Learning Geospatially Aware Place Embeddings via Weak-Supervision

Vamsi Krishna Penumadu*
penvamsi@amazon.com
Amazon
India

Nitesh Methani*
methani@amazon.com
Amazon
India

Saurabh Sohoney
sohoneys@amazon.com
Amazon
India

ABSTRACT

Understanding and representing real-world places (physical locations where drivers can deliver packages) is key to successfully and efficiently delivering packages to customer's doorstep. Prerequisite to this is the task of capturing similarity and relatedness between places. Intuitively, places that belong to a same building should have similar characteristics in geospatial as well as textual space. However, these assumptions fail in practice as existing methods use customer address text as a proxy for places. While providing the address text, customers tend to miss-out on key tokens, use vernacular content or place synonyms and do not follow a standard structure making them inherently ambiguous. Thus, modelling the problem from linguistic perspective alone is not sufficient. To overcome these shortcomings, we adapt various state-of-the-art embedding learning techniques to geospatial domain and propose Places-FastText, Places-Bert, and Places-GraphSage. We train these models using weak-supervision by innovatively leveraging different geospatial signals already available from historical delivery data. Our experiments and intrinsic evaluation demonstrate the significance of utilizing these signals and neighborhood information in learning geospatially aware place embeddings. Conclusions are further validated by observing significant improvements in two domain specific tasks viz., Pair-wise Matching (recall@95precision improves by 29%) and Candidate Generation (avg recall@k improves by 10%) as evaluated on UAE addresses.

CCS CONCEPTS

• **Computing methodologies** → **Natural language processing**; **Machine learning approaches**; • **Applied computing** → **Transportation**; • **Information systems** → **Language models**; **Top-k retrieval in databases**.

KEYWORDS

Place Embedding, Address Embedding, Graph Neural Networks, Weak Supervision, Address Matching, Link Prediction, Geoproximity, Geospatial BERT

*Both authors contributed equally to this research.

SIGSPATIAL '22, November 1–4, 2022, Seattle, WA, USA

ACM ISBN 978-1-4503-9529-8/22/11.
https://doi.org/10.1145/3557915.3561016

ACM Reference Format:
Vamsi Krishna Penumadu, Nitesh Methani, and Saurabh Sohoney. 2022. Learning Geospatially Aware Place Embeddings via Weak-Supervision. In *The 30th International Conference on Advances in Geographic Information Systems (SIGSPATIAL '22), November 1–4, 2022, Seattle, WA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3557915.3561016

1 INTRODUCTION

Last-mile delivery that involves all logistics activities related to the delivery of packages to customer's doorstep, has become extremely crucial for multiple e-commerce and delivery businesses. The key to plan an optimal and efficient delivery is understanding various characteristics of real-world places (any physical location where a delivery can be made). For e.g., knowing a group of units is part of the same physical building can enable package consolidation and attribute sharing, reducing the overall cost for making a successful delivery. Similarly, knowing that three separate buildings have a common mailroom can help in planning the time taken to deliver the package. However, in order to learn places representation, majority of the existing delivery systems [18, 25] rely primarily on customer address text - the only mandatory input from customers that conveys information about their whereabouts. In this work, we argue that places exhibit far more complex geospatial relationships that cannot be learnt by looking at the address text alone. Simply put, an address is just a text containing apartment, street and locality information arranged in a specific order whereas a place is a physical real-world entity (house, apartment complex, university campus, library, landmarks etc.) that is associated with a set of geospatial (latitude-longitude, address, bounding box polygon) and business-contextual attributes (residential/commercial place-type, operating hours, delivery preferences, etc.). A multi-building campus having a common mailroom for package delivery is a great example of a place with complex geospatial relationships which can not be learned from the address text. Existing systems' inability to handle place relationships leads to sub-optimal delivery planning, resulting in an increased package delivery time and multiple back and forth trips for drivers.

Learning from address text is even more challenging in emerging geographies such as United Arab Emirates (UAE), India and Mexico, where customer addresses are loosely structured and do not adhere to standard address writing guidelines, making addresses inherently ambiguous. For e.g., *Khalifa City B* and *Shakhbout* refer to the same sub-locality within the larger Khalifa neighbourhood of Abu Dhabi city in UAE. Customers use these synonyms interchangeably, making such ambiguous addresses challenging to comprehend. The problem becomes severe when customers miss-out on key components in the address lines. For e.g., the address *Street 1,*

Al Ghadeer Village Dubai does not contain any unit or building information making it difficult to learn a fine-grained representation for the place. This leads to a question on how to learn feature rich and spatially aware place embeddings from ambiguous and incomplete addresses. Another important category worth considering are cold-start addresses, i.e., addresses for which there is limited or no historical delivery information available. For such cases, customer provided address text is the only source of information available during inference. This makes it challenging as embedding models have to learn to generate spatially aware embeddings in the absence of geospatial signals.

