POI-TRANSFORMERS: POI ENTITY MATCHING THROUGH POI EMBEDDINGS BY INCORPORATING SEMANTIC AND GEOGRAPHIC INFORMATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Point of Interest (POI) data is crucial to location-based applications and various user-oriented services. However, three problems are existing in POI entity matching. First, traditional approaches to general entity matching are designed without geographic location information, which ignores the geographic features when performing POI entity matching. Second, existing POI matching methods for feature design are heavily dependent on the experts' knowledge. Third, current deep learning-based entity matching approaches require a high computational complexity since all the potential POI entity pairs need input to the network. A general and robust POI embedding framework, the POI-Transformers, is initially proposed in this study to address these problems of POI entity matching. The POI-Transformers can generate semantically meaningful POI embeddings through aggregating the text attributes and geographic location, and minimize the inconsistency of a POI entity by measuring the distance between the newly generated POI embeddings. Moreover, the POI entities are matched by the similarity of POI embeddings instead of directly comparing the POI entities, which can greatly reduce the complexity of computation. The implementation of the POI-Transformers achieves a high F1 score of 95.8% on natural scenes data sets (from the Gaode Map and the Tencent Map) in POI entity matching and is comparable to the state-of-the-art (SOTA) entity matching methods of DeepER, DeepMatcher, and Ditto (in entity matching benchmark data set). Compared with the existing deep learning methods, our method reduces the effort for identifying one million pairs from about 20 hours to 228 seconds. These demonstrate that the proposed POI-Transformers framework significantly outstrips traditional methods both in accuracy and efficiency.

1 INTRODUCTION

A Point of Interest (POI) is a dedicated geographic entity that people may be interested in, such as a university, an institute, or a corporate office, and is fundamental to the majority of location-based services (LBS) applications. Generally, a POI entity contains multiple attributes, such as name, category, geographic location. A collection of comprehensive, reliable, and up-to-date POI data is important to LBS, service capability and user experience (Rae et al., 2012; Zhao et al., 2019a). Therefore, updating the POI database in timely is substantial significant. In general, POI database updating is comparing the newly generated POI entities with the existing POI entities and adding the new ones into the database. In this process, POI entity matching is crucial since it needs to discriminate the new POI entities from the old ones based on their attributes.

Traditional POI entity matching algorithms usually involve numerous artificial matching rules associated with attributes (Fu et al., 2011; Safra et al., 2010). The most common idea of POI entity matching is calculating the similarity of attributes between two POI entities and obtaining the final score of all the similarities of attributes with weights. Nevertheless, most of the existing algorithms involve simple string similarity algorithms, such as Levenshtein distance, to calculate the similarity of attributes. This largely neglects the semantic information of the text attributes. Considering this problem, some studies introduced semantic models to the POI-related tasks as the semantic model can achieve state-of-art performance in natural language processing tasks (NLP) (Zhao et al.,

2019a;b). However, most of these POI-related models are heavily dependent on the experts' knowledge.

Entity matching has been researched for decades (Barlaug & Gulla, 2021). Current entity matching methods such as Ditto (Li et al., 2020), DeepMatcher (Mudgal et al., 2018), and DeepER (Ebraheem et al., 2018), can compare the similarity between attributes and extract the features of entities through deep learning, and then compare the similarities between potential pairs of entities. Previous studies state that the geographic similarity calculated from the geographic location is a substantially important element in POI entity matching (Almeida et al., 2018; Novack et al., 2018). However, current entity matching methods are mostly designed for general entities without geographic location information. Most of the entity matching methods, in general, learn the features from the attributes equally but the geographic location features are always ignored.

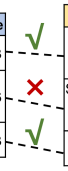
The pre-trained transformer network, such as BERT (Devlin et al., 2018), RoBERTa (Liu et al., 2019), and XLNet (Yang et al., 2019), can achieve state-of-the-art performances in various NLP tasks. A sentence embedding model was proposed by Reimers & Gurevych (2019) for solving the huge computational overhead in semantic similarity search. Meanwhile, Ebraheem et al. (2018) proposed a simple deep learning method, namely DeepER, to directly compare the similarity between entities by learning and tuning the distributed representations of entities. These studies demonstrated that a simple deep learning method can be utilized to translate the POI entities into POI embeddings through fully involving both the semantic of text attributes and geographic location information. Based on the similarity between embeddings, it can be efficiently carried out in many potential matching pairs in POI entity matching.

In this study, we propose a POI-Transformers framework to generate POI embeddings by completely involving the text attributes and geographical locations of POI entities. Experiments show that after training by the Siamese network architecture, the simple model POI-Transformers can well integrate semantic and geographical features, and the newly generated embeddings can fully represent POI entities. The proposed model achieves good performance in entity matching benchmark and SOTA performance in POI entity matching task, and reduces the effort for comparing many POI pairs.

The main contributions of this study can be summarized as follows:

- We propose a simple model, POI-Transformers, for generating POI entity embeddings, which can fully learn the representation embeddings of POI entities by the transformer-based model and geographic location encoding module. Since POI-Transformers use the transformer-based network to process text attributes, this proposed model can seamlessly switch to different transformer-based networks and support different languages.
- The POI embeddings generated from POI-Transformers can be used for POI entity matching task in real-world data. These fully learned embeddings can largely reduce the effort for finding the most similar pair from all POI entities.
- We compare the proposed POI-transformers with the traditional POI entity matching methods and entity matching methods. The results show that our model achieves comparable performance to the DeepER, DeepMatcher, and Ditto in the entity matching tasks. In the POI entity matching task, the proposed POI-Transformers achieves better performance than traditional POI matching methods (e.g. rule-based, weighted). These results demonstrate that this proposed framework can fully learn the text attributes and geographic location information in POI entity matching. Meanwhile, it further implies that adding a domain knowledge module to the original entity matching model might achieve a better performance in the field of entity matching.

