JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 39, 423-438 (2023)
DOI: 10.6688/JISE.202303_39(2).0011

Entity Matching Based on Attribute-Aware and Multi-Perspective Similarity Measurement*

XIN XING AND NING WANG+
School of Computer and Information Technology
Beijing Jiaotong University
Beijing, 100044 P.R. China
E-mail: {19120416; nwang}@bjtu.edu.cn

Entity matching (EM) identifies tuples from different data sources that refer to the same real-world entity. One of the main challenges of EM is attribute heterogeneity, that is, there are many different types of attributes in an entity. Present researches focus on using rules or neural networks to select similarity measures for different types of attributes. However, they select only one specific similarity measure for each attribute but ignore matching information from many other aspects. In addition, existing methods neglect the fact that different attributes have different contributions to final matching decision, and do not consider the influence of dirty data on matching results. In this paper, we propose an entity matching method based on attribute-aware and multi-perspective similarity measurement. Firstly, we propose a multi-perspective similarity measurement framework based on pre-trained language model DeBERTa to achieve the comprehensive multi-perspective similarity computation, which will capture the matching information from multiple perspectives such as literal, size and semantics. Secondly, we introduce an attribute attention mechanism to aggregate matching evidences from all aligned attributes according to the importance of each attribute for final matching decision. Finally, we use cross-attribute comparison to solve dirty data problems such as swap errors, and we further improve our model's matching capability through injecting external entity knowledge. Experimental results show that our framework for entity matching outperforms state-of-the-art methods on multiple real-world data sets.

Keywords: entity matching, similarity measurement, data integration, deep learning, natural language processing

1. INTRODUCTION

Entity matching (EM), also known as entity resolution (ER) or duplicate record detection, aims to identify tuples from different data sources that refer to the same real-world entity. Considering two tables in Fig. 1, tuple 1269 and tuple 1433 correspondingly from *Amazon* and *Google* would be resolved as the same entity because they refer to the same real-world product, and the same goes for tuple 22 and tuple 1435. Obviously, tuple 1269

Received October 28, 2021; revised January 2, 2022; accepted April 11, 2022.
Communicated by Raymond Wong.
+ Corresponding author.
* This work is supported by the National Key R&D Program of China (2018YFC0809800).

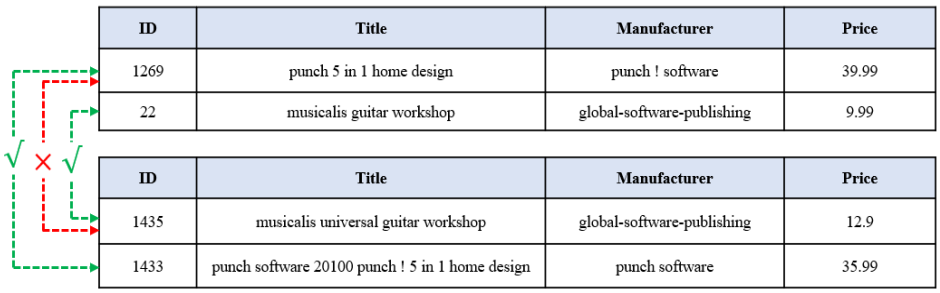

Fig. 1. Two entity tuples correspondingly from *Amazon* and *Google* refer to the same product.

and tuple 1435 is no-matching. As a fundamental essence for data cleaning and data integration, entity matching can greatly improve data quality, and thus facilitate downstream data analysis and decision making [1, 2]. It has been widely applied in knowledge graph construction, e-commerce, *etc*.

One main characteristics of entity tuples for EM is that attributes are heterogenous (*i.e*., they are of different data types, for example in Fig. 1, numeric attribute: **Price**, string attribute: **Manufacturer**, textual attribute: **Title**). Given two entity tuples, structured EM approaches first align the attributes of the two tables and compute the similarity of values between aligned attributes, then aggregate similarity results of all aligned attributes to make the final decision. Due to the heterogeneity of entity attributes, many similarity measures have been proposed for computing similarity of attribute values, including deep learning-based similarities for textual attributes [3], string similarities for string attributes [4], and numeric similarities for number attributes [5], *etc*.

Considering the heterogeneity of attributes and the diversity of similarity measures, one main challenge of EM is selecting appropriate similarity measure for each attribute to compute its similarity. Early researches used manual or heuristic methods to select appropriate similarity measures for different attributes [4, 6, 7], which are usually hard to be generalized to other EM tasks.

The latest work MPM uses neural networks to select optimal similarity measures for different attributes in an end-to-end way [8]. However, MPM selects only one specific similarity measure for each aligned attribute pairs, but ignores the matching information from many other perspectives. For example, besides literal meaning, an attribute of string type also has its length and its semantics, which makes up multiple perspectives of the attribute. When implementing the gate function for selection, the gate weight in MPM is learned by randomly initializing a vector as the attribute, completely neglecting the difference between attributes. Furthermore, MPM aggregates the matching evidences from all aligned attributes by concatenation operation, ignoring the fact that different attributes will bring different contribution to EM decision-making. In addition, when there exists dirty data problem such as swap errors, it is difficult for MPM to capture any matching information.