Existing places learning systems [2, 18] rely on approaches like Word2Vec[20], FastText [5] and Bert [9]. These models produce address embeddings from linguistic contexts and have shown promising results on address classification [2], address clustering [14], and address matching [25] tasks. However, they can be easily influenced by the textual similarity and other geography specific nuances present in the addresses. Consider address pairs with negative match verdict shown in Table 1. While all these pairs refer to different physical locations, FastText and Bert give high cosine similarity score to them. This exposes the inability of language learning models to capture geospatial interactions between places. Some efforts have been initiated in this direction [25] but they rely on traditional techniques like Word2Vec which are unfit as the generated embeddings are context free and static in nature [5].

To address the aforementioned shortcomings, we innovatively leverage different geospatial signals like geohashes and geoproximity and provide domain-specific supervision to FastText [5] and Bert [9]. This is one of the key contribution of our work where we effectively utilize these signals and adapt existing state-of-the-art embedding models to geospatial domain. We refer to these models as Places-FastText and Places-Bert as their final embeddings represent a real-world place and not just an address text. Additionally, since no manual labelling is required, these models can be trained in a completely weakly-supervised fashion, without introducing any new complexity during training. The significance of utilizing geospatial signals for learning embeddings is highlighted from Table 1 where our proposed variants give relatively low cosine similarity score to geospatially distant address pairs, when compared with the vanilla FastText and Bert models. We further validate our approach by evaluating Places-FastText and Places-Bert on two extrinsic tasks viz., Pair-wise Matching and Candidate Generation, on UAE addresses. While we take UAE geography as the motivation to improve upon, our learnings and observations are still applicable to other similar geographies where relying purely on address text is not sufficient. Empirical results demonstrate that our proposed variants work better than the state-of-the-art methods on both the tasks.

To further improve the speed and performance, we turn towards graph-based learning approaches that beautifully capture the interactions between neighboring places and learn feature rich embeddings even for incomplete addresses. This is another key contribution of our work where we adapt the existing work proposed in [11] and introduce Places-GraphSage, a novel way of combining geospatial domain knowledge with state-of-the-art language modelling and graph learning techniques. Our experiments show that whenever neighborhood information is available, Places-GraphSage significantly outperforms Places-FastText and Places-Bert. This ability of learning from neighbors is critical for emerging geographies like UAE where considerable portion of the overall traffic has incomplete and ambiguous addresses. To give a complete picture, we also evaluate all the variants on cold-start addresses i.e., addresses for which geocode and neighborhood information is not available at the time of inference.

In summary, the major contributions of our work are: (1) We highlight the difference between a place and an address, and motivate the need for using different geospatial signals to learn feature rich and spatially aware place embeddings as opposed to learning embeddings from address text alone; (2) We adapt three state-of-the-art models to geospatial domain and propose Places-FastText, Places-Bert, and Places-GraphSage that can be trained in a completely weakly-supervised fashion, without introducing any additional complexity during training; and (3) Our best configuration improves the performance of pair-wise matching by 29% and candidate generation by 10% which results in huge cost savings for multiple delivery planning systems.

2 RELATED WORK

2.1 Language Models

Language models are a natural choice to learn feature rich embeddings from customer address text. Traditional models like GloVe [21] and Word2Vec [20] are not directly applicable in geospatial domain as they have multiple limitations, e.g., static word representation which can appear in different contexts, inability to deal with vernacular words resulting in out-of-vocabulary (OOV), misspellings and polysemy. FastText [5] generates embeddings for OOV words by leveraging the sub-word information. However, it can not adjust the word embeddings based on its context. To address the challenges around context sensitivity, series of large pre-trained language models [6, 9, 16, 22, 23] have been introduced in the literature. In [2, 14, 18], the authors used different language models including recent state-of-the-art Bert-variants [8, 9, 16] for range of geospatial tasks including address classification, address clustering, Points of Interest (POI) mining and learning contextual embeddings for addresses. However, as opposed to our work, such approaches do not make use of any geospatial signals that are implicitly embedded in Last Mile delivery systems and are crucial for learning place embeddings.

