StructAM: Enhancing Address Matching through Semantic Understanding of Structure-aware Information

Zhaoqi Zhang[1], **Pasquale Balsebre**[1], **Siqiang Luo**[1], **Zhen Hai**[2], **Jiangping Huang**[3]
[1]Nanyang Technological University, Singapore
[2]DAMO Academy, Alibaba Group
[3]Chongqing University of Posts and Telecommunications, China
{zhaoqi001@e., pasquale001@e., siqiang.luo@, haiz0001@e., cs-huangjp@}ntu.edu.sg

Abstract

The task of address matching involves linking unstructured addresses to standard ones in a database. The challenges presented by this task are manifold: misspellings, incomplete information, and variations in address content are some examples. While there have been previous studies on entity matching in natural language processing, for the address matching solution, existing approaches still rely on string-based similarity matching or manually-designed rules. In this paper, we propose StructAM, a novel method based on pre-trained language models (LMs) and graph neural networks to extract the textual and structured information of the addresses. The proposed method leverages the knowledge acquired by large language models during the pre-training phase, and refines it during the fine-tuning process on the address domain, to obtain address-specific semantic features. Meanwhile, it also applies an attribute attention mechanism based on Graph Sampling and Aggregation (GraphSAGE) module to capture internal hierarchy information of the address text. To further enhance the accuracy of our algorithm in dirty settings, we incorporate spatial coordinates and contextual information from the surrounding area as auxiliary guidance. We conduct extensive experiments on real-world datasets from four different countries and the results show that StructAM outperforms state-of-the-art baseline approaches for address matching.

Keywords: Address Matching, Structured Information, Address Hierarchy, Graph Neural Network

1. Introduction

Nowadays, the booming e-commerce and food delivery industries rely heavily on location-based services. A well-formatted and high-quality address is a key factor in improving the quality of such services. In the logistics industry, matching original shipping addresses to standardized ones enhances distribution efficiency and reduces delivery costs. However, users on e-commerce websites may input incomplete, misspelled, and sometimes wrong addresses, making it challenging to directly match and align the data due to differences in structure and representation. This complexity poses a daunting task for couriers in delivering goods to specified locations and accurately locating delivery points. To overcome these challenges, dedicated algorithms need to be developed to handle the complexities associated with addressing information gaps, redundancies, ambiguities, and diversities in real-world address data while also allowing for error tolerance.

The problem described above is known as address matching (Drummond, 1995; Comber and Arribas-Bel, 2019), address standardisation or address validation, and is an essential component in geocoding systems. The main objective of address matching solutions consists in identifying and matching a given unstructured address with one in a structured database, as well as searching for similar or identical addresses. Most of the existing solutions (Adams and Janowicz, 2012; Karimzadeh et al., 2019), primarily employ string similarity analysis of input unstructured addresses. However, these methods are more suitable for situations where the available textual address information is clean and complete. In the so-called *dirty* settings, where input unstructured addresses may have missing information or informal place names, relying solely on textual information is no longer sufficient.

Table 1 illustrates a collection of **Match** (the given address pair points to the same location or place) and **Non-Match** (the given address pair points to different locations or places) examples of address pairs from our real-world datasets, to showcase the challenges presented by the task of address matching. In the first example, slight variations in the zipcode information (OX2 7JL $\neq$ OX2 7LJ) and house name (Portabello $\neq$ Portobello) could lead language models to split the words in different tokens and produce embeddings. However, it's crucial to emphasize that the content of the House Address column is identical. This similarity suggests that the user may have introduced some typographical errors. Furthermore, spatial coordinates suggest a distance exceeding 500 meters, potentially due to data collection errors. In the second instance, the first address corresponds to a restaurant, while the other is a hotel. However, their unstructured address content largely overlaps,

LREC-COLING 2024, pages 15350–15361
20-25 May, 2024.

Table 1: Examples of address pairs that are prone to misclassification ("-": missing information)

Label	Country	State	City	Zipcode	HouseAddress	HouseName	Address	Latitude	Longitude
Match	GB	-	Oxford	OX2 7JL	7 South Parade	Portabello	Portabello, 7 South Parade, Oxford, OX2 7JL, GB	51.9456	-1.27678
	GB	Oxfordshire	-	OX2 7LJ	7 South Parade	Portobello	Portobello, 7 South Parade, Oxfordshire OX2 7LJ, GB	51.6784	-1.26895
Non-Match	GB	Greater London	-	EC2A 3HU	81 Great Eastern St	Hoxton Grill	Hoxton Grill, 81 Great Eastern St, Greater London EC2A 3HU, GB	51.5254	-0.08299
	GB	Greater London	-	EC2A 3HU	81 Great Eastern St	The Hoxton	The Hoxton, 81 Great Eastern St, Greater London EC2A 3HU, GB	51.5257	-0.08281
Match	GB	Leicestershire	Lutterworth	LE17 4HB	16 St Johns Business Park	Elmhurst Energy	Elmhurst Energy, 16 St Johns Business Park, Lutterworth, LE17 4HB, GB	52.4562	-1.20131
	GB	-	-	-	-	-	16 St. John's Park, Lutterworth, GB	52.4508	-1.19723
Non-Match	GB	Carmarthenshire	Carmarthen	SA31 1GA	St Catherine St	-	St Catherine St, Carmarthen, Carmarthenshire SA31 1GA, GB	51.8590	-4.3094
	GB	-	-	SA31 1GA	-	St Catherine's Walk Car Park	St Catherine's Walk Car Park, SA31 1GA, GB	51.8590	-4.3095

leading to potential incorrect matches. Upon scrutinizing their distinct address hierarchy attribute details, it became apparent that the sole dissimilarity lies within the House Name column content. It's imperative to identify and amplify such minor differences between address fields when addressing the challenge of different points of interest that share similar names. In the third and final pair, despite the close spatial proximity, the vast amount of missing textual and hierarchical information poses a considerable challenge for conventional methods. From the case study, we can discern that the nearest neighbours of each address or place can offer valuable clues for address matching.