Considering the limitations of above methods, we propose an entity matching method based on attribute-aware and multi-perspective similarity measurement. Firstly, we propose a multi-perspective similarity measurement framework based on pre-trained

language model DeBERTa to achieve the comprehensive multi-perspective similarity computation, which will capture the matching information from multiple perspectives such as literal, size and semantics. Different from randomly initializing a vector to learn the gate weight in MPM, our model learns the weight based on the similarity results from multiple measures. Secondly, we introduce an attribute attention mechanism to aggregate matching evidences from all aligned attributes according to the importance of each attribute for final matching decision. Finally, we use cross-attribute comparison to solve dirty data problems such as swap errors, and we further improve our model's matching capability through injecting external entity knowledge. Experimental results show that our framework outperforms state-of-the-art methods on multiple real-world data sets.

2. RELATED WORK

Entity matching has been extensively studied since 1950s [9], thus a variety of methods for solving the EM problem have been proposed [10, 11]. Generally, these methods can be roughly divided into three categories: rule-based, crowdsourcing-based and machine learning-based.

Early methods were mainly rule-based or crowdsourcing-based. The rule-based methods determine whether tuples match or not through the rules established by experts or the rules automatically learned from known examples [4, 6, 12–14]. The crowdsourcing-based methods mainly rely on crowdsourcing workers to solve the entity matching problem by finishing tasks on crowdsourcing platforms [15–17].

At present, it is popular to use machine learning methods to perform entity matching. According to the machine learning model which is used in EM methods, we can divide this kind of methods into traditional methods and deep learning-based methods. Traditional machine learning based methods treat entity matching as a binary classification problem, which feed manually extracted features to classifiers (such as SVM, Naïve Bayes) to determine whether tuples match or not [18]. Magellan is a representative work in this direction, which focuses on building a complete EM system [18]. DeepER, a recent work [19] based on deep learning, trains LSTM-based model with word embeddings such as Glove to match tuples. DeepMatcher uses RNN extended by attention mechanism to perform entity matching between text instances [3]. MCA applies multi-attention mechanisms to ensure that contextual information and dependency are better captured [20], which focus on solving the matching problem of text instances. Seq2SeqMatcher [21] and HierMatcher [22] are works specially designed for heterogeneous EM, which focus on the heterogeneity of tuple structure (*i.e.*, tuples from different data sources have different numbers of attributes or attributes have different names). However, we focus on heterogenous attributes with different types (*i.e.*, numeric, textual and so on). There are also several works adopting transfer learning to tackle the issue of data scarcity [23–25].

Benefitting from its capability of learning general language representation and avoiding training a new model from scratch, Pre-Trained Language Models (PLMs) have been widely applied in various NLP tasks. DL-based methods of EM are also applying PLMs to resolve entity [26,27]. [26] is the first work in this direction, which compares four of the most recent transformer architectures (BERT, XLNet, DistilBERT and RoBERTa) on the task of entity matching. Ditto's [27] architecture is similar to [26], but three optimization

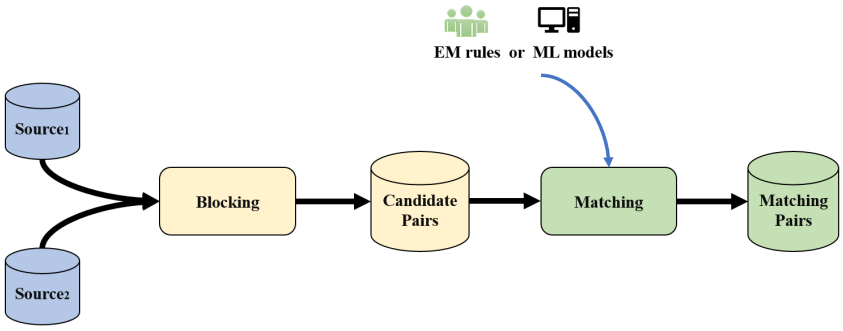

Fig. 2. A typical entity matching pipeline.

techniques are proposed to further improve Ditto's matching capability.

Considering the attribute heterogeneity in data sources, CST manually selected similarity measures for different attributes in their model [7]. To select similarity measures and thresholds for ER rules, Chaudhuri *et al*. proposed a recursive divide and conquer strategy [4], while Wang *et al*. designed three redundancy-based heuristic algorithms [6]. The latest work MPM proposed an end-to-end neural networks framework to select appropriate similarity measures for different attributes [8]. Compared with MPM which selects one specific similarity measure for each attribute from only one perspective, our method propose a hybrid measurement framework to compute the similarity of attribute values from multiple perspectives. Also different from the RNN variant used in MPM, we use DeBERTa with stronger general language learning ability to compute the semantic similarity of attribute values. In addition, we further take the attribute importance into account for entity matching decision and introduce two optimizations to improve EM quality.