2.2 Graph Neural Networks

Recent advancements in learning on graphs [11, 17, 24, 29] have shown the potential to leverage graph connectivity to learn effective representations of sentences. Embeddings learnt from graphs are being applied to multiple NLP tasks [3, 4, 7, 10, 12, 19, 26]. Authors in [11] proposed a method capable of generating node embeddings even for nodes that were unseen during training. [17] combines contextual information from Bert embeddings with global information learnt from GNNs to learn better embeddings. As studied by the authors in [27], a new trend of combining large pre-trained language models with GNNs have started where complimentary strengths of both the domains is combined to achieve state-of-the-art accuracy on different tasks. One such work is of [25] which learns geographical semantic representations for address strings

by combining Word2Vec+LSTM embeddings with Graph Convolutional Networks (GCNs). However, our work differs from [25] on three aspects: (1) our models leverage sub-word and contextual features using recent state-of-the-art techniques like FastText and Bert, (2) we rely on GraphSage, as opposed to GCNs, which is capable to generate embeddings even for unseen nodes, and (3) in addition to matching, we evaluate our embeddings against multiple real-world scenarios like incomplete addresses and cold-start addresses which are relevant for last mile delivery use cases.

2.3 Place Embeddings

Most research on place embedding originates by augmenting Points-of-Interest (POI) representations with the geospatial context. The authors of popular Place2Vec work [28] approach this problem from an information theoretic perspective. This method adjusts the geographic distance between places based on the uniqueness of place types within a certain distance as well as their popularity as a proxy for human activities. [13] further improves by using geohash information with the Place2Vec model. [30] construct POI-based spatial context by using the nearest neighbor approach and learns high-dimensional POI representations using the skip-gram training framework. While the above techniques exist, they have a different focus compared to our study as they utilize semantic relationships between different place types. For instance, [28] aims to learn embeddings such that Nightclubs and Bars are closer in latent space as compared to bakeries and barber shops. Our work focuses on learning representations for all delivery locations against popular POIs which introduces challenges in the form of data sparseness, lack of standardization, vernacular content and large scale. In this work, we focus on relatedness w.r.t. geospatial proximity and attempt to learn Place embeddings by leveraging historical delivery information available in abundance.

3 GEOSPATIAL SIGNALS

In this section, we describe four geospatial signals extracted from our in-house historical delivery datasets. We use these signals to provide linguistic as well as geospatial context to all our models.

3.1 Customer address

Customer address is a free-form text provided by customers to locate them. It contains information like house/apartment number, street name, locality/sub-locality information, landmarks, city, state, and postal-code. We run our experiments on a sample of few million UAE addresses for which a successful delivery was made in past few years. Before training, we do basic pre-processing like upper-casing the text, handling whitespaces, punctuations and splitting alphanumeric characters interspersed into each other (e.g., *B123,Khalifa CityB* $\rightarrow$ *B 123 , KHALIFA CITY B*). However, some inconsistencies are still left after the pre-processing (e.g., spelling mistakes, compound words, etc.), which we expect our models to overcome while learning place representations.

3.2 Geocodes

Geocodes are the latitude and longitude of a delivery point (DP) where a driver confirms a delivery. We use geocodes to compute the distance between two places and put a hard threshold of few meters to get the neighborhood information of a place. However, geocodes are not always exact and reliable. For instance, DP geocodes could be anywhere between the customer's door and the delivery van parked down the road. Additionally, in city "canyons", where line of sight to GPS satellites is severely limited and in emerging geographies where GPS quality is not up to the mark, the GPS measurement error can be substantial. To mitigate these errors, we consider only those addresses for which we have made atleast few successful deliveries in the past and aggregate them to arrive at a single geocode for the address.

One way to effectively use geocode signals is to compute the geohash of a place. Geohash is a geocoding system that encodes the geographic location of a place into a short string of letters and digits [13]. It guarantees that longer the shared prefix between two geohashes, the more spatially closer they are. In this work, we append address texts with geohashes obtained from their DP geocodes to explicitly provide the geospatial context to the embedding models.

3.3 Stop-sharing

Stop-sharing data informs us about the group of places that were served in a single-vehicle stop by the driver. These can be individual units inside an apartment, adjacent buildings or apartments, garden communities, etc. This information is primarily captured in our systems for better route planning and to avoid short transits, U-turns and parking costs for future deliveries. In this work, we use this information to learn neighborhood features while generating place embeddings. Note that unlike geocodes, stop-sharing doesn't put any hard threshold on distance to compute proximity and represents a binary label to define a relationship between two places.

As evident from the description, training with these signals do not require any manual annotations showcasing one of the key contributions of our work i.e., training geospatially aware models via weak-supervision.

4 LEARNING PLACE EMBEDDINGS

In this section, we discuss various models and propose modifications to adapt existing FastText, Bert and GraphSage models to generate spatially aware embeddings by exploiting geospatial signals discussed above. The gains we get from these alterations will become evident from the results of our experiments. Note that in real-world scenarios, historical delivery records are not available for a large chunk of addresses. Hence, for all our models (except baseline), we propose two variants: (1) The vanilla variant which utilizes geospatial and neighborhood information only during training and (2) The extended variant (denoted by ++) which leverages the information during training as well as inference. We also present the time and space complexity statistics for various models in Section 6.