Name	Category	Address	Longitude	Latitude
Coastal City	Shopping	NO. 33, Wenxin 5th Road	113.9352	22.5173
Shenzhen Diamond Apartment (No. 16 Branch)	Accommodation services	Guanhaitai,Nashan District, Shenzhen	113.9377	22.5146
Kangda Glasses (Wuhe Branch)	Optical Shop	664 Jihua Road, Bantian Street	114.0618	22.6343



Name	Category	Address	Longitude	Latitude
Coastal City Shopping Center	Shopping Malls	No. 33, Wenxin fifth road, Nanshan District, Shenzhen	113.9356	22.5170
Shenzhen Diamond Apartment (No. 13 Branch)	Hotel	Cuiyong Huating	114.1305	22.5623
Kangda Glasses	Optical Shop	31-3, middle Wuhe Avenue	114.0618	22.6342

Figure 1: Determining the matching POI entities from two data sets in the POI entity matching.

2 RELATED WORK

2.1 ENTITY MATCHING

Here, we summarize the entity matching methods used for the entity matching task, which aims to solve the problem of identifying entities from the real world (Barlaug & Gulla, 2021).

The attributed-aligned comparison strategy is commonly employed in entity methods. This strategy compares attributes in a one-to-one, and further combines the similarity representation on the record level (Barlaug & Gulla, 2021). Specifically, the rule-based method associated with the attributed-aligned comparison strategy is the most classic entity matching method since it is easy to understand and develop (Hernández & Stolfo, 1998; Lim et al., 1996; Wang & Madnick, 1989). Nevertheless, owing to much expert experience required for modifying rules in the rule-based methods, methods based on machine learning (especially deep learning) are gradually developed to automatically learn the features of entities. For example, DeepMatcher (Mudgal et al., 2018), Kasai et al. (2019) and Auto-EM (Zhao & He, 2019) utilized the deep learning method to compare attributes one-to-one before comparing the similarity of records.

To capture better language understanding, some studies introduced the cross-record attention for entity matching (Barlaug & Gulla, 2021). Seq2SeqMatcher (Nie et al., 2019), Ditto (Li et al., 2020) and Brunner & Stockinger (2020) used attention mechanisms to capture semantic features of all words across the compared records. They treat the entity matching task to a sequence-to-sequence matching task by processing the entity pairs into sentences and inputting these sentences into transformer networks. At present, by combine cross-record and multiple optimization techniques (domain knowledge, etc.), Ditto has achieved SOTA performance in the entity matching benchmark.

Both attribute-aligned and cross-record attention methods need to input entity pairs into the model simultaneously, which comes out a large amount of computation effort in the entity matching. Therefore, some studies have proposed approaches to alleviating this problem by comparing the representation of entities. For entity representation methods, it is possible to generate a representation of each record and directly obtain similarity between entity pairs (Barlaug & Gulla, 2021). DeepER (Ebraheem et al., 2018) and AutoBlock (Zhang et al., 2020) applied bidirectional LSTM and self-attention to get the record-level embedding representations, which can achieve good performance in entity matching tasks with low time complexity.

2.2 POI ENTITY MATCHING

POI entity matching can be regarded as a special case of entity matching on POI. As far as we know, the majority of the current methods are dependent on the attribute-aligned comparison (including rule-based and machine learning-based). McKenzie et al. (2014) proposed a weighted combination model on multiple attributes (e.g., name, type, and geographic location) of POI, and achieved high accuracy in the Foursquare and Yelp dataset. Li et al. (2016) proposed an entropy-weighted method to POI matching by integrating attributes with allocation weights via information entropy. This entropy-weighted method is applied to Baidu and Sina POI matching and achieved good performance. Meanwhile, some studies applied the weighted summation based on the graph method, and the weights of different attributes can further be obtained by an unsupervised method. For example, Novack et al. (2018) presented a graph-based POI matching method with two matching strategies. In which, POIs are regard as nodes and matching possibilities regarded as edges. Almeida et al. (2018) first proposed a data-driven learning method for automatic POI matching based on an outlier detection algorithm. However, these methods for feature design are heavily dependent on the experts' knowledge.

To improve the accuracy, the methods of text semantic are also applied to the POI matching task. Dalvi et al. (2014) considered both domain knowledge and geographical knowledge and presented an unsupervised POI matching model based on a language model. They assign weights to different words in POI names and their method can achieve an accuracy of about 90% in POI deduplication. Yu et al. (2018) proposed an approach based on semantic technologies to automate the POI matching and conflation, which achieved a conflation accuracy of 98% in shopping center POIs. However, as far as we know, employing POI embedding for POI matching task, which this paper aims to explore, has not been covered by existing studies.