3. PROBLEM DEFINITION AND METHOD OVERVIEW

3.1 Problem Definition

Entity matching aims to identify tuples corresponding to the same real-world entity. Formally, given entity tuples $e = \{< A_1, a_1 >, < A_2, a_2 >, ..., < A_m, a_m >\}$ and $e^{'} = \{< A_1, a_1^{'} >, < A_2, a_2^{'} >, ..., < A_m, a_m^{'} >\}$ from two different data sources, with aligned attribute A_1, A_2,..., A_m, and the values on the corresponding attributes are a_1, a_2,..., a_m(the same as $e^{'}$), the purpose of entity matching is to predict the probability $P(y = 1|e, e^{'})$ that e and $e^{'}$ correspond to the same real-world entity based on the similarity of aligned attribute values. A general entity matching pipeline is shown in Fig. 2.

The pipeline mainly includes two steps: Blocking and Matching. The purpose of the Blocking step is to avoid comparing all possible tuple pairs in the two tables. With only a few candidate pairs going on to the next Matching step, search space for entity resolution is greatly reduced. The Matching step determines whether the candidate pair corresponds to the same real-world entity by measuring the similarity of them. Our work in this paper focuses on the Matching step.

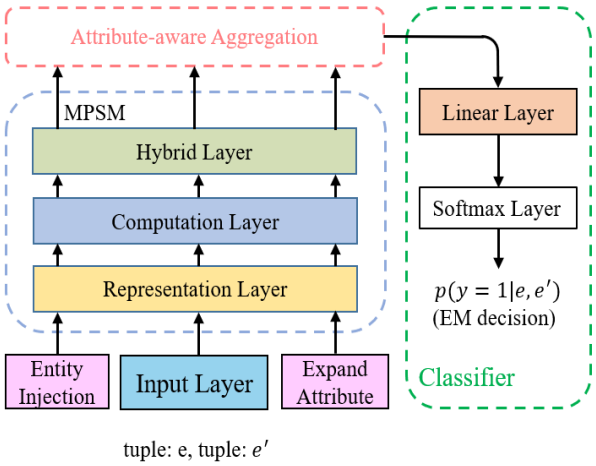

Fig. 3. The framework of our AAMPSM model.

3.2 Method Overview

Our entity matching framework in Fig. 3 is composed of three modules: multi-perspective similarity measurement (MPSM), attribute-aware aggregation and classifier. Firstly, we achieve comprehensive multi-perspective similarity computation through MPSM module. Then, we aggregate the matching evidences (similarity results) from all aligned attributes according to the importance of attributes through the attribute-aware aggregation module. Finally, the matching evidences are sent to the classifier for making EM decision. In addition, we alleviate the impact of dirty data such as swap errors on the model through cross attribute comparison, while external entity knowledge is injected to further improve our model's matching capability.

4. METHOD

In Section 3 we introduce our method, then in this section we will describe each module of our method in detail.

4.1 Multi-Perspective Similarity Measurement

With the rapid growth of data scale, an entity usually contains various types of attributes, such as textual, numeric and so on. Some current work focuses on how to select appropriate similarity measures for different types of attributes. However, these methods lose much matching information from other perspectives by only selecting one specific similarity measure for each attribute. For a pair of attribute values, we hope to measure their similarity from the perspectives of semantics, string, numeric and so on. We believe that the similarity result based on the comprehensive multi-perspective measurement is more accurate. We introduce a hybrid similarity measurement for each attribute by taking multiple perspectives of attributes into account, which will achieve comprehensive multi-perspective similarity computation. Compared to other methods, our method will capture

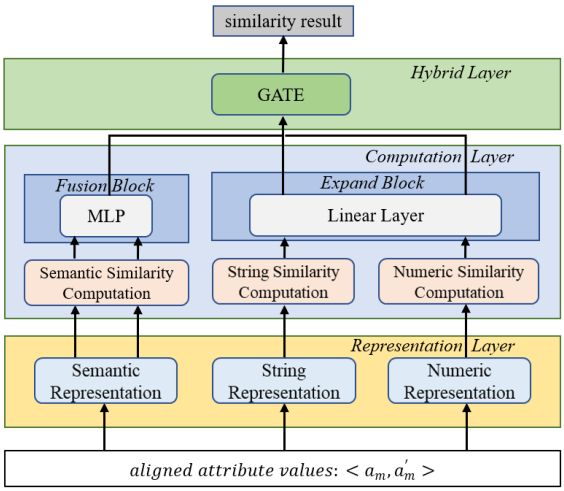

Fig. 4. The framework of MPSM module.

more matching evidences from attributes. In addition, considering that different types of attributes tend to measure their similarity from a specific point of perspective, we designs a gate function to realize multi-perspective similarity measurement, which can take this fact into account at the same time.

Given two entity tuples e and e', similar to other structured matching methods, we rewrite the two tuples into the following form in the Input Layer,

$$A_1 : < a_1, a_1^{'} >, A_2 : < a_2, a_2^{'} >, ..., A_m : < a_m, a_m^{'} >$$

where $A_1, A_2, ..., A_m$ is aligned attributes, $a_1, a_2, ..., a_m$ are their values. Then the attribute value pairs will be feed into MPSM for similarity computation. The framework of MPSM module is shown in Fig. 4, which contains three layers: Representation Layer, Computation Layer, Hybrid Layer.

