 datasets, our method also achieves the second-best results. Among all the independent representation models, our model is consistently the best. Except for the SEMI-TEXT-w and REL-TEXT datasets, our proposed method outperforms interdependent representation models such as DeepMatcher and Ditto.

Table 2. F1-score results on the test set of each dataset. The results marked with * are taken from [8]. Bold denotes the best performing results and underlined the second best.

Datasets	DeepER*			SentenceBert			SupCon			DeepMatcher*			Ditto*			PromptER		
	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F
REL-HETER	100	73.3	87.2	75.0	95.5	84.0	100	81.8	90.0	100	87.9	93.6	100	100	100	100	100	**100**
SEMI-HOMO	89.4	85.8	87.5	93.6	92.6	93.1	93.8	90.5	92.1	89.0	83.2	86.1	94.7	91.6	93.1	89.5	97.5	**93.3**
SEMI-HETER	61.7	18.2	28.2	29.3	27.0	28.1	55.5	69.8	61.8	35.8	24.5	29.1	84.6	48.4	61.6	97.3	46.5	**64.0**
SEMI-REL	49.0	39.2	43.6	54.4	63.9	58.8	77.2	81.4	79.3	50.9	64.1	56.7	95.8	86.9	91.1	88.8	95.6	**92.1**
SEMI-TEXT-c	75.6	31.1	44.2	66.7	58.2	62.2	84.3	75.5	79.7	80.2	29.1	42.7	82.2	81.3	81.8	81.8	76.9	**82.1**
SEMI-TEXT-w	75.9	26.1	38.8	35.5	28.9	31.9	33.7	59.7	43.1	80.2	29.1	42.7	63.6	66.3	**64.9**	46.5	47.4	46.9
REL-TEXT	72.7	41.6	52.9	57.6	44.1	50.0	66.4	42.3	51.7	78.4	40.4	53.4	65.6	60.1	**62.7**	67	49.8	57.1

For REL-HETER dataset, both sides of the data consist of relationship tables, making the data relatively clean and containing explicit information for matching decisions. As a result, all models perform well on this dataset. DeepMatcher and Ditto, as interdependent representation models, perform better than independent representation models due to their fine-grained comparison of characters between tuples. However, our proposed PromptER can bridge this gap and achieve 100% F1-score.

For SEMI-HOMO and SEMI-HETER datasets, the two contrastive learning based models, SupCon and PromptER, achieve the best results. Our proposed prompt embedding and prompt augmentation methods reduce the bias of pre-trained language models in record representation, resulting in an improvement of 1.2% and 3.2% F1-score respectively over the SupCon baseline.

For SEMI-REL and SEMI-TEXT-c datasets, Ditto performs better than SupCon, especially on the SEMI-REL dataset where Ditto leads by 11.8% in F1 score. While PromptER outperforms Ditto on these two datasets, demonstrating that PromptER performs more stable than SupCon on various datasets.

For the SEMI-TEXT-w and REL-TEXT datasets, Ditto achieves the best performance, while SupCon and PromptER perform relatively poorly. This is because the input records pairs contain long texts, which are relatively "dirty" as they contain a lot of information that is irrelevant to the matching decision. This information also participates in the representation process and dilutes the record embedding. But SupCon and PromptER

heavily rely on the comparison between record embeddings, which results in a decrease in performance.

5.3 Entity Blocking Results

We compare the blocking performance of PromptER with two other PLM based blocking methods including SentenceBert and SupCon, and the results are shown in Figure 5. The x-axis is the K value which represents the candidate set size ratio while the y-axis shows the recall. We do not report the results on the REL-HETER dataset because the task is relatively simple, and when K equals 1, the recall of all the three methods already reaches 100%.