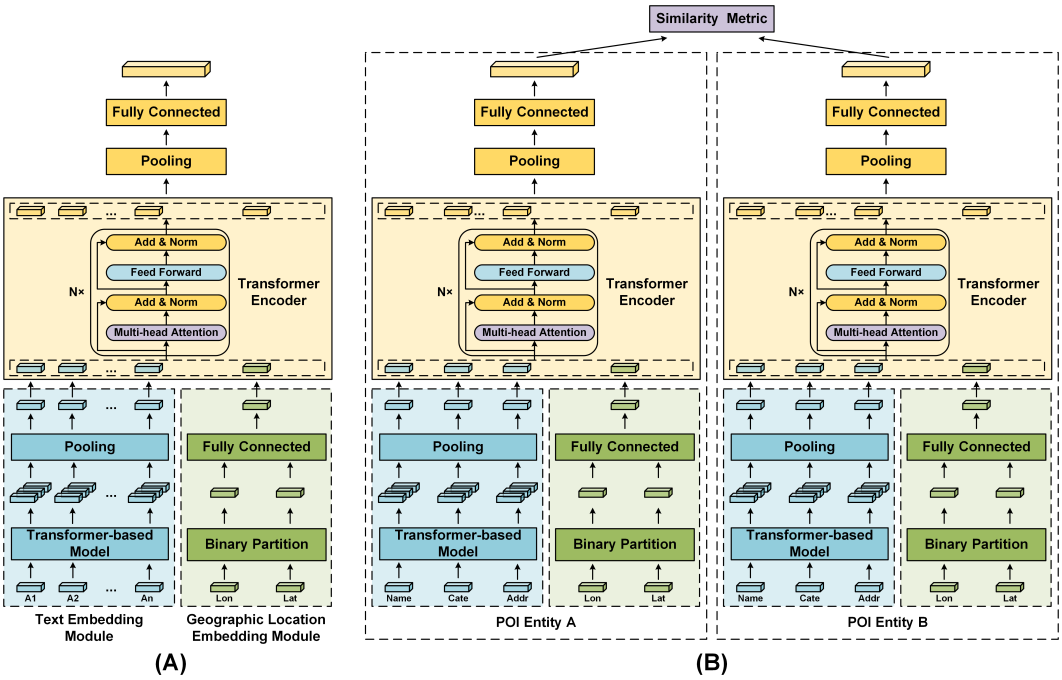


Figure 2: The proposed POI-Transformers framework. It includes two modules in processing the original POI attributes, namely text embedding module and geographic location embedding module. The text embedding module processes the text attributes, and the geographic location embedding module only process the geographic information (e.g. longitude and latitude). (A) The general POI-Transformers, it takes multiple attributes (A1, A2, ..., An) as text embedding module input, and longitude (Lon) and latitude (Lat) as geographic location embedding module input. (B) The specific POI-Transformers used for evaluation, this study considers the attributes of Name, Category (Cate), Address (Addr), and geographic location as input.

3 Model Architecture

The architecture of the POI-Transformers is shown in Figure 2. It is a combination of the transformer-based model and geographic location embedding module, which is an extension of the general entity matching. In this work, we aim to achieve the POI entity matching by incorporating semantic and geographic information. Firstly, the semantic feature vectors are extracted from the text attributes (name, category, address, etc.) of the POI entity by using the Transformer-based model (BERT, etc.) and further trained to be fixed-sized attribute embeddings by pooling strategies. Meanwhile, we design a geographic location embedding module to translate the two-dimensional geographic location (longitude and latitude) to meaningful embeddings. Secondly, a transformer encoder layer is employed to encode the text embeddings and location embeddings by a multi-head attention mechanism. Finally, a pooling layer and a fully connected layer are adopted to obtain POI entity embeddings.

Figure 2(B) describes a specific POI-Transformers for evaluation in this study. In this framework, we consider the text attributes of name, category and address in the Transformer-based model as these attributes are most important to POI entities. Combining with the geographic information (longitude and latitude), the three text attributes, in turn, can be used for identifying a POI entity in the nature world. In the training process, the Siamese networks are adopted to update the weights of semantic and geographic attributes to ensure the newly generated POI embedding meaningful semantically and geographically and valid in similarity metrics (such as cosine, Euclidean).

3.1 TEXT EMBEDDING MODULE

The text embedding module designed in our study attempts to translate the multiple text attributes of POI entities into semantic embeddings through the transformer-based network. Transformer-based pre-trained models, such as BERT and RoBERTa, can achieve the state-of-the-art performance, which in turn makes the transformer-based models widely used. The SOTA entity matching method Ditto has proved that transformer-based networks can fully learn the knowledge from entity attributes by treating entity-pair as sequence-pair (Li et al., 2020).

Here, we consider each POI text attribute as a sentence and generate a corresponding embedding that can represent this text attribute. In this study, a transformer-based network is employed to extract the semantic text embeddings of POI text attributes, such as name, category, address. After the transformer-based network, we further utilize a pooling layer to derive fix-sized semantic vectors of POI text attributes. In the pooling layer, the output of a special CLS token is not used to represent the text since there is no evidence showing the embedding of the CLS token is semantically meaningful (Reimers & Gurevych, 2019). Instead, mean-strategy polling is utilized in the pooling layer of the POI-Transformers framework. This means an average value of embeddings of all tokens is set as the embedding of the POI text attributes. In addition, to simplify the model and maintain the consistency of POI embeddings, only one transformer-based network is utilized for extracting the text attributes.

3.2 GEOGRAPHIC LOCATION EMBEDDING MODULE

The geographic location of POI is two-dimensional spatial information, involving longitude and latitude. To obtain the geographic information on POI, we design a geographic location embedding module in the POI-Transformers framework to translate the two-dimensional location into geographically meaningful embeddings, which can easier to identify the difference between the input geographic locations.