Representation Layer. In order to compute attribute values' similarity from multi-perspective, we use three kinds of representation for each attribute in this layer. Specifically, for each attribute value: (1) its numeric value (numeric representation); (2) its character sequence (string representation); (3) its sequences satisfying the following form, that is, the input form required by DeBERTa model (semantic representation):

$$seq_{left} = [CLS] + S_m + [SEP] + S_m^{'} + [SEP] \tag{1}$$

$$seq_{right} = [CLS] + S_m^{'} + [SEP] + S_m + [SEP] \tag{2}$$

where S_m and $S_m^{'}$ represent the values of the two tuples on the attribute A_m, respectively. We believe this treatment will teach DeBERTa model to make symmetric decisions, which will be used for computing semantic similarity of attribute values in computation layer. In particular, for non-numeric attributes, we use the length of the attribute value as its numeric representation, which will keep as much information about attribute values as possible.

Computation Layer. This layer compares aligned attribute values from multi-perspective using a set of learnable similarity measures. Specifically, we use Rel_sim ($2|a-b|/(|a|+|b|)$), where a and b are numbers) for numeric similarity computation, Jaro_sim [28], Lev_sim [29] and Jac_sim [30] for string similarity computation, DeBERTa [31] for semantic similarity computation. It should be noted that we use the PLM DeBERTa, which recently sits atop the SuperGLUE leaderboard, to compute the semantic similarity of attribute values. Different from the vanilla RNNs model or other common PLMs (such as: BERT, RoBERTa) used in other EM methods, DeBERTa further improves the ability of model learning general language representation by disentangled attention mechanism and enhanced mask decoder, which is very important for semantic similarity computation of attribute values in EM tasks. For all aligned attribute values, we use the above measures to compute their similarity from multiple perspectives about numeric, string, semantics. It should be noted that our framework is flexible and can be easily extended with advanced similarity measures.

For semantic similarity computation, we use DeBERTa as follows: firstly, we feed seq_{left} and seq_{right} to DeBERTa respectively and the correspondingly output of DeBERTa marked as res_{left} and res_{right},

$$res_{left} = DeBERTa(seq_{left}) \tag{3}$$

$$res_{right} = DeBERTa(seq_{right}) \tag{4}$$

where seq_{left} and seq_{right} are the sequences from semantic representation of Representation Layer. Then, we get the similarity result as follows:

$$\mathbf{s}_{DeBERTa} = MLP([res_{left}, res_{right}]) \tag{5}$$

where MLP is a multi-layer perceptron with two hidden layers, and $[\cdot,\cdot]$ is a concatenation operation.

For subsequent unified computation, we project all similarity values from multi-perspective to d-dimension similarity vectors which have the same dimension with DeBERTa's output. Specifically, we let a similarity value passes through a linear layer and use a nonlinear activation function to activate it (Here is tanh), then the output is the similarity vector corresponding to the similarity value. These similarity vectors will be sent to the hybrid layer together for further computation.

Hybrid Layer. For EM results are sensitive to similarity measures, we apply a gate mechanism for adaptively selecting appropriate similarity measures for each attribute in this layer, and then get a hybrid measurement of the attribute based on the gate weights. The main motivation is that the weight introduced by the gate mechanism can measure the effect of different similarity results on the final EM decision.

Specifically, for each aligned attribute A, let the result outputted by n similarity measures in the comparison layer is $\boldsymbol{s} = [\boldsymbol{s}_1,\ \boldsymbol{s}_2, ...,\ \boldsymbol{s}_n]$, our gate mechanism will learn a soft mask vector $\mathbf{g} = [g_1,\ g_2, ...,\ g_n]$ for similarity measures selection, where $g_1 + g_2 + ... + g_n = 1$. We learn $\boldsymbol{g}$ using:

$$\boldsymbol{g} = softmax(\sigma(W_1\boldsymbol{s}_1 + W_2\boldsymbol{s}_2 + ... + W_n\boldsymbol{s}_n + b)) \tag{6}$$

where W_1, W_2,..., W_n and b are parameters to be learned. Then the similarity result from

hybrid layer can be expressed as:

$$\boldsymbol{C} = \sum_{i=1}^{n} g_i \boldsymbol{s}_i. \tag{7}$$

This soft selection can not only highlight the appropriate similarity measures for each attribute, but also achieve the purpose of computing the similarity of aligned attribute values from multiple perspectives.

4.2 Attribute-Aware Aggregation

Intuitively, the matching evidences from different attributes have different importance on final matching decision. For the example in Fig. 1, compared with attribute **Price**, the matching evidence from the attribute **Manufacturer** is more helpful to resolve entity.