To effectively overcome the aforementioned challenges, we present StructAM, a structure-aware address matching model that can exploit address hierarchy, spatial coordinates, and neighbourhood context information. The contributions of this paper can be summarized as follows:

- We develop a fine-grained semantic address hierarchy learning module based on GraphSAGE (Hamilton et al., 2017) with an additional Attribute Attention Mechanism, which utilizes structured address attributes to capture hierarchy features.
- We incorporate a fuzzy geocode module, which relies on structured latitude and longitude inputs to acquire spatial information with the incorporation of a bucketing module to enhance our decision-making process.
- We introduce a neighbourhood context mining module based on Graph Attention Network (GAT) (Veličković et al., 2017), which leverages structured information both from the address text and the network topology of the nearest neighbours of each address to create a more contextualised representation.
- We conduct extensive experiments on four real-world datasets, and the experimental results demonstrate the advantages of our proposed method over well-established baselines, and show its robustness against the previously mentioned challenges.

2. Related Work

The rule-based standardization and normalization approach (Fan et al., 2009; Elmagarmid et al., 2014) is one of the earliest methods employed for address matching tasks. However, this human-designed technique has the limitation of relying on domain expertise and manual efforts to create and maintain rule sets. Extensive research has proven that this dependency renders static methods less effective when confronted with new data sources or matching across diverse domains. Other common strategies for address matching involve approximate string-based metrics such as the Levenshtein distance (Levenshtein et al., 1966), longest common subsequence, and n-gram (Flatow et al., 2015). Nevertheless, these string-based methods often fall short of capturing accurate contextual word information due to their primary focus on character sequence comparison, potentially yielding less precise and reliable matching outcomes.

In subsequent studies, various approaches have been proposed to address these challenges by leveraging word embedding and deep learning algorithms. These methods transform address strings into word vectors and integrate them with neural networks to quantify similarity. For instance, (Lin et al., 2020) utilized Word2Vec (Mikolov et al., 2013a) to embed words into a lower-dimensional vector space, then employed the ESIM model (Chen et al., 2016), a deep text-matching model, to infer local and global cues for determining address compatibility. Additionally, as a specific type of spatial entity, certain models designed for entity-matching tasks, such as DeepER (Ebraheem et al., 2017) and DeepMatcher (Mudgal et al., 2018), also adapted for address matching. DeepER applies Glove (Pennington et al., 2014) for word embeddings, which are used as input for LSTM (Hochreiter and Schmidhuber, 1997) to learn semantic and contextual information about entities and generate high-dimensional representations of feature vectors. Similarly, DeepMatcher uses FastText (Bojanowski et al., 2017) to embed the textual se-

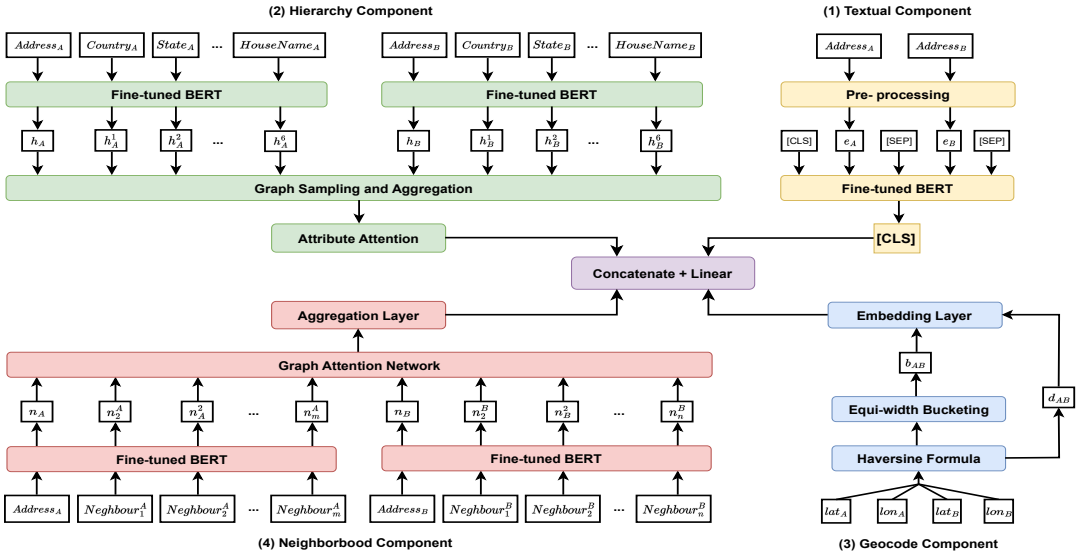


Figure 1: An overview structure of StructAM which consists of four main components: (1) ***Textual Component*** which compares textual similarity based on Fine-tuned BERT, (2) ***Hierarchy Component*** which utilizes GraphSAGE to learn inclusion relationship between administrative regions and apply Attribute Attention to determine the importance of different administrative regions, (3) ***Geocode Component*** which buckets spatial distance between the given address pair, (4) ***Neighbourhood Component*** which applies Graph Attention Network to collect neighbourhood information for the given address pairs.