4.1 Address-FastText (Baseline)

This is the original FastText [5] model pre-trained on a sample of few million UAE address lines. It takes address text as input and produces a 300-dim embedding vector by taking an average over the embeddings of the individual word vectors. Note that it does not use any geospatial signal during training and inference.

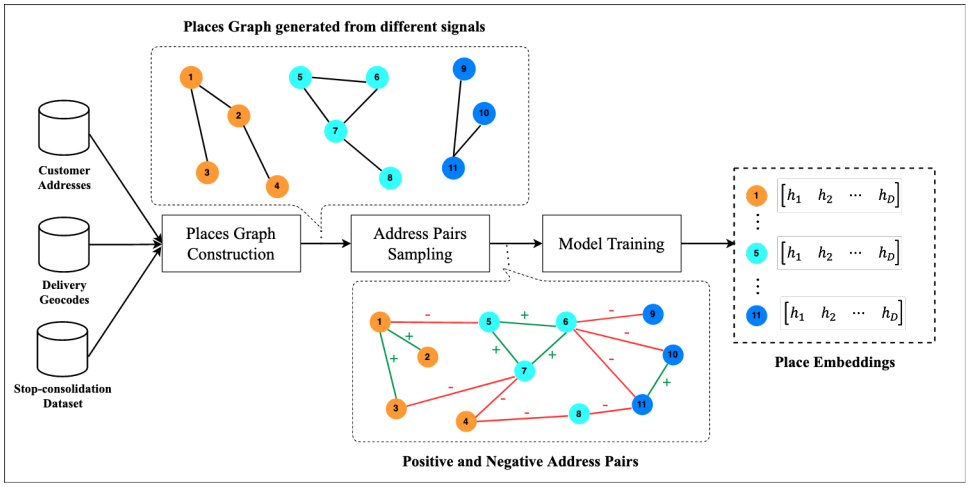


Figure 1: High-level overview of our proposed Places-GraphSage framework. It has three important stages. Firstly, we construct a places graph having few million nodes and few tens of millions of edges. The graph is constructed using stop-sharing and geoproximity signals. Secondly, we sample positive and negative address pairs from the places graph to enable training at such a scale. Finally, we train GraphSage and learn node embeddings corresponding to the places.

4.2 Places-FastText

This is the geospatially adapted FastText variant where we augment address text with their corresponding geohashes during training. This provides extra supervision to FastText in learning geospatially aware place embeddings in addition to the textual features coming from the address text. For a fair comparison with the baseline, we train both the variants, Places-Fasttext and Places-Fasttext++, to generate place embeddings.

4.3 Places-Bert

This is built on top of the bert-base-uncased architecture [9] with a difference that instead of using originally proposed Masked Language Modelling (MLM) and Next Sentence Prediction (NSP) as auxiliary tasks, it only uses a slightly modified version of NSP during pre-training. We do not use MLM head as we noticed that it spends most of the training time to restore masked words which can be numbers (apartment or building numbers) or proper nouns (locality, city or state names) in the address text. A similar observation was made by the authors in [1] in the context of sentiment analysis. While MLM training may be useful for learning using natural language sentences, adapting it for addresses is counter-intuitive (refer Appendix A.1 for Places-Bert performance using MLM task). To enable Bert training in geospatial context, we propose Address Pair Similarity as auxiliary task, an innovative variant of NSP where we generate positive and negative address pairs from geocodes and stop-sharing datasets. In particular, when choosing an address pair <a,b> as a training record, half of the time we pick a pair that was served in a single vehicle stop by our drivers and use that as a positive pair. Similarly we pick pairs which were served in different stops as negatives. This allows Bert to better capture the notion of places geospatial proximity as opposed to original MLM and NSP tasks. We further empower Places-Bert to explicitly capture the notion of geospatial proximity by appending geohashes at the end of the address texts. We also train Places-Bert++ that utilizes geohashes at the time of inference.

For training, we use Adam optimizer [15] with an initial learning rate of 5e-5 and a batch size of 128 for 6 epochs. To generate the sentence embeddings, we take the average of token embeddings from the last layer and generate a 768-dim feature vector. From our experiments (see Appendix A.1 for reference) we observe that embeddings from the last layer gives the best performance when compared to CLS token embedding as well as the aggregated embeddings from the last four layers.