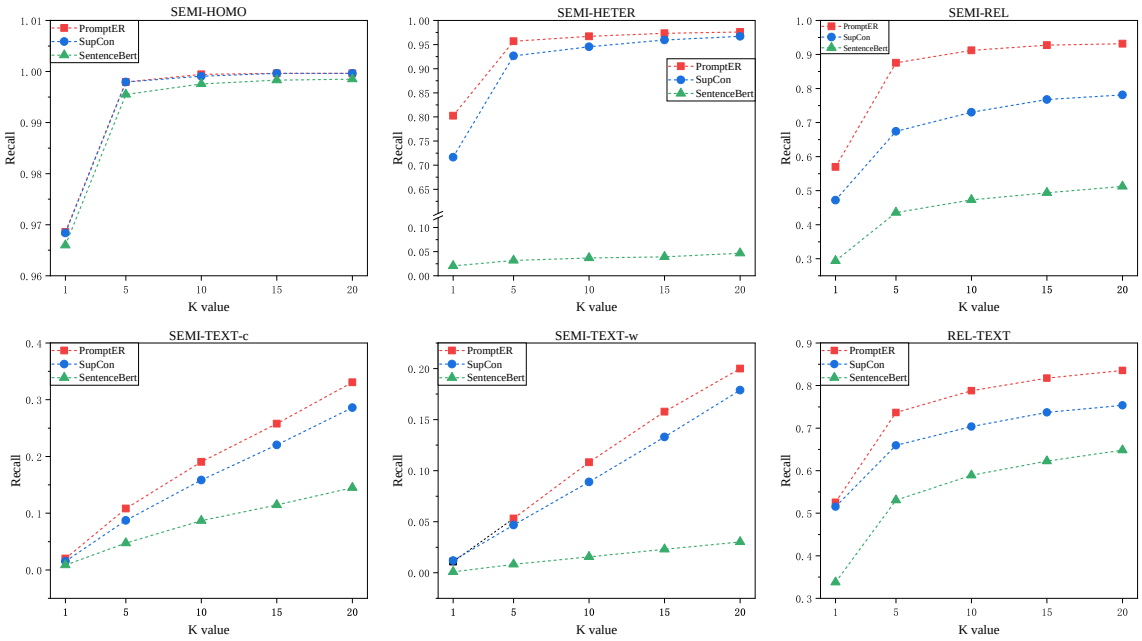

Fig. 5. The blocking performance comparison of PromptER, SupCon and SentenceBert.

All the plots in Figure 5 show that as K increases, both recall and candidate set size ratio increase. Ideally, we want high recall and small candidate set ratio. From this perspective, PromptER outperforms SupCon and SentenceBert on all six datasets, as it achieves higher recall with a smaller K value. For the SEMI-HOMO, SEMI-HETER, and SEMI-REL datasets, PromptER demonstrates strong blocking ability by achieving a recall rate over 90% with a small K value of 5. Especially on the SEMI-REL dataset, PromptER surpasses SupCon by 15% to 20% in recall rate from k=5 to k=20. Although in SEMI-TEXT-c/w and REL-TEXT datasets, suffering from the same "dirty" text problem, the PromptER still performs better than SupCon and SentenceBert. This result demonstrates that PromptER is more robustness to noise input.

5.4 Ablation Studies

To further demonstrate the effectiveness of our proposed prompt embedding method, we conduct ablation experiments between PromptER and baseline models. These experiments include evaluating the performance of prompt embedding on different PLMs, analyzing the impact of prompt embedding on the distribution of record representations and assessing the influence of prompt embedding on the convergence speed.

The performance of prompt embedding on different PLMs: In this experiment, we use a baseline model that encodes records using [CLS] embedding instead of the proposed prompt embedding method. Except for the embedding method, the baseline model has the same model structure, sampling method, and hyperparameters as PromptER. Especially, to demonstrate the ability of prompt embedding to different PLMs, we chose Bert and Roberta separately as the backbone networks and report the F1-value improvement.

Table 3. The influence of Prompt embedding for different PLMs. The results are F1-score on the test set of each dataset.

Datasets	Bert-base-uncased		Roberta-base	
	Baseline	PromptER	Baseline	PromptER
REL-HETER	52.4	87.2(+34.8)	89.3	100(+10.7)
SEMI-HOMO	89.2	92.4(+3.2)	92.2	93.3(+1.1)
SEMI-HETER	24.6	59.0(+34.4)	62.1	64.0(+1.9)
SEMI-REL	82.7	86.0(+3.3)	81.2	92.1(+10.9)
SEMI-TEXT-c	68.2	77.6(+9.4)	79.6	82.1(+2.5)
SEMI-TEXT-w	37.3	39.7(+2.4)	42.8	46.9(+4.1)
REL-TEXT	46.4	57.5(+11.1)	50.3	57.1(+6.8)

From the results shown in Table 3, it can be observed that prompt embedding significantly improves the performance of both BERT-based and RoBERTa-based models. The improvement range for BERT-based models is between 2.4% to 34.8%, with an average improvement of 14.2%. For RoBERTa-based models, the improvement ranges between 1.1% to 10.9%, with an average improvement of 5.4%. The experiment results show that our proposed prompt embedding can obtain high quality record embedding and effectively improve the performance of PLMs.