quences and utilizes Recurrent Neural Networks (RNNs) with additional attention mechanisms to capture representations of sequence data between entities. Furthermore, MPM (Fu et al., 2019), Seq2Seq (Nie et al., 2019), MCA (Zhang et al., 2020), and HierMatcher (Fu et al., 2021) are modified based on DeepMatcher to improve their performance on specific datasets. However, it is important to note that deep learning models may struggle to accurately capture the complex semantic relationships between phrases in addresses, such as the ambiguity of "St." meaning "Street" or "Saint". More recent advancements include the incorporation of pre-trained language models like BERT (Devlin et al., 2018), yielding improved performance in address matching. These language models offer robust semantic representation and contextual understanding, enhancing the accuracy and robustness of address matching tasks. Examples include Ditto (Li et al., 2020), which fine-tunes Transformer-based language models on extensive text corpora, combining domain expertise, text summarization, and data augmentation for performance optimization. Nevertheless, most existing methods treat geographic coordinates as scalar features for similarity calculation, disregarding potentially valuable detailed geographic information. Some latest BERT-based studies, such as Geo-BERT (Liu et al., 2021) and Geo-ER (Balsebre et al., 2022), have sought to address this limitation by incorporating geocoding techniques. These methods associate geographic coordinates with addresses, enabling precise spatial localization for improved address matching. Geo-BERT constructs graphs incorporating spatial distance information and employs graph representation learning, while Geo-ER introduces a distance component to compute and embed spatial distance. Nonetheless, none of the aforementioned solutions has accounted for certain critical factors, such as the varying importance of administrative regions within each address and the presence of address hierarchy information in administrative affiliations between different regions.

3. Methodology

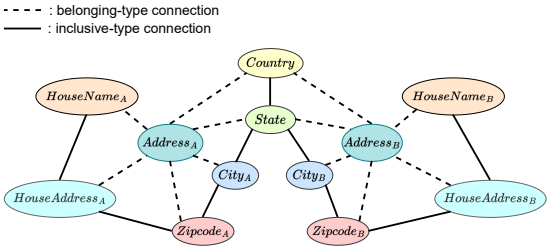
In this section, we establish our problem statement, present a comprehensive framework of StructAM illustrated in Figure 1 and expound upon StructAM, component by component in detail.

3.1. Problem Definition and Model Overview

Given two realistic addresses, denoted as $(Addr_A, Addr_B)$, our goal is to determine if they correspond to the same real-world point of interest (POI). Each address is identified by a set of textual attributes $\{Country, State, City, Zipcode, HouseAddress, HouseName, Address\}$ and geographic coordinates $\{latitude, longitude\}$.
The architecture of StructAM is summarized in Figure 1. It can be divided into four main components that jointly contribute to the final matching decision:

Country	State	City	Zipcode	House Address	House Name	Address
GB	London		SW15 1BL	Dryburgh street	Putney Leisure Centre	Putney Leisure Centre, Dryburgh street, London SW15 1BL, GB
GB	London	Putney		Dryburgh Road	DC Leisure Centre	DC Leisure Centre, Dryburgh Road, Putney, GB

(a) A candidate address pair example.



(b) A local graph example

Figure 2: The illustration of a local graph. Different coloured nodes represent address and administrative regions of different levels.

(1) *Textual Component*, which learns the textual features of the address pair, (2) *Hierarchy Component*, which captures the inclusive relationship between adjacent administrative regions and dynamically assign weights to each administrative region to obtain a new representation, (3) *Geocode Component*, which reflects geographic information of the address pair derived from geographic coordinates, (4) *Neighbourhood Component*, which collects information about the surrounding addresses.

3.2. Textual Component

The textual component aims to compare the textual parts, and we adopt Transformer-based language models to enable semantic comprehension. We follow a fined-tuned version of BERT to learn the textual information, employing the Masked Language Model and Next Sentence Prediction for both pre-training and fine-tuning the language model on our datasets. This helps to understand the textual information better and thus disambiguates it by learning contextual information. We combine the address strings of each candidate address pair $(Addr_A, Addr_B)$ as follows:

$$\text{Comb}_{Addr} = \text{[CLS]}Address_A\text{[SEP]}Address_B\text{[SEP]}$$

Through the Next Sentence Prediction task, we can consider the hidden state of the [CLS] token as the combined representation of the entire input sequence that contains information about the similarity between the address pair. Therefore, we exclusively select [CLS] token as the output for the textual component. We refer to the final encoding of the candidate address pair for the textual component as S_{text} and concatenate it with the outputs from the other components to make the final decision.

3.3. Hierarchy Component

Previous entity matching studies typically involved comparing the attributes or properties to determine their similarity. Nevertheless, distinct from other categories of entities, an address is typically described as a combination of multiple administrative regions interconnected in a specific sequence. It is evident that the significance of various administrative regions varies, and there is an inclusive relationship between them. Furthermore, considering the varying precision of different administrative levels, their importance should also differ. For instance, a *street name* is often one of the most specific parts of an address, making it more important relative to the *country*. Solely focusing on extracting the textual information from the address strings by language modelling, we will overlook the address hierarchy features mentioned earlier.