Here, we generate meaningful geographic vectors for the longitude and latitude of POI by utilized a location encoding method, the GeoHash (Liu et al., 2014) algorithm, which can encode the numerical longitude and latitude of a specific region on the Earth into strings. In this study, the primary purpose of the GeoHash in the POI-Transformers framework is to convert the longitude and latitude into binary vectors. To be specific, for a given geographic location (lon1, lat1), the location encoding layer in the GeoHash algorithm recursively can divide the longitude into intervals and mark the longitude code with 0 if the lon1 belongs to the left interval. If the lon1 belongs to the right interval, the longitude code is marked by 1. When the number of divisions reaches the set conditions, a code similar to 1101001 is obtained. The binary code of latitude can be also obtained as the way of longitude code. A longer binary array implies a more precise geographic location. When the times of dichotomies reach 30, the maximum error is approximately at 0.0186 meters. Therefore, the code '0' can represent the longitude range (-180, 0), code '00' can represent the latitude range (-180, -90). Similarly, the code '0' represent the latitude range (-90, 0) while code '00' represent the latitude range (-90, -45).

After we get the binary array of longitude and latitude, we can generate a geographic binary array with longitude bits occupied in even digits and latitude bits occupied in odd digits. For instance, the longitude binary code '0' represent the longitude range -180 to 0, and the latitude binary code '1' represent the latitude range 0 to 90, then the geographic binary array '01' can represent a region that longitude range from -180 to 0, and latitude range from 0 to 90. More details can be found in the appendix A.2. After the location encoding layer, a fully connected layer is added for obtaining location embeddings with the same dimension as the semantic vectors.

3.3 EMBEDDING FUSION MODULE

We then incorporate all the embeddings at the POI entity level in the embedding fusion module to ensure the matched POI entities have a large cosine similarity.

The linkages between different attributes of each POI entity possibly facilitate measuring the cosine similarity between POI entities. Specifically, the linkages between (i) category and name. For one thing, each category, such as hospital, university and shopping mass, may contain various names of POI entities. For another, one name of POI entity is likely to belong to different categories. (ii) geographic location and address. The address of POI entities can be obtained by the longitude and

latitude from the interface of the electronic map. In turn, the longitude and latitude can also be searched by the address of POI entities through the interface of the electronic map. Hence, these linkages between the attributes can help discriminate against various POI entities. Moreover, a rule-based method with a weighted average of the similarity scores of all the attributes is generally used for measuring the similarity of two POI entities. Nevertheless, the weights of all the attributes of POI entities, in the traditional POI entity matching, are manually set with prior experience.

To learn the linkages knowledge between the attributes of POI entities, we introduce a transformer encode layer with multi-head self-attention in the embedding fusion module. The self-attention mechanism can link the different parts of a single sequence to obtain the representation of the sequence. This means that we can get the representation of linkages between different attributes when inputting the attributes of POI entities into the self-attention. Hence, the linkages between the attributes of each POI entity can be fully learned by using the multi-head self-attention mechanism. In addition, the attention mechanism can adjust automatically the weights of all attributes in POI entities. In natural language processing, the core function of the attention mechanism is to weigh the input attributes by learning the importance of different parts of a sentence (Vaswani et al., 2017). Compared with the fixing weights set by manual, the weights based on the importance of attributes are much reasonable. Furthermore, in order to obtain fix-sized POI embeddings, we introduce a pooling layer after the transformer encoder layer.

4 EXPERIMENTS

4.1 DATA SETS

We experimented with all the 12 publicly available entity matching data sets used for evaluating Ditto (Li et al., 2020) and DeepMatcher (Mudgal et al., 2018) and a POI entity matching data set generated by ourselves.

For entity matching data sets, each of them consists of the candidate pairs sampled and labeled from two structured entity record tables. In addition, similar to the Ditto and DeepMatcher, we also use the dirty version of the DBLP-ACM, DBLP-Scholar, iTunes-Amazon, and Walmart-Amazon data sets to evaluate the robustness of the proposed model. These dirty data sets are generated by randomly moving each attribute value to the attribute title with a 50% probability. The Abt-Buy data set is dominated by texts and is characterized by the long text attribute. The overview of all the entity matching data sets can be found in appendix A.1.

In this study, we annotated a POI entity matching data set QM-GD-POI generated by a POI dataset of the Tencent Map (QM POI, `https://map.qq.com/`) and a POI data set of the Gaode Map (GD POI, `https://www.amap.com/`). All POI entities contain five attributes: name, category, address, longitude and latitude. We use the open POI query API of Tencent Map and Gaode Map to obtain 7,103 and 6,868 POI entities respectively. Then, we sampled and labeled 9,606 candidate pairs from these two newly generated POI data sets. We also generated the dirty version of QM-GD-POI. Since the attributes of name, longitude and latitude in the POI data set are generally not missing, we remove the type and addresses with a 50% probability to generate a dirty data set.

The training, validation, and test sets of 12 publicly entity matching data sets are set at the ratio of 3:1:1. In the structured and dirty QM-GD-POI data sets, we used the ratio 6:1:3 to construct training, validation and test sets.

4.2 EXPERIMENT SETUP

We implemented POI-Transformers in PyTorch (Paszke et al., 2019) and the Transformers (Wolf et al., 2020) library. We currently use the BERT-Base Chinese model as the base model to extract text semantic features. Further, the BERT-Base Chinese model can replace with other transformer-based pre-training models. We conducted all experiments on a server with Intel i9-10850K CPU @ 3.6GHZ, 64GB memory, NVIDIA GeForce RTX 3090 GPU.

We compared the proposed POI-Transformers with the existing entity matching methods, such as DeepMatcher, Ditto, Magellan, and DeepER and POI entity matching methods Rule-based,

Weighted, iForest on POI entity matching dataset. We also compared variants of POI-Transformers without the Geographic Location Embedding Module (POI-Transformers*).