To solve the problem, we introduce an attribute-aware aggregation module to aggregate the matching evidences from all aligned attributes according to the importance of each attribute for the final EM decision. Specifically, let the matching evidence (a vector) from aligned attribute A_m be $\boldsymbol{C}_m$ from MPSM module, we randomly initialize a query vector q for computing attention weights, which will be learned during training, then for all attributes, the attention weights can be calculated as follows:

$$\alpha_i = softmax(q \cdot \boldsymbol{C}_i). \tag{8}$$

And the weights represent the importance of corresponding attribute on deciding the final EM results. Finally, we aggregate the matching evidences from all aligned attributes according to the attention weights for obtaining the final matching evidence:

$$\boldsymbol{r} = \sum_{i=1}^{m} \alpha_i \boldsymbol{C}_i. \tag{9}$$

4.3 Classifier

As usual, We regard EM as a binary classification task. The entity's matching evidence $\boldsymbol{r}$ from aggregation layer is fed to the classifier for entity matching. Specifically, the matching probability $P(y|e,e')$ of tuples e and e' is yielded by:

$$P(y|e,e') = softmax(W\boldsymbol{r} + b). \tag{10}$$

4.4 Model Learning

Given a training set D containing a set of training instances $(e_i, e_i^{'}, y_i)$, where e_i and $e_i^{'}$ are a pair of tuples and y_i is a golden label, we train our model by minimizing the focal loss:

$$loss = \frac{-1}{|D|} \sum_{i=1}^{|D|} [\alpha_t y_i (1-p)^{\gamma} log(p) + (1-\alpha_t)(1-y_i) p^{\gamma} log(1-p)] \tag{11}$$

where $|D|$ is number of training examples, p is the probability of $y_i = 1$ (matching) outputted by our model, α_t and γ are hyperparameters to be set.

Compared with the cross-entropy loss usually used in other EM methods, the focal loss can better deal with the problem of serious sample imbalance, and make our model pay more attention to difficult tuples by introducing parameters α_t and γ. Furthermore, the proposed framework can implement global optimization by modeling all components in a single neural network and making end to end learning for all components.

4.5 Optimization Techniques

External entity knowledge injection. The task of EM is to determine whether the tuple pair corresponds to the same real entity. For the diversity of data sources, it is likely that the values in two tuples are the same, but they actually correspond to completely different entities. For example, although "Paris" (the capital of France) and "Paris Hilton" are both called "Paris", they are totally different. Therefore, we also need to consider the ambiguity of entities. If we can get the real entity information corresponding to each tuple, the EM task will be greatly simplified. We inject real entity information outside the data source into the model to increase the available matching evidences for the model, so as to improve the quality of entity matching. Specifically, we first connect all attribute values of the tuple as the description of the tuple, and then obtain the entity information corresponding to the tuple description from Wikipedia by using *Dexter*, which is a famous open source tool for entity linking. Finally, the model feeds the matching evidences obtained by the real entity information into the aggregation module to participate in the final decision. The experimental results show that our model can effectively improve the quality of entity matching by injecting the matching information outside the data source.

Cross-attribute comparison. With the growth of data scale, swap errors (attribute values are swapped or some attribute values appear in another attribute) often occur due to human operation errors or machine failures, which is common in data sources. When swap errors occur, traditional structured matching methods can not capture the matching evidences on the attributes, and may even bring wrong information to the final matching decision. In addition, some attributes of tuple may be related, so matching information will also be included in non-aligned attributes. In order to alleviate the influence of such dirty data on matching result and capture the evidences from non-aligned attributes, we propose to use cross-attribute comparison. We first construct an extended attribute through connecting all attribute values, then perform comprehensive multi-perspective similarity measurement on the extended attribute, and take the similarity results as part of the matching evidence into the subsequent aggregation steps. By this way, the matching evidences located on different attributes can be easily captured, and the impact of swap errors on final decision can be alleviated indirectly. At the same time, this method also enables our model to deal with the heterogeneity problem of tuples (*i.e.*, two tuples have different attribute names or different numbers of attributes) that traditional structured matching methods cannot solve.

5. EXPERIMENT EVALUATION

5.1 Datasets

We conduct experiments on open datasets published by the research group of DeepMatcher [3], in which *Walmart-Amazon* and *Amazon-Google* contain product data from

different data sources with attribute heterogeneity and dirty data in the real world. The statistics of these datasets are depicted in Table 1. The columns include the abbreviation of datasets, application domain (*Domain*), candidate pairs after blocking (*#Pair*), matching pairs (*#Match*), number of attributes (*#Attr*). As other EM solutions, each dataset is split into the training, validation, and test sets using the ratio of 3:1:1. Following previous studies, we evaluate all solutions using F_1-*score*, defined as $2PR/(P+R)$, where $P(precision)$ is the fraction of matching predictions that are correct, and $R(recall)$ is the fraction of correct matches being predicted as matches.

Table 1. Datasets for our experiments.

Dataset	Domain	#Pair	#Match	#Attr
Amazon-Google	software	11460	1167	3
Walmart-Amazon	electronics	10242	962	5
DBLP-ACM	citation	12363	2220	4
DBLP-Scholar	citation	28707	5347	4
iTunes-Amazon	music	539	132	8
Abt-Buy	product	9575	1028	3

5.2 Baselines

We compare our method with following baselines:

Magellan [32]: A state-of-the-art non-DL based ER baseline. Magellan uses a variety of similarity measures to generate a large number of features, and trains various classifiers on this basis, such as decision tree, random forest and SVM, *etc*.

DeepMatcher [3]: A representative work of DL based EM, which uses a recurrent neural network with additional attention mechanism for distributed representation (a vector) of attributes, and a binary classifier for entity matching.