The influence of prompt embedding for records similarity distribution: In order to further investigate the influence of prompt embedding on record representation, we plot the cosine similarity distribution of record embedding for SentenceBert, SupCon and PromptER. As is shown in Figure 6, The representation of SentenceBert shows high cosine similarity between both matching and mismatching record pairs. SupCon reduces the similarity between mismatching record pairs but also sacrifices the similarity between matching records. PromptER, on the other hand, decreases the similarity between records of different entities while maintaining high similarity between records of the same entity. This demonstrates that prompt embedding is better to distinguish between matching and mismatching records, resulting in better performance in entity matching and entity blocking tasks.

The influence of prompt embedding on convergence speed: For the training efficiency, we compare the number of training steps required for PromptER and baseline to reach convergence. The baseline model has the same model structure, sampling method, and hyperparameters as PromptER, but directly using the [CLS] embedding. According to Figure 7, PromptER converges much faster than baseline. On the SEMI-HETER, SEMI-REL, and REL-HETER datasets, PromptER reaches convergence after only 100 to 200 training steps, while baseline requires over 500 steps. On other datasets including REL-TEXT, SEMI-HOMO and SEMI-TEXT, PromptER also requires significantly fewer training steps to reach convergence compared to baseline. The above results indicate that our model requires less training time and thus reduce the time cost of model training.

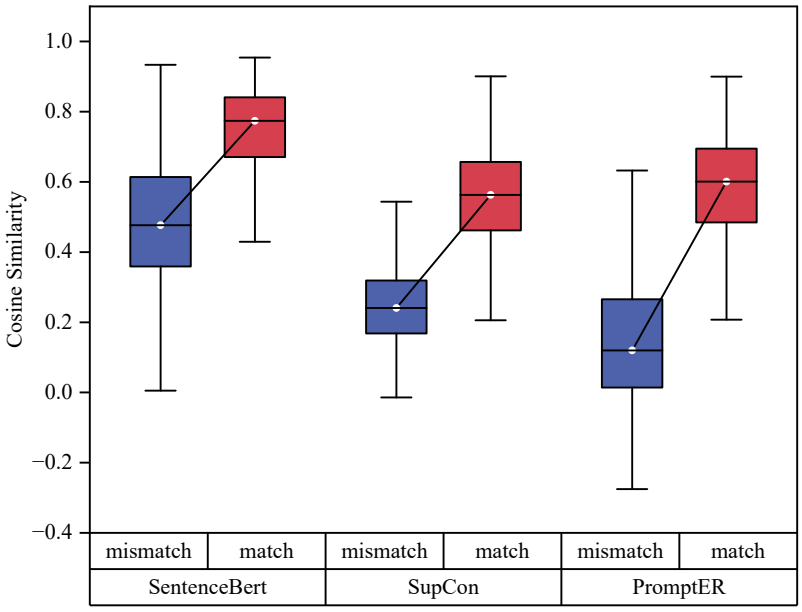

Fig. 6. The distribution of Cosine Similarity between matching and mismatching record pairs on the SEMI-REL dataset for SentenceBert, SupCon, and PromptER.