Although BERT is highly effective when applied in many NLP applications, it has limitations as it is prone to errors caused by spelling mistakes and slight variations in the input text. These limitations stem from the sensitivity of BERT to token-level changes and its reliance on the context present in the training data. Additionally, in cases where different nearby POIs share similar names, address attributes and spatial information cannot provide sufficient information. The second and first candidate pairs in Table 1 show such challenges respectively. To address these limitations, we propose a solution using Graph Neural Network (GNN) with an additional attribute attention mechanism. This approach aims to overcome the limitations of BERT in capturing subtle distinctions found in address data. By leveraging the ability of GNN to model complex interactions within the address structure, capturing the inclusive relationship and improving the model's performance in cases where traditional attribute-based matching falls short. The attribute attention mechanism dynamically assigns different weights to each administrative region when aggregating them into a new representation.

We first obtain a pair of address hierarchy textual tuples $Attr_A = \{Country_A, State_A, City_A, Zipcode_A, HouseAddress_A, HouseName_A\}$ and $Attr_B = \{Country_B, State_B, City_B, Zipcode_B, HouseAddress_B, HouseName_B\}$. Considering the inclusive relationship among various administrative regions, we construct an undirected local graph through the available geographic hierarchy information with the following rules:

- Consider each candidate address as a node of type *address*;
- Consider each administrative region, regardless of whether the information is missing, as a node of type *region* (In cases where the contents of administrative regions at the same level within the address pair are the same

string, they are merged into a single node);
- Connect each administrative region node to the candidate address nodes in this region, defining the edge type as *belonging*;
- Connect every two adjacent administrative regions, by an edge of type *inclusive*.

Figure 2 provides an illustrative demonstration, where Figure 2a displays the information regarding a candidate address pair, while Figure 2b depicts the construction of the local graph using the information from Figure 2a. We first employ BERT to train the address text and administrative region text, transforming them into the corresponding word embedding sequence that serves as the values for each node. For the k-th region-type node $node_k^A$, we identify a set of nodes $neighbor_k^A$ that share an inclusive-type edge with it. Subsequently, we utilize GraphSAGE (Hamilton et al., 2017) to obtain an updated representation for each region-type node, incorporating the inclusion information. The process is elaborated as follows:

$$n_k^A = Aggregate(neighbor_k^A) \tag{1}$$

$$r_k^A = W_a \cdot [\, e_k^A \parallel n_k^A \,] \tag{2}$$

In the above equations, the function $Aggregate(\cdot)$ performs a max-pooling operation, W_a is a learnable weight matrix and the symbol $\parallel$ denotes concatenation. e_k^A represents the original embedding of node $node_k^A$, and r_k^A signifies the updated representation of the same node, which is derived by the GraphSAGE technique. To enhance the depth of contextual information associated with each administrative region, and to capture the interrelations between these regions, we combine the representation of each node with e^A, the embedding of the address-type node with a belonging-type edge between each region-type node. This process is crucial for creating a more comprehensive contextual understanding and incorporating the connections between administrative regions into the final representation, and obtaining ultimate representation c_k^A for every administrative region:

$$c_k^A = W_a \cdot [\, r_k^A \parallel e^A \,] \tag{3}$$

We perform the same operation for each region-type node to obtain the new representations c_k^B of $node_k^B$. After executing the above procedures, every region-type node now contains textual and inclusive information. We then utilize an attribute attention mechanism to determine the significance of various administrative regions. In the local graph, both address-type nodes have k region-type neighbours. Their neighbour nodes represent administrative regions corresponding to each level. We combine the information from the k-th region-type neighbour nodes of the same level for both address-type nodes:

$$c_k = W_s \cdot [\, c_k^A \parallel c_k^B] \tag{4}$$

The pair representation in the k-th attribute is denoted as c_k. To determine the importance of each attribute, we incorporate an attribute attention mechanism. This mechanism assigns weights to different attributes, allowing us to learn hierarchy information dynamically.

$$\gamma_k = softmax(W_k \cdot c_k) \tag{5}$$

$$S_{hier} = \sum_k (\gamma_k \cdot c_k) \tag{6}$$

The normalized attention score, denoted as γ_k, essentially represents the importance of the k-th attribute in determining whether the candidate address pair matches. By calculating the normalized attention scores for all attributes, we can obtain the eventual encoding, represented as S_{hier}, for the address pair in the hierarchy component.

3.4. Geocode Component

In address matching tasks, the spatial distance between addresses is a crucial factor, which can reflect geographic proximity. However, limited by equipment accuracy and manual labour, errors may occur during the acquisition phase of spatial coordinates. This may result in two POIs pointing to the same location being misclassified as **Non-Match** due to the large spatial distance. The first candidate pair in Table 1 is a typical example. To solve this issue, we propose incorporating a geocode component that takes into account the spatial distance with a fuzzy module to identify these errors.

For a given address pair $(Addr_A, Addr_B)$, the spatial distance between them is calculated using the Haversine formula:

$$d_{AB} = Haversine(\varphi_A, \varphi_B, \lambda_A, \lambda_B, r) \tag{7}$$

In this formula, d_{AB} is the spatial distance representation, where (φ_A, λ_A) are the latitude and longitude coordinates for $Addr_A$. The variable r represents the Earth's radius. The computed spatial distance is then normalized to fall within the [-1, 1] range. By applying linear transformation and offset, we can reduce the noise and variability in the data to some degree, thereby enhancing the expressive capability of the spatial distance:

$$e(d_{AB}) = \alpha_d^{\mathsf{T}}(\frac{2 \cdot d_{AB}}{max_{dist}} - 1) + \beta_d \tag{8}$$

The equation includes two learnable parameters, α_d and β_d. The maximum distance between candidate address pairs in the dataset, denoted as max_{dist}, may vary in different datasets.