DeepMatcher: DeepMatcher (Mudgal et al., 2018) is one of the SOTA deep learning-based entity matching approaches. DeepMatcher customizes the RNN to conduct attribute-aligned similarity representation of attributes, and then aggregates the representations of attributes to obtain entity similarity representation between entities.

Ditto: Ditto (Li et al., 2020) is the SOTA entity matching system based on pre-trained Transformer-based language models. Ditto considers the entity matching task as a sequence classification task by splicing entity pairs into sequences. Meanwhile, Ditto developed three optimization techniques (domain knowledge, TF-IDF summarization, and data augmentation) to improve the performance. We use the full version of Ditto with all 3 optimizations in this study.

Magellan: Magellan (Konda, 2018) is a SOTA traditional non-neural entity matching system. This system calculates the similarity features between attributes (Levenshtein distance, etc.), and then uses these features to build a random forest, logistic regression and other traditional machine learning models for entity matching identifying. After model selection, the random forest in Magellan performs best in our POI entity matching dataset, so we report the F1 score in POI entity matching of Magellan obtained by random forest.

DeepER: DeepER (Ebraheem et al., 2018) uses bidirectional RNN with LSTM hidden units on word embeddings to translate each entity to a representation vector. It achieves good accuracy and high efficiency in entity matching tasks.

POI-Transformers: The full version of our proposed model with Geographic Location Embedding Module. In POI entity matching, we used the cosine similarity and SentEval toolkit (Conneau & Kiela, 2018) to evaluate the POI embeddings obtain by the POI-Transformers. When evaluated by the cosine similarity, we set a matching threshold. The entity pairs with embedding cosine similarity higher than the threshold are considered the positive matching pair. SentEval is an evaluation toolkit for evaluating the quality of POI embeddings. We utilized the logistic regression classifier in the SentEval to evaluate the POI embeddings for POI entity matching and entity embeddings for entity matching. To train the POI-Transformers framework, we utilize the softmax objective function to update the weights of POI embeddings as that in Sentence-BERT (Reimers & Gurevych, 2019).

POI-Transformers*: In this version, the Geographic Location Embedding Module is deleted, and the longitude and latitude are directly input into the Text Embedding Module to obtained the representation embeddings.

Rule-based: We designed a rule-based method for POI entity matching. In this method, we first calculated the similarity of the name, category, address and distance between the POI entity pairs. Then, we performed a weighted summation of the similarity between each attribute to obtain the similarity between POI entity pairs. The weights of name, category, address and distance similarity were set to 0.65, 0.1, 0.1, 0.15, respectively according to the experts' knowledge.

Weighted: Li et al. (2016) proposed a Entropy-Weighted method for POI entity matching. This method first calculates the similarity of attributes between POI entity pairs, and then allocates weights of similarity of each attribute by information entropy.

iForest: Almeida et al. (2018) proposed an outlier detection based approach to POI entity matching. This method first computes the similarity of the name, website, address, category and geographic coordinates between POI entity pairs. Further, it obtains similarity between POI entity pairs by using the iForest method.

BERT: Fine-tuning the pre-trained BERT (Devlin et al., 2018) model by our POI matching dataset to do a classification task. We construct POI sentences by concatenating name, category, address, longitude, and latitude for input and get the similarity of two POI sentences.

4.3 RESULTS

All the models run with 20 epochs in the training process and returned the checkpoint with the highest F1 score on the validation set. Table 3 and Table 4 show the results of the entity matching data sets and POI entity matching data sets respectively. We reported the F1 scores of DeepMatcher,

Ditto, Magellan, and DeepER in entity matching data sets from Li et al. (2020) and Barlaug & Gulla (2021).

Table 1: F1 scores on the entity matching data sets. The result of DeepMatcher, Ditto, Magellan, and DeepER are highest available found in Li et al. (2020) and Barlaug & Gulla (2021).

Dataset	DeepMatcher	Ditto	Magellan	DeepER	POI-Transformers
Structured					
Amazon-Google	69.3	75.6	49.1	56.1	63.4
Beer	72.7	94.4	78.8	50.0	77.6
DBLP-ACM	98.4	99.0	98.4	97.6	98.0
DBLP-Google	94.7	95.6	92.3	90.8	92.0
Fodors-Zagats	100.0	100.0	100.0	97.7	98.4
iTunes-Amazon	88.0	97.1	91.2	72.5	88.3
Walmart-Amazon	66.9	86.8	71.9	50.6	60.2
Dirty					
DBLP-ACM	98.1	99.0	91.9	89.6	91.2
DBLP-Google	93.8	95.8	82.5	86.1	88.3
iTunes-Amazon	74.5	95.7	46.8	67.8	63.2
Walmart-Amazon	46.0	85.7	37.4	36.4	59.5
Textual					
Abt-Buy	62.8	89.3	43.6	43.0	55.4

As shown in Table 1, due to the powerful learning ability of deep learning, the models based on deep learning (Ditto, BS, and POI-Transformers) can achieve better performance in entity matching. Meanwhile, we found that the attributed-aligned comparison methods (DeepMatcher) and cross-record interactive methods (Ditto, BS) based on deep learning are generally achieved better performance than the methods based on entity representation. In addition, The POI-Transformers proposed by this study achieved a better performance than the existing entity representation method (DeepER) and the traditional method (Magellan). In some data sets, the POI-Transformers can achieve better performance than the DeepMatcher. These results suggest that in entity matching, the entity representation methods currently have no advantage in accuracy over the POI-Transformers. However, the reduction in computation effort by entity representation methods cannot be ignored, especially in the POI entity matching task with a large number of real-world data set.