4.4 Places-GraphSage

In this model, we combine linguistic and geospatial information in the form of a graph and generate place embeddings by learning low-dimensional feature representation of the nodes in the graph. We refer to this model as Places-GraphSage inspired from [11]. As shown in Figure 1, Places-GraphSage consists of three key components viz., (1) places graph construction algorithm where we establish relationships between places using different signals, (2) edge sampling strategy which generates a batch of positive and negative address pairs to train a Graph Neural Network (GNN), and (3) computationally efficient node embedding generation algorithm. We formulate this as a link-prediction task where model learns by predicting whether two nodes in a network are likely to be associated or not. One key advantage of this approach is that it allows us to generate embeddings even for the nodes that were unseen during training. This is critical for constantly evolving geographies like UAE where we see considerable portion of the overall traffic coming from new places. In the subsequent sections, we describe each of these components in detail.

4.4.1 Places Graph Construction: In this stage we construct a places graph as a homogeneous graph of few million nodes and few tens of millions of edges where nodes represent the real-world places

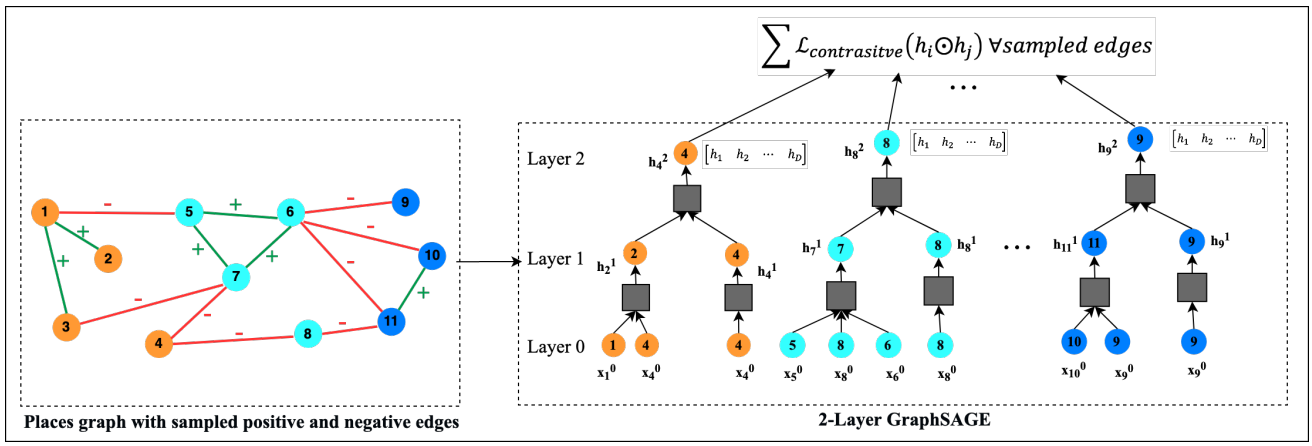


Figure 2: Illustration of training Places-GraphSage on sampled +ve and -ve address pairs. In the figure, x_i^0 represents the initial FastText features for node i and h_i^j represents the transformed feature representation for the i-th node in j-th layer.

and edges represent the geospatial relationships between them. In particular, initially a node is represented with address embedding learnt using state-of-the-art language models like FastText and Bert, pre-trained on UAE address corpus. An edge is present between two nodes if atleast one of the two conditions is satisfied: 1) two places are served by drivers in a single vehicle stop, as per our stop-con dataset, or (2) two places are in very close proximity of each other, as per our geocodes dataset. We consider places to be in a close proximity if they belong to the same geohash grid of size 38.2m x 19m. Thus, places graph, by design, is equipped with linguistic features (in the form of node representations) and spatial features (in the form of graph connectivity).

4.4.2 Edge Sampling Strategy: By default, all the edges in the places graph are positive edges as they represent a valid relationship between two nodes. But to train GNNs, we need a strategy to include negative edges in the existing graph. One naive way is to extract all the connected components and introduce an edge between all the nodes of all disconnected components. We see two shortcomings with this approach: (1) it creates trivial negative edges between nodes from different localities, cities or even states, (2) it adds a bias during training as model can just rely on locality, city, and state level features and still achieve good performance. Other features like unit, apartment, building, and street will be either completely ignored or will have a very minimal contribution in the final loss. To address these shortcomings, we develop an edge sampling strategy that exploits geocodes and locality+city information from the address text. Specifically, for every node, we take all its neighbors within few hundred meter distance and randomly select K nodes that belong to a different connected component. Such a hard constraint ensures all the sampled nodes belong to the same city. We further ensure that 50% of the K nodes are from the same locality, forcing the model to learn unit, apartment, building and street level features along with the locality and city information.