In light of the potential inaccuracies that may arise from geocoding techniques, it is important to acknowledge that the latitude and longitude coordinates obtained may not be completely precise. As

a result, this can impact the accuracy of spatial information. To address this issue, we propose implementing an Equi-width Bucketing method to assess the spatial distance between candidate address pairs. This approach involves grouping the spatial distance based on their distribution, which can help to introduce a level of fuzziness and account for any potential errors.

We allocate the specified number of buckets according to the value of max_{dist} and map the spatial distance to the corresponding bucket:

$$b_{AB} = \lfloor \ (\ 10 \cdot |d_{AB}| \ ) \ \rfloor \tag{9}$$

$$S_{dist} = W_d \cdot [\ e(b_{AB}) \parallel e(d_{AB}) \ ] \tag{10}$$

The given equation involves various mathematical functions. The notation $|\cdot|$ denotes a function that calculates the absolute value, while $\lfloor \cdot \rfloor$ signifies a rounding-down function. Subsequently, we convert the bucket variable b_{AB} to an embedded vector representation denoted as $e(b_{AB})$ and concatenate it with $e(d_{AB})$, which we represent as S_{dist} as the consequent output of the geocode component.

3.5. Neighbourhood Component

In some cases, two addresses still cannot be deemed a **Match** using only textual and distance information. Our study aims at correctly classifying address pairs that do not have rich textual information (including HouseName content), and, although in spatial proximity, are still classified as **Non-Match** by existing solutions. The condition of the last candidate pair in Table 1 can be an example of such a case. Thus, we develop a neighbourhood component to aid in addressing this kind of problem by collecting and comparing surrounding information.

We first determine two neighbour tuples for each candidate address $\mathcal{N}_A = \{\mathcal{N}_1^A, \mathcal{N}_2^A, ..., \mathcal{N}_m^A\}$ and $\mathcal{N}_B = \{\mathcal{N}_1^B, \mathcal{N}_2^B, ..., \mathcal{N}_n^B\}$ that are spatially close to each address in the candidate address pair. Addresses whose spatial distance to the candidate address pair is less than max_{dist} are regarded as neighbours. Additionally, we remove $Addr_B$ from the neighbour tuple $\mathcal{N}_A$, and vice versa to avoid the candidate addresses paying attention to each other. We construct a global graph, which amalgamates structured node and edge data:

- Consider each candidate address and the surrounding addresses with a distance of less than the max distance from it as a node;
- Connect each candidate address node to its surrounding address node (except for another address in the given address pair);
- Weight of each edge is the inverse of the spatial distance between two addresses.

We apply GAT (Veličković et al., 2017) with a single attention head to acquire a contextual representation of the surrounding address. We first perform a feature transformation on the encoding of $Addr_A$ and its neighbour $\mathcal{N}_m^A$:

$$\begin{aligned} h_A &= W_n \cdot e(Addr_A) \\ h_m &= W_n \cdot e(\mathcal{N}_m^A) \end{aligned} \tag{11}$$

To imbue the edge with meaningful relevance and distance information between nodes, thereby furnishing pivotal information for ensuing attention calculations, we employ the ensuing approach to compute the attention weight for $Addr_A$ and its neighbours $\mathcal{N}_m^A$:

$$e_{Am} = \text{LeakyReLU}(\ ([h_A \| h_m]) \ \cdot w_{Am}) \tag{12}$$

$$w_{Am} = \phi \cdot \frac{1}{d_{Am}} \tag{13}$$

where ϕ is a learnable parameter, w_{Am} corresponds to the edge weight between node $Addr_A$ and its neighbour node $\mathcal{N}_m^A$. To ensure the attention weight falls within the interval [0, 1], we apply normalization as follows:

$$\alpha_{Am} = \frac{\exp(e_{Am})}{\sum_{k \in \mathcal{N}_A} \exp(e_{Ak})} \tag{14}$$

where α_{Am} is normalized attention score. Finally, we leverage the attention coefficient to weight and sum the features of neighbour nodes to aggregate neighbour information:

$$n_A = \text{ReLU}(\sum_{m \in \mathcal{N}_A} (\alpha_{Am} \cdot h_m)) \tag{15}$$

We repeat the above operation to gain n_B and indicate S_{neigh}, concatenated by n_A and n_B as the ultimate encoding of the neighbour information. Furthermore, we concatenate S_{neigh} with the output of the first three components and feed the sequence to the classifier.

4. Experiments

To evaluate the effectiveness of our proposed approach, a series of experiments are conducted on four real-world datasets, aiming to benchmark its performance against established baseline methods. To gain deeper insights into the impact of individual components within StructAM on performance enhancement, we have also conducted an ablation study. This study seeks to isolate and analyze the specific contributions of each model component to the observed gains in performance.

Table 2: Statistics on the datasets used in the experiments. The column $Positive(\%)$ represents the number and percentage of positive pairs in the corresponding dataset. The column $Dirty(\%)$ represents the number and percentage of diry pairs (an address with three missing attributes in the address pair) in the corresponding dataset.

Country	Size	# Positive (%)	# Dirty (%)
SG	10,414	3,978 (38.20%)	6,810 (65.39%)
GB	10,688	6,944 (64.97%)	3,571 (34.29%)
MY	22,379	10,912 (48.76%)	12,264 (54.80%)
US	101,086	70,653 (69.89%)	5,261 (5.20%)

4.1. Datasets

In our data collection process, we obtained 144,567 annotated address pairs from a publicly available dataset, provided by Foursquare, with the spatial distance of each pair not exceeding 1000m. The collected datasets encompass a range of four countries, namely Singapore (SG), Great Britain (GB), the United States (US), and Malaysia (MY). We include textual and spatial information coupled with corresponding labels for each candidate address pair. The statistics for each dataset are summarized in Table 2.