MPM [8]: An end-to-end framework which can select appropriate measure for different attributes to settle ER problems.

MCA [20]: Multi-attention mechanism is applied to ensure better capture of context information and relevance.

Ditto [27]: A novel EM solution based on pre-trained language models such as BERT. Three optimization techniques are proposed to further improve Ditto's matching capability through injecting domain knowledge, summarizing long strings, and augmenting training data with (difficult) examples.

For fair comparison, the results reported for above baselines are all from the corresponding papers.

5.3 Model Training

Parameters. Our method was implemented with PyTorch. We use Adam optimizer, with learning rate $\eta = 0.00001$, $\beta_1 = 0.9$ and $\beta_2 = 0.99$. During training, we use early stopping, patience is 4, and mini-batch size is 2.

Hardware. All experiments were conducted on a single Nvidia RTX 3090 GPU (24GB Memory) with Inter(R) Xeon(R) Gold 6226R CPU@2.90GHz and 128GB memory.

5.4 Overall Results

Table 2 shows the performance of our model and all baselines. We can get following observations from Table 2:

Table 2. F_1-*score* comparison for our method and baselines on the datasets.

Datasets	Magellan	DeepMatcher	MPM	MCA	Ditto	AAMPSM
Amazon-Google	49.1	69.3	70.7	71.4	75.6	**79.2**
Walmart-Amazon	71.9	67.6	73.6	74.7	86.7	**87.3**
DBLP-ACM	98.4	98.4	–	98.9	99.0	**99.0**
DBLP-Scholar	92.3	94.7	–	95.2	95.6	**95.6**
iTunes-Amazon	91.2	88.5	–	–	97.1	**98.2**
Abt-Buy	43.6	62.8	–	70.8	89.3	**93.0**

(1) By performing comprehensive multi-perspective similarity comparison and attribute-aware aggregation, our method achieves the best performance on all datasets, significantly outperforming the previous methods on most of the datasets. Compared with the most advanced non-DL system Magellan, our method improves F_1 on all datasets by up to 49.4%. Compared with the recently proposed DL based method DeepMatcher, our method improves F_1 on all datasets by up to 30.2%. Compared with the end-to-end neural network method MPM, which focuses on selecting the optimal similarity measures for different attributes, our method achieves 8.5% and 13.7% F_1 improvements on *Amazon-Google* and *Walmart-Amazon*, respectively; (2) Due to the heterogeneity of attributes and the diversity of similarity measures, it is particularly important to select appropriate similarity measures for different attributes. With multi-perspective similarity comparison, our method outperforms DeepMatcher and Ditto, which use the same similarity measure for each attribute. Compared with MCA, which focuses on solving the matching problem of text instances, our method achieves greater F_1 improvement especially on *Amazon-Google* and *Walmart-Amazon* with the heterogeneity of attributes. Furthermore, if only one specific similarity measure is selected for each attribute as MPM, a lot of matching information from other perspectives will be lost, so the final EM performance will decline. The experimental results also prove that the performance of our method significantly outperforms MPM on *Amazon-Google* and *Walmart-Amazon* with the heterogeneity of attributes.

Table 3. Effectiveness of MPSM on the datasets.

Datasets	AAMPSM-one	AAMPSM
Amazon-Google	75.0	**79.2**
Walmart-Amazon	25.7	**87.3**
DBLP-ACM	82.3	**99.0**
DBLP-Scholar	72.5	**95.6**
iTunes-Amazon	55.3	**98.2**
Abt-Buy	19.4	**93.0**

Table 4. Effectiveness of attention-aware aggregation on the datasets.

Datasets	AAMPSM-mean	AAMPSM
Amazon-Google	77.0	**79.2**
Walmart-Amazon	87.1	**87.3**
DBLP-ACM	98.3	**99.0**
DBLP-Scholar	93.1	**95.6**
iTunes-Amazon	96.3	**98.2**
Abt-Buy	90.1	**93.0**

Table 5. Attention weights of attributes in *Amazon-Google* and *Walmart-Amazon* datasets.

Amazon-Google		Walmart-Amazon	
attribute	weight	attribute	weight
title	0.1950	title	0.1478
manufacturer	0.2132	category	0.1423
price	0.2009	brand	0.1423
value	0.1947	modelno	0.1424
entity	0.1963	price	0.1421
		value	0.1408
		entity	0.1423

5.5 Ablation Study

Effectiveness of MPSM. Firstly, we analyze the effectiveness of the multi-perspective similarity measurement (MPSM) module by comparing AAMPSM and the model that selects only one specific similarity measure for each attribute. The results are shown in Table 3. AAMPSM-one in Table 3 is the variant of AAMPSM, which only selects the most appropriate similarity measure for each attribute in selection layer (that is, selecting the similarity measure corresponding to the maximum weight in the gate mechanism). By introducing a hybrid similarity measure on each attribute to achieve comprehensive multi-perspective similarity comparison, our method achieves different degree of F_1 improvement by up to 73.6% on dataset *Abt-Buy*. We believe that multi-perspective matching information is important to determine whether a pair of tuples match or not. Therefore, we use the gate mechanism to select similarity measures and construct a hybrid similarity measure to compare the similarity of attribute values from the perspectives of semantics, string and numeric. The experimental results in Table 3 also verify the effectiveness of MPSM.