We can also find that Ditto and BS outperform other models in the textual data set Abt-Buy. This is possible because attributed-aligned methods and entity representation methods require to transform attribute text to other forms. When the training set is not enough, the learned features cannot fully represent the features of an attribute. The cross-record interactive model can directly interact with the original attributes across records, so as to obtain more meaningful features. In actual, there is no long text in the POI entity, and all attributes contain short text.

Table 2: F1 scores on the POI entity matching datasets.

Method	DeepMatcher	Ditto	Magellan	BERT	POI-Transformers*
Structured	92.4	95.1	90.0	94.5	91.5
Dirty	91.4	92.4	86.4	91.6	86.9
Method	Rule-based	Weighted	iForest	POI-Transformers (Cosine Similarity)	POI-Transformers (SentEval)
Structured	83.33	87.5	88.5	89.3	95.8
Dirty	81.26	86.5	88.2	83.6	92.1

Table 2 shows the result of the POI entity matching data sets. We can find that the transformer-based models (Ditto, Magellan, and POI-Transformers) can achieve better performance than other models in the POI entity matching task. The POI-Transformers* has no advantage over traditional models since we remove the Geographic Location Embedding Module. The cosine similarity between the embeddings of POI entities is directly used for POI entity matching task, which is better than the traditional Rule-based, Weighted, and iForest, but perform worse than other deep learning models. When we used the SentEval to evaluate the embedding generated by POI-Transformers, we find that its performance is a little better than the SOTA entity matching method Ditto, but it is a little worse in the dirty data version. This indicates that the POI-Transformers proposed in this study has achieved SOTA performance in POI entity matching after adding the Geographic Location Embedding Module. Meanwhile, these results also suggest that POI-Transformers has more advantages in dealing with structured data and needs to be improved in dealing with dirty data. However, as far as we know, there are generally not many dirty data in the real-world POI data set.

Table 3: Computation time (seconds) of different number of matching pairs in POI entity matching. Lower is better. All models run in CPU.

# Pairs	DeepMatcher	Ditto	Magellan	BERT	POI-Transformers*
10,000	79.81	796.34	8.92	609.27	13.80
250,000	1,821.72	19,312.45	212.34	15,195.21	152.65
1,000,000	7,156.93	73,958.36	834.64	60,599.56	503.31
# Pairs	Rule-based	Weighted	iForest	POI-Transformers (Cosine Similarity)	POI-Transformers (SentEval)
10,000	0.08	0.03	0.05	9.91	12.58
250,000	1.19	0.86	1.20	71.67	146.05
1,000,000	4.65	3.45	4.70	228.02	491.16

In order to evaluate the computational efficiency for different models, we selected 100, 500, and 1,000 records from Tencent Map POI and Gaode Map POI respectively to from 10,000, 250,000, and 1,000,000 matching pairs. Table 3 shows the computation time of different numbers of matching pairs in POI entity matching. We can see that the computation effort of traditional POI entity matching methods is very low, but it can be seen from Table 3 that the accuracy is the worst. When the cosine similarity of POI embeddings is directly used for POI entity matching, the computation amount is lower than Magellan when the size of data set increases. The three deep learning models, especially the transformer-based modes (Ditto and BERT), have a large amount of computation (approximately 20 hours and 17 hours, respectively). When using SentEval to evaluate the embeddings generated by POI-Transformers, it takes less than 500 seconds to calculate one million matching pairs. These results demonstrate that our POI-Transformers have advantages both in accuracy and computational efficiency in the task of POI entity matching, and have the significance of practical deployment.

5 Conclusions

In this paper, we propose a novel model, the POI-Transformers, for the POI matching task based on pre-trained Transformer-based language models. This model uses a simple architecture to effectively incorporate the semantic features and geographic features to obtain meaningful POI entity embeddings. The POI entities are matched by the similarity of POI embeddings instead of directly comparing the POI entities, which can greatly reduce the complexity of computation. The experimental results show that our proposed POI-Transformers is comparable to SOTA entity matching models (DeepER, DeepMatcher, and Ditto) in entity matching tasks. Moreover, our model achieves the highest F1 score on natural scenes data sets in POI entity matching, and reduces the computation effort for identifying one million pairs from about 20 hours to 228 seconds. The high accuracy and efficiency of the POI-Transformers can help to deploy and use in real-world data set. In addition, our results also demonstrate that domain knowledge fusion in the deep learning model can achieve better results in specific entity matching tasks.

REFERENCES

Alexandre Almeida, Ana Alves, and Rui Gomes. Automatic poi matching using an outlier detection based approach. In *International Symposium on Intelligent Data Analysis*, pp. 40–51. Springer, 2018.

Nils Barlaug and Jon Atle Gulla. Neural networks for entity matching: A survey. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 15(3):1–37, 2021.

Ursin Brunner and Kurt Stockinger. Entity matching with transformer architectures-a step forward in data integration. In *International Conference on Extending Database Technology, Copenhagen, 30 March-2 April 2020*. OpenProceedings, 2020.

Alexis Conneau and Douwe Kiela. Senteval: An evaluation toolkit for universal sentence representations. *arXiv preprint arXiv:1803.05449*, 2018.

Nilesh Dalvi, Marian Olteanu, Manish Raghavan, and Philip Bohannon. Deduplicating a places database. In *Proceedings of the 23rd international conference on World wide web*, pp. 409–418, 2014.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Muhammad Ebraheem, Saravanan Thirumuruganathan, Shafiq Joty, Mourad Ouzzani, and Nan Tang. Distributed representations of tuples for entity resolution. *Proceedings of the VLDB Endowment*, 11(11):1454–1467, 2018.