4.2. Comparison Methods

We compare our work with the best-performing algorithms currently available. Following previous studies, we use the Precision (P), Recall (R) and F1 score (F1), which is computed as the harmonic mean between precision and recall as the performance metric. The results are reported in terms of the Precision, Recall and F1 score on the test set and the epoch that yields the best performance on the validation set.

- **DeepMatcher** (Mudgal et al., 2018) utilizes FastText (Bojanowski et al., 2017) to convert text-value attribute pairs into word embeddings to represent the textual information in the records and leverages RNNs with additional attention mechanisms to process, align and compare attributes.
- **Ditto** (Li et al., 2020) capitalizes on the potential of pre-trained language models, fine-tune and cast entity matching task as a sequence-pair classification problem. On top of that, three optimization techniques are developed to further improve its matching capability through injecting domain knowledge, summarizing long strings, and augmenting training data.
- **Geo-ER** (Balsebre et al., 2022) is a recent exploration in the realm of geospatial data integration, leverages the BERT (Devlin et al., 2018) as its foundation. Enhanced by a distance embedding element and a neighbourhood attention mechanism.
- **ChatGPT** released by OpenAI emerging as a notable AI language model, which is able to harness the power of zero-shot learning to tackle address matching tasks.
- **LLAMA2-7B** is an advanced large language model developed by Meta AI, which can be fine-tuned using supervised learning and reinforcement learning to improve its performance across address matching tasks.

4.3. Experimental Settings

For each dataset, we randomly partition 70%, 20%, and 10% of the address pairs as the training set, validation set, and testing set, and maintain a consistent ratio of positive to negative samples during the splitting process. We utilize *bert-base-uncased* model for pre-training. The hidden size of BERT is 768, thus both textual and hierarchy embeddings have an embedding size of 768, while the embedding size of geocode embedding is set to 512 and the embedding size of neighbourhood embedding is set to 256. During the training phase, the model is optimized by Adam optimizer, with a learning rate of 3e-5. Additionally, a linearly decreasing learning rate schedule and the number of epochs, batch size and dropout rate are set at 10 or 15 (depending on the dataset size), 32 and 0.2, respectively, to enhance generalization. All the experiments involving deep learning frameworks are executed on a V100-SXM2 GPU.
For DeepMatcher, we utilize both textual attributes and spatial coordinates as input. For Ditto, we follow the Ditto paper by fine-tuning RoBERTa (Liu et al., 2019). Additionally, we do not inject any domain knowledge for a fair comparison, turn off summarization since there are no descriptive attributes for this task, and use data augmentation

Table 3: Experimental results on Precision (%), Recall (%) and F1 score (%). Bold indicates the highest F1 score. The last row shows the improvement of StructAM with respect to the best baseline.

	SG			GB			MY			US		
Model	P	R	F1	P	R	F1	P	R	F1	P	R	F1
DeepMatcher	77.3	74.9	76.1	86.7	85.8	86.2	86.9	83.6	85.2	89.1	87.9	88.5
ChatGPT	78.5	76.3	77.4	87.4	86.2	86.8	88.0	85.5	86.7	90.4	89.0	89.7
LLAMA2-7B	79.8	76.7	78.2	88.0	86.8	87.4	88.0	86.0	87.0	89.3	91.1	90.2
Ditto	81.7	78.2	79.9	89.3	86.7	88.0	87.8	85.2	86.5	89.3	92.3	90.8
Geo-ER	85.3	79.7	82.4	91.7	88.8	90.2	89.1	85.0	87.0	91.6	92.8	92.2
StructAM	87.2	82.9	**85.0**	93.8	91.0	**92.4**	90.2	87.2	**88.7**	92.7	94.9	**93.8**
Improvement			+2.6			+2.2			+1.7			+1.6

with all the operators applied uniformly at random. For Geo-ER, we preprocess the datasets and collect the surrounding information following the formula presented by the GeoER paper based on the current datasets. For ChatGPT, we employ large-scale labelled address pairs for training to enrich ChatGPT's comprehension of address matching contexts and design the following prompt template to generate text to evaluate whether two addresses Match:

Do the following two addresses point to the same location or place? Answer with "match" if they do and "non_match" if they do not.
Address 1: {address_1}
Address 2: {address_2}

For LLAMA2-7B, we also create a prompt template and finetune the model using a large dataset of labelled address pairs. The prompt template is set as follows:

prompt_template = "You are a classification model. Based on the following two address information, you need to predict the relationship label between the given address pair from {all_labels}. One address pair has only one label.
Input address pair: {address_pair}
Output: "

4.4. Performance and Analysis

Table 3 showcases the experimental results obtained from the test data. The algorithm that achieved the highest F1 score on each dataset is emphasized in bold. For clarity, the last row demonstrates the improvement of StructAM over the comparison algorithm with the best performance. The results indicate the consistent outperformance of StructAM over the top-performing baseline algorithms by a margin of at least 1.6% and up to 2.6%, which shows that StructAM excels in solving *dirty* address matching. Based on these outcomes, we

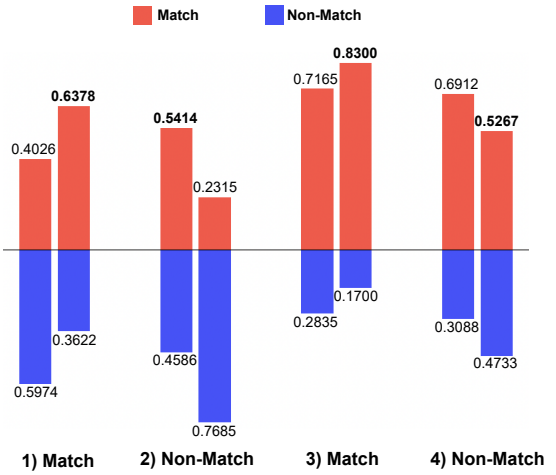


Figure 3: The comparison of prediction values between Geo-ER (left) and StructAM (right).

can draw several key comparisons and summarize them as follows.