Effectiveness of attention-aware aggregation. Then, we analyze the effectiveness of the attribute-aware aggregation module by comparing our model AAMPSM and the model AAMPSM-mean that aggregates the matching evidences from all aligned attributes by averaging them. The results are shown in Table 4. By introducing attribute-aware aggregation module, our method AAMPSM achieves different degree of F_1 improvements on all datasets. As our intuition, each attribute has different impact on the final matching decision, and even some attributes will mislead the matching decision. Therefore, we introduce an attention mechanism to fully take the importance of attributes into account when aggregating matching evidences. The results in Table 4 also demonstrate the ef-

Table 6. Effectiveness of optimization techniques on the datasets.

Datasets	AAMPSM-base	AAMSPM(-eki)	AAMPSM(-cac)	AAMPSM
Amazon-Google	77.2	77.8	77.6	**79.2**
Walmart-Amazon	81.8	87.0	82.0	**87.3**
DBLP-ACM	96.4	98.7	97.6	**99.0**
DBLP-Scholar	89.1	93.1	90.3	**95.6**
iTunes-Amazon	87.3	91.5	98.1	**98.2**
Abt-Buy	92.3	92.4	92.9	**93.0**

fectiveness of our attention-aware aggregation. The attention weight of each attribute in two datasets *Amazon-Google* and *Walmart-Amazon* are shown in Table 5, which further clarify the fact that different attributes have different importance.

Effectiveness of optimizations. Finally, we analyze the effectiveness of two optimization techniques proposed in our paper by comparing AAMPSM and its variants AAMPSM-base without any optimization, AAMPSM(-eki) without external entity knowledge injection, and AAMPSM(-cac) without cross-attribute comparison. The results are shown in Table 6. Compared with AAMPSM-base without any optimization, AAMPSM improves the performance of EM on all datasets, especially with 6.5% F_1 improvement on *DBLP-Scholar*, which verifies the effectiveness of the two optimization techniques introduced in this paper. Furthermore, we conduct experiments to validate the improvements by introducing only one of the optimization technique, AAMPSM(-eki) for introducing only cross-attribute comparison, and AAMPSM(-cac) for only introducing external entity knowledge injection. It can be found from Table 6 that either **External Entity Knowledge Injection** or **Cross-Attribute Comparison** can improve the performance of entity matching effectively. In general, while introducing all optimization techniques, the performance is the best. At the same time, any optimization technique is effective, which is validated by experimental results.

6. CONCLUSION

This paper proposes an EM method based on attribute-aware and multi-perspective similarity measurement, which can compare the similarity of attribute values from the perspectives of semantics, string and numeric, and fully consider the importance of attributes for determining the final matching decision. Experimental results show that our method significantly outperforms state-of-the-art results. In future work, we want to take more perspectives into account for comparing the similarity of attribute values in EM. In addition, we will focus on how to alleviate the impact of other common dirty data (such as spelling errors) in real world.

REFERENCES

1. A. Doan, A. Halevy, and Z. Ives, *Principles of Data Integration*, Elsevier, 2012.
2. X. L. Dong and D. Srivastava, "Big data integration," *Synthesis Lectures on Data Management*, Vol. 7, 2015, pp. 1-198.