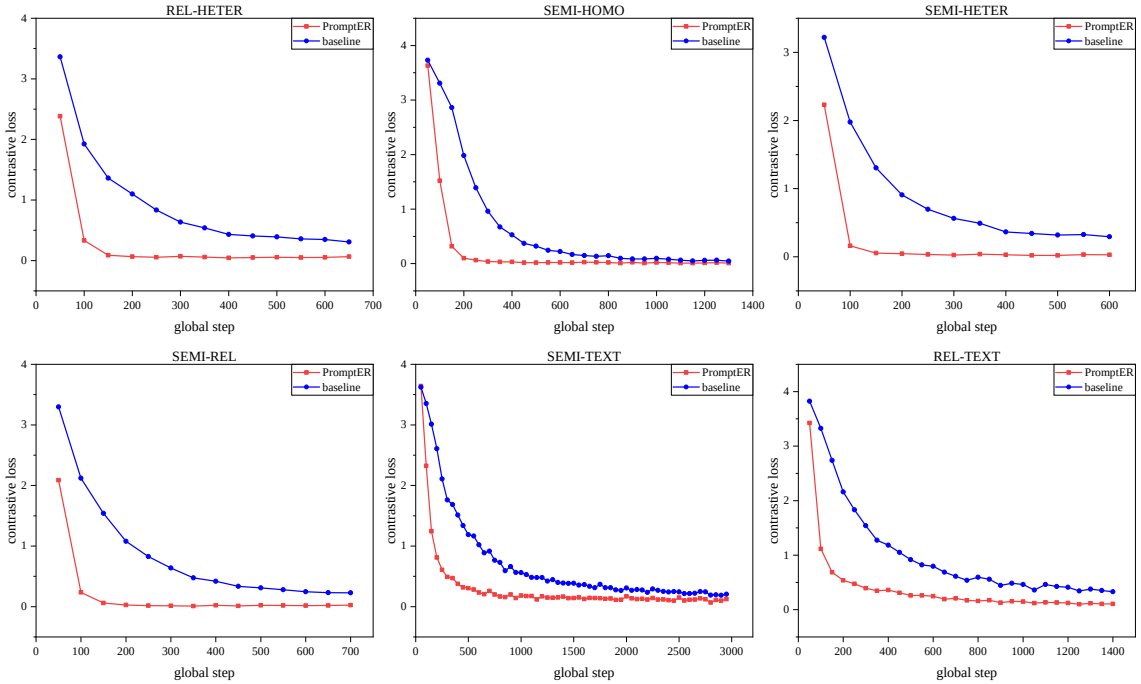

Fig. 7. The convergency speed comparison.

6 Conclusion

In this paper, we tackle the efficiency and effectiveness conflict problem of generalized entity resolution through our proposed PromptER model. PromptER introduces prompt embedding for records representation, which can better utilize original PLM layers by leveraging the large-scale pretraining knowledge and avoid embedding bias. To further improve model performance, we design the prompt-based data augmentation method to avoid the damage to the original input caused by traditional data augmentation methods. Additionally, we propose a new evaluation method for contrastive pre-training that eliminates the extra time cost and overfitting issues associated with manually specifying pre-training steps.

In the experiments section, we compare PromptER with the current state-of-the-art entity matching and blocking methods. In most cases, PromptER outperforms these methods, achieving superior results. The ablation experiment demonstrates the effectiveness of prompt embedding for various PLMs. It also shows that prompt embedding is able to learn more discriminative records representation and significantly reduce the training time for convergency.

In future research, we plan to explore more effective methods for automated prompt template generation, such as using T5 to generate templates. Some existing research has shown that the selection of negative samples can have a significant impact on the effectiveness of contrastive learning. Therefore, in future work, we will also focus on the selection of positive and negative samples for each batch.

References

1. Vassilis Christophides, Vasilis Efthymiou, Themis Palpanas, George Papadakis, and Kostas Stefanidis. An overview of end-to-end entity resolution for big data. ACM Comput. Surv., 53(6), dec 2020.
2. Yuliang Li, Jinfeng Li, Yoshihiko Suhara, AnHai Doan, and Wang-Chiew Tan. Deep entity matching with pre-trained language models. Proc. VLDB Endow., 14(1):50–60, sep 2020.
3. Chen Ye, Shihao Jiang, Hua Zhang, Yifan Wu, Jiankai Shi, Hongzhi Wang, and Guojun Dai. Jointmatcher: Numerically-aware entity matching using pre-trained language models with attention concentration. Know.-Based Syst., 251(C), sep 2022.
4. Ralph Peeters and Christian Bizer. Dual-objective fine-tuning of bert for entity matching. Proc. VLDB Endow., 14(10):1913–1921, jun 2021.
5. Bing Li, Yukai Miao, Yaoshu Wang, Yifang Sun, and Wei Wang. Improving the efficiency and effectiveness for bert-based entity resolution. Proceedings of the AAAI Conference on Artificial Intelligence, 35(15):13226–13233, May 2021.
6. 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. SIGMOD '18, page 19–34, New York, NY, USA, 2018. Association for Computing Machinery.
7. Cheng Fu, Xianpei Han, Le Sun, Bo Chen, Wei Zhang, Suhui Wu, and Hao Kong. End-to-end multiperspective matching for entity resolution. In Sarit Kraus, editor, Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019, pages 4961–4967. ijcai.org, 2019.
8. Jin Wang, Yuliang Li, and Wataru Hirota. Machamp: A generalized entity matching benchmark. In Proceedings of the 30th ACM International Conference on Information and Knowledge Management, CIKM '21, page 4633–4642, New York, NY, USA, 2021. Association for Computing Machinery.
9. Nils Barlaug and Jon Atle Gulla. Neural networks for entity matching: A survey. ACM Trans. Knowl. Discov. Data, 15(3), apr 2021.
10. 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., 16(2):369–378, oct 2022.
11. Ursin Brunner and Kurt Stockinger. Entity matching with transformer architectures - A step forward in data integration. In Proceedings of the 23rd International Conference on Extending Database Technology, EDBT 2020, Copenhagen, Denmark, March 30 - April 02, 2020, pages 463–473. OpenProceedings.org, 2020.