First, it is important to note that algorithms that incorporate geospatial information (StructAM and Geo-ER) consistently yield superior performance. In contrast, DeepMatcher relies on the deep learning framework to compare attributes, which struggles to effectively capture nonlinear relationships in cases involving inter-attribute connections. While fine-tuned LLAMA2-7B outperforms zero-shot learning ChatGPT, StructurAM still surpasses fine-tuned LLAMA2-7B. This is primarily due to the inherent design differences between BERT and autoregressive language models like LLAMA2 and ChatGPT. BERT's architecture facilitates straightforward application in text classification tasks, as its [CLS] token output can be directly linked to a classification head. Conversely, LLAMA2 and ChatGPT lack this clear mechanism for classification. While it's possible to utilize these models for classification by prompting them to classify input text, this approach introduces reliability con-

cerns, as the models may simply provide an interpolated answer instead of genuinely understanding the text. Furthermore, BERT's bidirectional architecture enables it to consider both the preceding and succeeding context at each position, offering a more comprehensive comprehension of the input text. This bidirectionality proves particularly advantageous for classification tasks, providing a richer contextual understanding compared to LLAMA2 and ChatGPT's unidirectional Transformer architecture. Additionally, it can be challenging for such large language models to capture spatial information. Despite the fact that Ditto integrates pre-trained language models, it falls short of effectively capturing the semantic correlations between geospatial data and textual information. This result highlights the pivotal role of geospatial features in the address matching task and demonstrates the effectiveness of our geocode and neighbourhood component.

Secondly, we observe that StructAM consistently outperforms Geo-ER, the best baseline. This underscores the challenge of comprehensively capturing hierarchy information in addresses solely through reliance on language models. Additionally, it highlights the limitations of language models in rectifying errors stemming from spelling mistakes and subtle variations. Hence, it is necessary to design a hierarchy component that can capture the inclusive relationship between administrative regions and the significance of different administrative regions. This will significantly enhance the accuracy of address matching tasks. In Figure 3, we showcase StructAM's capability to incorporate structured information. The figure illustrates the probability values of four specific address pairs, which correspond to the examples in Table 1, calculated by Geo-ER and StructAM respectively, with the ground truth label marked below the bar chart. In the last example, the prediction result differs from the ground truth label, probably due to the absence of fine-grained address attributes.

4.5. Ablation Study

We conduct an ablation study to evaluate the impact of individual components within StructAM by systematically comparing our original framework with variants in which each component is removed one at a time. The comprehensive results of this ablation study are presented in Table 4. Our observations indicate that the model has substantial enhancements owing to the incorporation of our proposed three components. To delve into specifics, the individual impact of each component is prominently pronounced. Specifically, the geocode component demonstrates improvements ranging from 0.7% to 1.2%, boasting an average advancement of 1.0%. This suggests that incorporating spatial information into the model helps in enhancing its performance. Similarly, the neighbourhood component yields significant performance gains, exhibiting an average increase of 0.7%, with a minimum surge of 0.6% and a maximum boost of 0.8%. This indicates that this component is valuable, although the variations in improvement are relatively small, which may be due to the limitations of the address collection range, and the sparse distribution resulting in many addresses not having neighbours that meet the requirements. This highlights a potential area for future work, where improving the address collection range and incorporating other data sources might lead to even more substantial gains from this component. Notably, the hierarchy component stands out as the most impactful, driving an average surge of 1.6%, accompanied by a minimum rise of 1.2% and an impressive maximum leap of 2.0%.

Drawing insights from the ablation study, it becomes evident that each component containing structured information plays a vital role and significantly contributes to the overall outcomes. Additionally, their collective impact is greater than the sum of their individual contributions. This suggests that the components work synergistically to improve the model's performance.

Table 4: Ablation study on each dataset, one component is removed, and F1 scores (%) are reported.

	SG	GB	MY	US	Avg
StructAM	**85.0**	**92.4**	**88.7**	**93.8**	-
No Hier	-2.0	-1.8	-1.2	-1.4	-1.6
No Dist	-0.7	-1.2	-0.9	-1.0	-1.0
No Neigh	-0.8	-0.6	-0.7	-0.8	-0.7

5. Conclusion and Future Work

We introduce StructAM, an algorithm designed to elevate the precision of address matching tasks. The experimental results distinctly establish its supremacy over pre-existing methodologies. The effectiveness of our approach can be attributed to its proficient handling of structured information inherent to addresses, including address hierarchy, fuzzy geocoding, and neighbourhood context. This efficacy is further validated through ablation studies, which meticulously delineate the effectiveness of each individual component. Future work directions involve incorporating a wider array of data sources, such as street-view imagery, and developing novel algorithms to utilize visual information for enhancing the decision process and offering more precise predictions.