4.4.3 Node Embedding Generation: The final stage is to train a GNN that accurately predicts an edge between two nodes. We adopt the same methodology as proposed in [11] and adapt it for places learning as shown in Figure 2. In particular, all the nodes are first initialized with the address embeddings generated from a pre-trained language model. Each node then aggregates the representation of its neighboring nodes into a single vector and concatenates it with its current representation. This concatenated feature vector is further transformed using a fully connected layer to generate final node embeddings. The parameters of the network are updated by minimizing the contrastive pair-wise loss between the nodes associated with the positive and negative edges. It ensures that model learns the geospatial semantics represented between edges and encodes that information in the learnt node embeddings. Two key advantages of this method are - (1) **Scalability:** In most of the GNN architectures, node embeddings are generated by looking at the entire neighborhood. It makes them extremely expensive for our use case, given the size of the places graph. To address this gap, we use GraphSage [11] which uniformly samples a fixed-size set of neighbours keeping the memory and expected runtime of a single batch fixed. This makes it scalable for learning on graphs with millions of nodes and edges; (2) **Embeddings for unseen places:** [11] generates embeddings even for unseen places for which no neighborhood information is available, by transforming its initial node features with the learned aggregation functions. To let the model do better on these cases, we mask neighborhood information in some records during training and force the model to still predict the positive and negative labels. This makes it suitable for generating place embeddings for evolving geographies where a large number of shipments pertain to new addresses.

We train this model by initializing node features with 300-dim address embeddings generated from Address-FastText. We use mean as a aggregation function to combine the information from the neighbors and transform it to a 300-dim embedding vector which represents the place embeddings for the corresponding node. To train this model at scale, we sample a small percentage of edges out of all the edges in the places graph. We use Adam optimizer with a learning rate of 0.001 to train the shallow neural network.

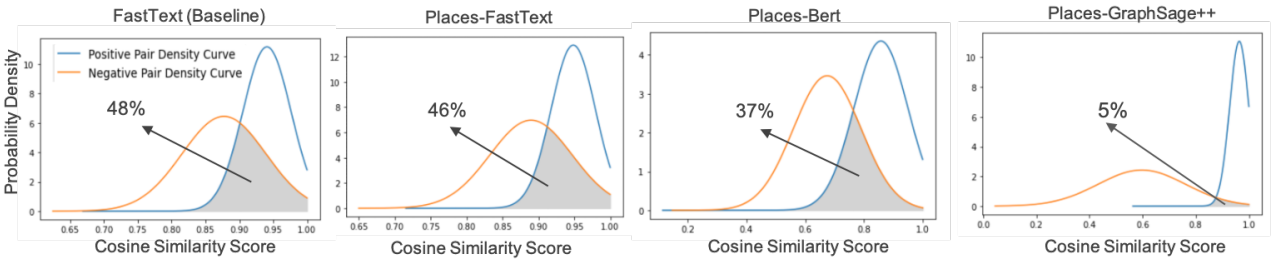


Figure 3: Evaluating similarity in place embeddings generated by different models. The overlapping area represents models inability to differentiate between the related and unrelated places. There is a consistent reduction in the overlap as we move towards geospatially rich embedding models.

Note that unlike previously discussed models, this model by design is capable of using neighborhood information while generating embeddings. Thus, for a fair comparison with other approaches, we only use neighborhood information during training in our first variant, called as Places-GraphSage. We further propose Places-GraphSage++ that utilizes neighborhood information during training as well as inference.

5 EVALUATION

5.1 Intrinsic Evaluation

We start with evaluating the ability of the embedding learning models to differentiate between the related and unrelated places. As mentioned before, we consider proximity as a measure of relatedness in this work. For a qualitative evaluation, in Table 1, we present sample address pairs with actual on-earth distance between them (geoproximity) and the predicted cosine similarity values obtained from various models. Clearly, geospatially rich embeddings are able to better predict the relatedness even with misleading textual signals. For a quantitative evaluation, we sample a few thousand address pairs of each type (related and unrelated) from the same weakly-supervised dataset. We generate place embeddings and look at the distribution of the cosine similarity between pairs (Figure 3). We observe consistent reduction in the overlap between the positive and negative probability density curves as we move towards geospatially rich embeddings. An interesting phenomenon is observed in the Places-GraphSage++ curve where with cosine similarity for positive pairs peak around the value 1 with very small variance compared to the negative curve. Unlike that the negative curve appears to be more spread out which aligns with the way we generate negative examples - sampling few from same locality and others from same city.