12. Ralph Peeters and Christian Bizer. Supervised contrastive learning for product matching. In Companion Proceedings of the Web Conference 2022, WWW '22, page 248–251, New York, NY, USA, 2022. Association for Computing Machinery.
13. 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, WSDM '20, page 744–752, New York, NY, USA, 2020. Association for Computing Machinery.
14. Jiacheng Huang, Wei Hu, Zhifeng Bao, Qijin Chen, and Yuzhong Qu. Deep entity matching with adversarial active learning. VLDB J., 32(1):229–255, 2023.
15. Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. ACM Comput. Surv., 55(9), jan 2023.
16. Ning Ding, Shengding Hu, Weilin Zhao, Yulin Chen, Zhiyuan Liu, Haitao Zheng, and Maosong Sun. OpenPrompt: An open-source framework for prompt-learning. In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: System Demonstrations, pages 105–113, Dublin, Ireland, May 2022. Association for Computational Linguistics.
17. Bohan Li, Hao Zhou, Junxian He, Mingxuan Wang, Yiming Yang, and Lei Li. On the sentence embeddings from pre-trained language models. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 9119–9130, Online, November 2020. Association for Computational Linguistics.
18. Tianyu Gao, Xingcheng Yao, and Danqi Chen. SimCSE: Simple contrastive learning of sentence embeddings. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 6894–6910, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.
19. George Papadakis, Ekaterini Ioannou, Emanouil Thanos, and Themis Palpanas. The Four Generations of Entity Resolution. Synthesis Lectures on Data Management. Morgan and Claypool Publishers, 2021.
20. Delaram Javdani, Hossein Rahmani, Milad Allahgholi, and Fatemeh Karimkhani. Deepblock: A novel blocking approach for entity resolution using deep learning. In 2019 5th International Conference on Web Research (ICWR), pages 41–44, 2019.
21. Saravanan Thirumuruganathan, Han Li, Nan Tang, Mourad Ouzzani, Yash Govind, Derek Paulsen, Glenn Fung, and AnHai Doan. Deep learning for blocking in entity matching: A design space exploration. Proc. VLDB Endow., 14(11):2459–2472, jul 2021.
22. Muhammad Ebraheem, Saravanan Thirumuruganathan, Shafiq Joty, Mourad Ouzzani, and Nan Tang. Distributed representations of tuples for entity resolution. Proc. VLDB Endow., 11(11):1454-1467, jul 2018.
23. Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In Proceedings of the 37th International Conference on Machine Learning, ICML'20. JMLR.org, 2020.
24. Ting Jiang, Jian Jiao, Shaohan Huang, Zihan Zhang, Deqing Wang, Fuzhen Zhuang, Furu Wei, Haizhen Huang, Denvy Deng, and Qi Zhang. PromptBERT: Improving BERT sentence embeddings with prompts. In Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pages 8826–8837, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics.
25. Zexuan Zhong, Dan Friedman, and Danqi Chen. Factual probing is [MASK]: learning vs. learning to recall. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021, pages 5017–5033. Association for Computational Linguistics, 2021.
26. Mario Almagro, David Jim´enez, Diego Ortego, Emilio J. Almaz´an, and Eva Martınez. Block-scl: Blocking matters for supervised contrastive learning in product matching. CoRR, abs/2207.02008, 2022.
27. Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019, pages 3980–3990. Association for Computational Linguistics, 2019.