6. Acknowledgments

This study is supported by the Ministry of Education (MOE), Singapore, under its Academic Research Funding (MOE-T2EP20122-0003). This research is also partially supported by the Humanities and Social Sciences Fund of the Ministry of Education of China (20YJCZH047).

7. Bibliographical References

Benjamin Adams and Krzysztof Janowicz. 2012. On the geo-indicativeness of non-georeferenced text. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 6, pages 375–378.

Alfred V. Aho and Jeffrey D. Ullman. 1972. *The Theory of Parsing, Translation and Compiling*, volume 1. Prentice-Hall, Englewood Cliffs, NJ.

American Psychological Association. 1983. *Publications Manual*. American Psychological Association, Washington, DC.

Rie Kubota Ando and Tong Zhang. 2005. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6:1817–1853.

Galen Andrew and Jianfeng Gao. 2007. Scalable training of L_1-regularized log-linear models. In *Proceedings of the 24th International Conference on Machine Learning*, pages 33–40.

Pasquale Balsebre, Dezhong Yao, Gao Cong, and Zhen Hai. 2022. Geospatial entity resolution. In *Proceedings of the ACM Web Conference 2022*, pages 3061–3070.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the association for computational linguistics*, 5:135–146.

Ashok K. Chandra, Dexter C. Kozen, and Larry J. Stockmeyer. 1981. Alternation. *Journal of the Association for Computing Machinery*, 28(1):114–133.

Qian Chen, Xiaodan Zhu, Zhenhua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2016. Enhanced lstm for natural language inference. *arXiv preprint arXiv:1609.06038*.

Sam Comber and Daniel Arribas-Bel. 2019. Machine learning innovations in address matching: A practical comparison of word2vec and crfs. *Transactions in GIS*, 23(2):334–348.

James W. Cooley and John W. Tukey. 1965. An algorithm for the machine calculation of complex Fourier series. *Mathematics of Computation*, 19(90):297–301.

Grant DeLozier, Jason Baldridge, and Loretta London. 2015. Gazetteer-independent toponym resolution using geographic word profiles. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29.

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

William J Drummond. 1995. Address matching: Gis technology for mapping human activity patterns. *Journal of the American Planning Association*, 61(2):240–251.

Muhammad Ebraheem, Saravanan Thirumuruganathan, Shafiq Joty, Mourad Ouzzani, and Nan Tang. 2017. Deeper–deep entity resolution. *arXiv preprint arXiv:1710.00597*.

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

Ahmed Elmagarmid, Ihab F Ilyas, Mourad Ouzzani, Jorge-Arnulfo Quiané-Ruiz, Nan Tang, and Si Yin. 2014. Nadeef/er: Generic and interactive entity resolution. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, pages 1071–1074.

Wenfei Fan, Xibei Jia, Jianzhong Li, and Shuai Ma. 2009. Reasoning about record matching rules. *Proceedings of the VLDB Endowment*, 2(1):407–418.

David Flatow, Mor Naaman, Ke Eddie Xie, Yana Volkovich, and Yaron Kanza. 2015. On the accuracy of hyper-local geotagging of social media content. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, pages 127–136.

Cheng Fu, Xianpei Han, Le Sun 0001, Bo Chen, Wei Zhang, Suhui Wu, and Hao Kong. 2019. End-to-end multi-perspective matching for entity resolution. In *IJCAI*, pages 4961–4967.

Cheng Fu, Xianpei Han, Jiaming He, and Le Sun. 2021. Hierarchical matching network for heterogeneous entity resolution. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pages 3665–3671.

Dan Gusfield. 1997. *Algorithms on Strings, Trees and Sequences*. Cambridge University Press, Cambridge, UK.

Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Suela Isaj, Esteban Zimányi, and Torben Bach Pedersen. 2019. Multi-source spatial entity linkage. In *Proceedings of the 16th International Symposium on Spatial and Temporal Databases*, pages 1–10.

Morteza Karimzadeh, Scott Pezanowski, Alan M MacEachren, and Jan O Wallgrün. 2019. Geotxt: A scalable geoparsing system for unstructured text geolocation. *Transactions in GIS*, 23(1):118–136.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Vladimir I Levenshtein et al. 1966. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710. Soviet Union.

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

Yue Lin, Mengjun Kang, Yuyang Wu, Qingyun Du, and Tao Liu. 2020. A deep learning architecture for semantic address matching. *International Journal of Geographical Information Science*, 34(3):559–576.

Xiao Liu, Juan Hu, Qi Shen, and Huan Chen. 2021. Geo-bert pre-training model for query rewriting in poi search. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2209–2214.

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

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Interspeech*, volume 2, pages 1045–1048. Makuhari.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26.

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

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

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Mohammad Sadegh Rasooli and Joel R. Tetreault. 2015. Yara parser: A fast and accurate dependency parser. *Computing Research Repository*, arXiv:1503.06733. Version 2.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903*.

Yansong Wang, Meng Wang, Chaoling Ding, Xinghua Yang, and Jian Chen. 2022. Chinese address recognition method based on multi-feature fusion. *IEEE Access*, 10:108905–108913.

Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and

Tie-Yan Liu. 2021. Do transformers really perform badly for graph representation? *Advances in Neural Information Processing Systems*, 34:28877–28888.

Dongxiang Zhang, Yuyang Nie, Sai Wu, Yanyan Shen, and Kian-Lee Tan. 2020. Multi-context attention for entity matching. In *Proceedings of The Web Conference 2020*, pages 2634–2640.