Shunkai Fu, Jiyang Zhang, and Rong Mu. Ranking factors in devising practical poi search model. In *Proceedings 2011 IEEE International Conference on Spatial Data Mining and Geographical Knowledge Services*, pp. 267–272. IEEE, 2011.

Mauricio A Hernández and Salvatore J Stolfo. Real-world data is dirty: Data cleansing and the merge/purge problem. *Data mining and knowledge discovery*, 2(1):9–37, 1998.

Jungo Kasai, Kun Qian, Sairam Gurajada, Yunyao Li, and Lucian Popa. Low-resource deep entity resolution with transfer and active learning. *arXiv preprint arXiv:1906.08042*, 2019.

Pradap Venkatramanan Konda. *Magellan: Toward building entity matching management systems*. The University of Wisconsin-Madison, 2018.

Lin Li, Xiaoyu Xing, Hui Xia, and Xiaoying Huang. Entropy-weighted instance matching between different sourcing points of interest. *Entropy*, 18(2):45, 2016.

Yuliang Li, Jinfeng Li, Yoshihiko Suhara, AnHai Doan, and Wang-Chiew Tan. Deep entity matching with pre-trained language models. *arXiv preprint arXiv:2004.00584*, 2020.

Ee-Peng Lim, Jaideep Srivastava, Satya Prabhakar, and James Richardson. Entity identification in database integration. *Information Sciences*, 89(1-2):1–38, 1996.

Jiajun Liu, Haoran Li, Yong Gao, Hao Yu, and Dan Jiang. A geohash-based index for spatial data management in distributed memory. In *2014 22Nd international conference on geoinformatics*, pp. 1–4. IEEE, 2014.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Grant McKenzie, Krzysztof Janowicz, and Benjamin Adams. A weighted multi-attribute method for matching user-generated points of interest. *Cartography and Geographic Information Science*, 41 (2):125–137, 2014.

Sidharth Mudgal, Han Li, Theodoros Rekatsinas, AnHai Doan, Youngchoon Park, Ganesh Krishnan, Rohit Deep, Esteban Arcaute, and Vijay Raghavendra. Deep learning for entity matching: A design space exploration. In *Proceedings of the 2018 International Conference on Management of Data*, pp. 19–34, 2018.

Hao Nie, Xianpei Han, Ben He, Le Sun, Bo Chen, Wei Zhang, Suhui Wu, and Hao Kong. Deep sequence-to-sequence entity matching for heterogeneous entity resolution. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pp. 629–638, 2019.

Tessio Novack, Robin Peters, and Alexander Zipf. Graph-based matching of points-of-interest from collaborative geo-datasets. *ISPRS International Journal of Geo-Information*, 7(3):117, 2018.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32: 8026–8037, 2019.

Adam Rae, Vanessa Murdock, Adrian Popescu, and Hugues Bouchard. Mining the web for points of interest. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, pp. 711–720, 2012.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*, 2019.

Eliyahu Safra, Yaron Kanza, Yehoshua Sagiv, Catriel Beeri, and Yerach Doytsher. Location-based algorithms for finding sets of corresponding objects over several geo-spatial data sets. *International journal of geographical information science*, 24(1):69–106, 2010.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pp. 5998–6008, 2017.

Y Richard Wang and Stuart E Madnick. The inter-database instance identification problem in integrating autonomous systems. In *ICDE*, pp. 46–55, 1989.

Thomas Wolf, Julien Chaumond, Lysandre Debut, Victor Sanh, Clement Delangue, Anthony Moi, Pierric Cistac, Morgan Funtowicz, Joe Davison, Sam Shleifer, et al. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45, 2020.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32, 2019.

Feiyan Yu, David A McMeekin, Lesley Arnold, and Geoff West. Semantic web technologies automate geospatial data conflation: conflating points of interest data for emergency response services. In *LBS 2018: 14th International Conference on Location Based Services*, pp. 111–131. Springer, 2018.

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*, pp. 744–752, 2020.

Chen Zhao and Yeye He. Auto-em: End-to-end fuzzy entity-matching using pre-trained deep models and transfer learning. In *The World Wide Web Conference*, pp. 2413–2424, 2019.

Ji Zhao, Dan Peng, Chuhan Wu, Huan Chen, Meiyu Yu, Wanji Zheng, Li Ma, Hua Chai, Jieping Ye, and Xiaohu Qie. Incorporating semantic similarity with geographic correlation for query-poi relevance learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 1270–1277, 2019a.

Ji Zhao, Meiyu Yu, Huan Chen, Boning Li, Lingyu Zhang, Qi Song, Li Ma, Hua Chai, and Jieping Ye. Poi semantic model with a deep convolutional structure. *arXiv preprint arXiv:1903.07279*, 2019b.

A APPENDIX

A.1 OVERVIEW OF DATA SETS

Table 4 shows the overview of publicly available entity matching data sets (from Barlaug & Gulla (2021) and Li et al. (2020)) and a POI entity matching data set (QM-GD-POI) generate by ourselves.

Table 4: Overview of publicly available entity matching data sets and a POI entity matching data set (QM-GD-POI) generate by ourselves. # Records denotes the number of records in each data set (# records of left data source - # records of right data source). # Pos Matches represents the number of positive matches in the data sets, and # Candidates lists the number of records of each data set.