3. S. Mudgal, H. Li, T. Rekatsinas *et al.*, "Deep learning for entity matching: A design space exploration," in *Proceedings of International Conference on Management of Data*, 2018, pp. 19-34.
4. S. Chaudhuri, B.-C. Chen, V. Ganti, and R. Kaushik, "Example-driven design of efficient record matching queries," in *Proceedings of the 33rd International Conference on Very Large Data Bases*, Vol. 7, 2007, pp. 327-338.
5. N. Koudas, A. Marathe, and D. Srivastava, "Flexible string matching against large databases in practice," in *Proceedings of the 30th International Conference on Very Large Data Bases*, Vol. 30, 2004, pp. 1078-1086.
6. J. Wang, G. Li, J. X. Yu, and J. Feng, "Entity matching: How similar is similar," in *Proceedings of the VLDB Endowment*, Vol. 4, 2011, pp. 622-633.
7. K. Chao, G. Ming, X. Chen, W. Qian, and A. Zhou, "Entity matching across multiple heterogeneous data sources," in *Proceedings of International Conference on Database Systems for Advanced Applications*, 2016, pp. 133-146.
8. C. Fu, X. Han, L. Sun, B. Chen, and H. Kong, "End-to-end multi-perspective matching for entity resolution," in *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, 2019, pp. 4961-4967.
9. H. B. Newcombe, J. M. Kennedy, S. J. Axford, and A. P. James, "Automatic linkage of vital records," *Science*, Vol. 130, 1959, pp. 954-959.
10. A. H. Doan and A. Y. Halevy, "Semantic integration research in the database community: A brief survey," *AI Magazine*, Vol. 26, 2005, pp. 83-94.
11. N. Koudas, S. Sarawagi, and D. Srivastava, "Record linkage: similarity measures and algorithms," in *Proceedings of ACM SIGMOD International Conference on Management of Data*, 2006, pp. 802-803.
12. M. A. Hernández and S. J. Stolfo, "The merge/purge problem for large databases," *ACM Sigmod Record*, Vol. 24, 1995, pp. 127-138.
13. R. Singh, V. V. Meduri, A. Elmagarmid, S. Madden, P. Papotti, J.-A. Quiané-Ruiz, A. Solar-Lezama, and N. Tang, "Synthesizing entity matching rules by examples," *Proceedings of the VLDB Endowment*, Vol. 11, 2017, pp. 189-202.
14. L. Li, J. Li, and H. Gao, "Rule-based method for entity resolution," *IEEE Transactions on Knowledge and Data Engineering*, Vol. 27, 2014, pp. 250-263.
15. J. Wang, T. Kraska, M. J. Franklin, and J. Feng, "Crowder: Crowdsourcing entity resolution," *arXiv Preprint*, 2012, arXiv:1208.1927.
16. C. Gokhale, S. Das, A. Doan, J. F. Naughton, N. Rampalli, J. Shavlik, and X. Zhu, "Corleone: Hands-off crowdsourcing for entity matching," in *Proceedings of ACM SIGMOD International Conference on Management of Data*, 2014, pp. 601-612.
17. D. Firmani, B. Saha, and D. Srivastava, "Online entity resolution using an oracle," *Proceedings of the VLDB Endowment*, Vol. 9, 2016, pp. 384-395.
18. P. V. Konda, *Magellan: Toward Building Entity Matching Management Systems*, The University of Wisconsin-Madison, 2018.
19. M. Ebraheem, S. Thirumuruganathan, S. Joty, M. Ouzzani, and N. Tang, "Distributed representations of tuples for entity resolution," in *Proceedings of the VLDB Endowment*, Vol. 11, 2018, pp. 1454-1467.
20. D. Zhang, Y. Nie, S. Wu, Y. Shen, and K.-L. Tan, "Multi-context attention for entity matching," in *Proceedings of the Web Conference*, 2020, pp. 2634-2640.

21. H. Nie, X. Han, B. He, L. Sun, B. Chen, W. Zhang, S. Wu, and H. Kong, "Deep sequence-to-sequence entity matching for heterogeneous entity resolution," in *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 2019, pp. 629-638.
22. C. Fu, X. Han, J. He, and L. Sun, "Hierarchical matching network for heterogeneous entity resolution," in *Proceedings of the 29th International Joint Conference on Artificial Intelligence and 17th Pacific Rim International Conference on Artificial Intelligence*, 2020, pp. 3665-3671.
23. S. Thirumuruganathan, S. A. P. Parambath, M. Ouzzani, N. Tang, and S. Joty, "Reuse and adaptation for entity resolution through transfer learning," *arXiv Preprint*, 2018, arXiv:1809.11084.
24. J. Kasai, K. Qian, S. Gurajada, Y. Li, and L. Popa, "Low-resource deep entity resolution with transfer and active learning," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 5851-5861.
25. C. Zhao and Y. He, "Auto-EM: End-to-end fuzzy entity-matching using pre-trained deep models and transfer learning," in *Proceedings of the World Wide Web Conference*, 2019, pp. 2413-2424.
26. U. Brunner and K. Stockinger, "Entity matching with transformer architectures-a step forward in data integration," in *Proceedings of International Conference on Extending Database Technology*, 2020, pp. 463-473.
27. Y. Li, J. Li, Y. Suhara, A. Doan, and W.-C. Tan, "Deep entity matching with pre-trained language models," *arXiv Preprint*, 2020, arXiv:2004.00584.
28. W. E. Winkler and Y. Thibaudeau, "An application of the fellegi-sunter model of record linkage to the 1990 U.S. decennial census," Technical Report No. RR91-09, US Bureau of the Census, 1997.
29. V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions and reversals," *Doklady Akademii Nauk SSSR*, Vol. 10, 1966, pp. 707-710.
30. P. Jaccard, "The distribution of the flora in the alpine zone," *New Phytologist*, Vol. 11, 2010, pp. 37-50.
31. P. He, X. Liu, J. Gao, and W. Chen, "Deberta: Decoding-enhanced bert with disentangled attention," *arXiv Preprint*, 2020, arXiv:2006.03654.
32. A. H. Doan, P. Konda, S. Paul, Y. Govind, and M. Christie, "Magellan: toward building ecosystems of entity matching solutions," *Communications of the ACM*, Vol. 63, 2020, pp. 83-91.

Xin Xing received his Bachelor degree in Optoelectronic Information Science and Engineering from Nanjing University of Posts and Telecommunications, China. He is an MS student in School of Computer and Information Technology, Beijing Jiaotong University, China. His research interests include data quality, entity matching and machine learning.

Ning Wang received her Ph.D. degree in Computer Science in 1998 from Southeast University in Nanjing, China. She is currently serving as a Professor in School of Computer and Information Technology, Beijing Jiaotong University, China. Her research interests include web data integration, big data management, data quality and crowdsourcing.