5.2 Extrinsic Evaluation

We evaluate the usefulness of various place embeddings for downstream package delivery systems that benefit from capturing geospatial proximity. As such relatedness in form of proximity can be defined at different granularities - 1) fine-grained, e.g. if two places belong to the same building or community, or 2) coarse-grained, e.g. if two places belong to the same locality/district. In this work we focus on fine-grained pair-wise matching and particularly look at two models - a high-precision model that performs pair-wise matching and a high-recall model that generates candidates for matching.

5.2.1 Pair-wise Matching. We utilize an in-house built pair-wise matching model which is trained to predict if two given addresses belong to the same building/community. Solving pair-wise matching at a finer granularity is important for various applications, such as de-duplication of address dataset, enabling attribute sharing (e.g., working hours, security codes, etc.), and optimal delivery planning (e.g., through location sharing). The existing model is trained using gradient boosting and utilizes a list of hand-crafted features generated from address text - 1) individual address features such as address length, number of alphanumeric tokens, 2) pair-wise features such as edit distance, phonetic distance, and 3) embedding representation of the two address texts. The model is trained and evaluated on a manually curated pair-wise ground truth. In its original form the model uses Address-FastText embeddings described in Section 4 . We replace these embeddings with other proposed embeddings and observe improvement in matcher performance. We use R@95P (recall at 95% precision) and AUC-PR for match class as metric to measure model performance. Note that due to confidentiality reasons we cannot reveal our true operating point and the actual evaluation set. However, these metrics as well as the sample set on which we report the numbers can still be considered for an unbiased evaluation of the embedding models.

5.2.2 Candidate Generation. Candidate generation is a precursor task to pair-wise matching. Reducing the search space through a high-recall model is a standard practice in entity matching solutions. Given a query address, which is to be resolved, the objective of candidate generator is to fetch top-K similar addresses from the address catalogue. Similar to Pair-wise matching, we utilize an in-house built candidate generator model that is capable of retrieving candidates at scale given the embedding space. This is proven to be a SOTA technique in the industry as this design scales to millions of addresses. Similar to the matcher, the baseline setting uses Address-FastText embeddings and we replace those with other embedding techniques and measure their effectiveness in improving candidate generator's recall. Note that curating a dataset for measuring recall is non-trivial and practically challenging as annotators have to filter matching candidates from a large database of addresses. Alternatively, for this evaluation we make use of the same manually curated matcher ground-truth dataset. We obtain a few thousand

Table 1: Example address pairs with actual on-earth distance (geoproximity) and cosine similarity obtained from various embedding learning models. Geospatially rich embeddings are able to better predict the similarity even with misleading textual signals. The fine-grained information in the address text is masked to hide actual customer addresses.

Address 1	Address 2	Verdict	Geospatial Distance (in meters)	Address FastText	Places FastText	Places Bert	Places GraphSage++
XXX, YYY Tower, Business Bay	Business Bay, YYY Tower, 13th Floor	Positive	12	0.96	0.97	0.93	0.97
KKK street, YYY Tower, Dubai Marina	LLL Street, YYY Tower VV, Dubai Marina	Positive	28	0.9	0.97	0.93	0.99
Apartment VV, YYY Residence, Barsha Z, Al barsha	Maple 1, XXX estate, Villa VV, Al Barsha	Negative	6711	0.9	0.94	0.7	0.69
XXX Road, Apartment VV, YYY Tower, DIFC	KKK street, Room no. VV, ZZZ hotel, Motor City	Negative	18522	0.85	0.89	0.61	0.44

Table 2: Performance of different embedding models on Pair-wise Matching and Candidate Generation tasks. Note that our in-house matching and candidate generator models use Address-FastText in their original form. Places-GraphSage++, our best performing model, provides improvement of 36% and 4.5% over the baseline for Recall@95 precision and AUC-PR metrics, respectively.

	Model	Matching		Candidate Generation
		R@95P	AUC-PR	AvgR@K
(a)	Address-FastText (baseline)	32.93	89.14	64.22
(b)	Places-FastText	33.62	89.72	66.29
(c)	Places-Bert	51.1	91.34	71.48
(d)	Places-GraphSage	50.17	91.96	70.21
(e)	Places-Bert++	54.28	92.04	72.89
(f)	**Places-GraphSage++**	**62.57**	**93.53**	**74.36**

positive pairs and evaluate the models' ability to recall the right candidate in top-K neighbors (K varies from 10 to 1000 with a step size of 50). It is worth mentioning that both the pair-wise matching and candidate generation models are integral part of multiple delivery planning systems currently deployed in production and serving real-traffic on a daily basis. Hence, any improvement observed in them should directly convert to significant customer impact and millions of dollars of savings for the downstream systems.