Dataset	Domain	# Records	# Attributes	# Pos Matches	# Candidates
Abt-Buy	Product	1,081 - 1,092	3	1,028	9,575
Amazon-Google	Software	1,363 - 3,226	3	1,167	11,460
Beer	Beer	3,274 - 4,345	4	68	450
DBLP-ACM	Citation	2,616 - 2,294	4	2,220	12,363
DBLP-Scholar	Citation	2,616 - 64,263	4	5,347	28,707
Fodors-Zagats	Restaurant	533 - 331	6	110	946
iTunes-Amazon	Music	4,875 - 5,619	8	132	539
Walmart-Amazon	Electronics	2,554 - 22,074	5	962	10,242
QM-GD-POI	POI	7,103 - 6,833	5	4,833	9,606

A.2 DETAIL OF GEOGRAPHIC LOCATION EMBEDDING MODULE

As illustrated in Figure 3, the left interval of longitude is set as (-180, 0) by the GeoHash algorithm and the right interval is set as (0, 180). Similarly, the left interval of latitude is divided into (-90, 0) while the corresponding right interval is (0, 90). As a result, "01" represents the area where longitude is from -180 to 0 degrees and latitude is from 0 to 90 degrees. As for the "01" area, the GeoHash algorithm continues to bisect the latitude and longitude of this region and the "0101" denotes the area where the longitude ranges from (-180, -90), and the latitude ranges from (45, 90).

Through the continuous dichotomy in the GeoHash algorithm, any geographic location on the Earth can be encoded into a unique binary array. A longer binary array implies a more precise geographic location. When the time of dichotomies reaches 30, the maximum error is approximately 0.0186 meters. After we get the binary array of longitude and latitude, we can generate a geographic binary array with longitude bits occupied in even digits and latitude bits occupied in odd digits. For instance, in Figure 3(A) the longitude binary code of the left top region is '0', and the latitude binary is '1', then the geographic binary array '01' can represent the left top region. More examples can be found in the appendix A.2. After the location encoding layer, a fully connected layer is added for obtaining location embeddings with the same dimension as the semantic vectors.

Let's give an example from Wikipedia (`https://en.wikipedia.org/wiki/Geohash`), the encoded longitude "0111 1100 0000" represents an area with the longitude from -5.625to -5.449 degree with a maximum error 0.044 degree (about 4,400 meters) after 12 times binary divisions (Table 6), and the encode latitude "1011 1100 1001" represents an area with the latitude from 42.539 to 42.627 (Table 5). Then we can generate a geographic binary array with longitude bits occupied in even digits and latitude bits occupied in odd digits. With this criterion, the geographic binary array of the above example can be depicted as "0110 1111 1111 0000 0100 0001".

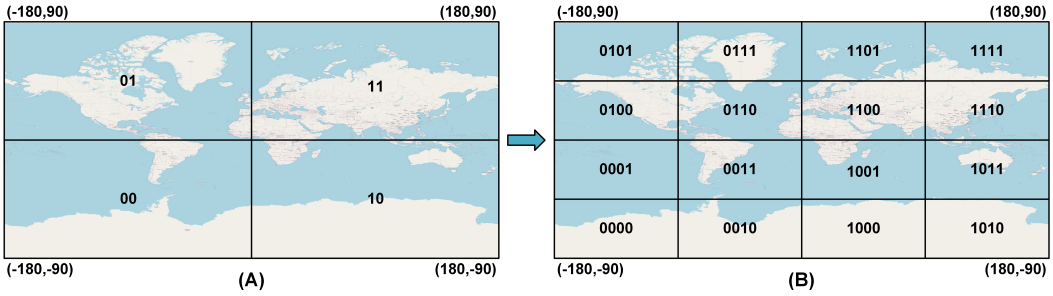


Figure 3: The illustration of binary partition in longitude and latitude.

Table 5: Examples and errors of latitude code “1011 1100 1001”.

bit position	bit value	min	mid	max	mean value	maximum error
0	1	-90.000	0.000	90.000	45.000	45.000
1	0	0.000	45.000	90.000	22.500	22.500
2	1	0.000	22.500	45.000	33.750	11.250
3	1	22.500	33.750	45.000	39.375	5.625
4	1	33.750	39.375	45.000	42.188	2.813
5	1	39.375	42.188	45.000	43.594	1.406
6	0	42.188	43.594	45.000	42.891	0.703
7	0	42.188	42.891	43.594	42.539	0.352
8	1	42.188	42.539	42.891	42.715	0.176
9	0	42.539	42.715	42.891	42.627	0.088
10	0	42.539	42.627	42.715	42.583	0.044
11	1	42.539	42.583	42.627	42.605	0.022

Table 6: Examples and errors of longitude code “0111 1100 0000”.

bit position	bit value	min	mid	max	mean value	maximum error
0	0	-180.000	0.000	180.000	-90.000	90.000
1	1	-180.000	-90.000	0.000	-45.000	45.000
2	1	-90.000	-45.000	0.000	-22.500	22.500
3	1	-45.000	-22.500	0.000	-11.250	11.250
4	1	-22.500	-11.250	0.000	-5.625	5.625
5	1	-11.250	-5.625	0.000	-2.813	2.813
6	0	-5.625	-2.813	0.000	-4.219	1.406
7	0	-5.625	-4.219	-2.813	-4.922	0.703
8	0	-5.625	-4.922	-4.219	-5.273	0.352
9	0	-5.625	-5.273	-4.922	-5.449	0.176
10	0	-5.625	-5.449	-5.273	-5.537	0.088
11	0	-5.625	-5.537	-5.449	-5.581	0.044