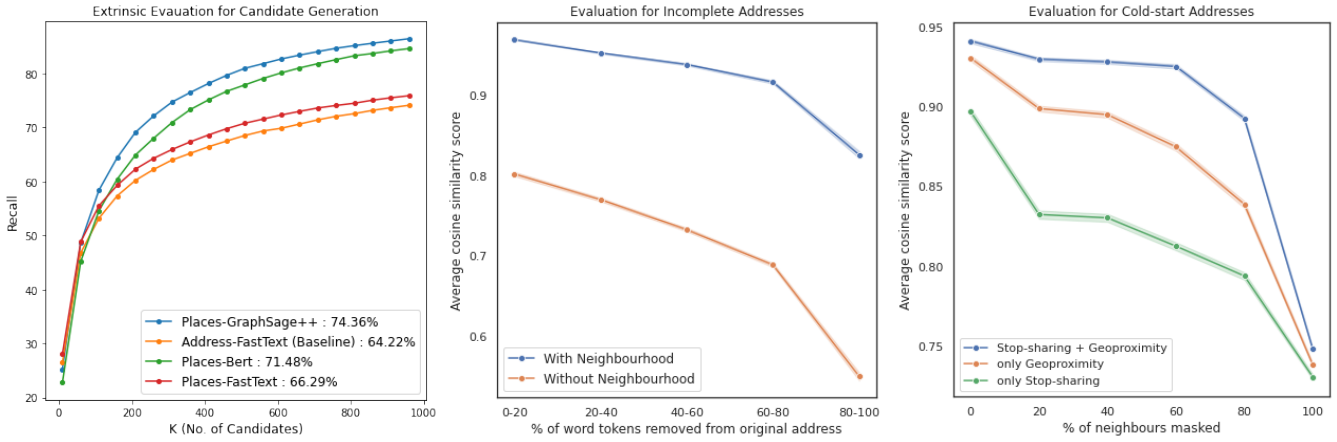
Results: We compare the performance of six different embedding techniques on Pair-wise Matching and Candidate Generation tasks and present Recall and AUC-PR numbers in Table 2. We also present recall at different number of fetched candidates in the plot in Figure 4a. In Table 2, we first focus on the top two rows where addition of geohash to the address text allows to learn better FastText embeddings. Performance improves significantly as we move towards Bert and GraphSage configurations as they capture contextual and neighborhood information along with the address text. For GraphSage we try two different configurations of initial node embeddings - FastText based and Bert based. We only report numbers with FastText based initialization as due to slightly better performance, simplified design and lower runtime. Within rows (c) and (d), we do not find a consistent winner, however, we recommend preferring GraphSage over Places-Bert due to significant runtime advantages as discussed in Section 6. Moving to rows (e) and (f), the models that utilize geospatial information at the time of inference show significant gains across all the metrics. For this configuration GraphSage is able to out-perform the Bert variant. It can be concluded that GraphSage captures the geospatial knowledge in a more systematic and effective way than Bert where geohashes are appended to the address lines. Overall, Places-GraphSage++, our best performing model provides 36% and 4.5% improvement over the current production baseline for R@95 and AUC-PR metrics, respectively.

6 MISCELLANEOUS EVALUATIONS

6.1 Against incomplete addresses

Given the package delivery use-case in emerging geographies, two subsets of our test dataset need special attention. One of them is shipments against incomplete addresses. As mentioned in Section 3, due to their free-form nature, customer addresses tend to miss-out on tokens (both important or unimportant) and can affect the learning systems in an adversarial manner. We observe a significant portion of such addresses in the UAE address dataset that we evaluate our model against. However, categorizing these addresses based on a completeness score can be non-trivial to do automatically and expensive to do manually. We simulate this behaviour with the help of an in-house address parser and evaluate the robustness of our best performing model against addresses which have significant portion of tokens missing. In particular, we remove X% (X varies from 0% to 100% with a step size of 20) of the address text tokens from Street, Building, Landmark and Locality components identified by the address parser. We sample a few thousand addresses and two matching candidates using the same weak-supervision technique used in previous experiments. We then modify each sampled address by removing X% tokens randomly, where X is obtained by sampling from a uniform distribution, and compare average cosine similarity between its two neighbors. We repeat the process for Places-GraphSage and Places-GraphSage++ to study how much neighborhood information is able to compensate for the loss of information in address text.

In Figure 4b, dominance of the blue curve (with neighbourhood) over the orange curve (without neighbourhood) speaks for the robustness that we gain by looking at neighbourhood information along with text. We would like to highlight the right most point in the blue curve where even with 80-100% of tokens missing, we retain close to 85% of the similarity information through the use of neighbourhood. Note that despite such improvements, we can not replace the need for address texts with the neighborhood information as our input mechanisms still heavily rely on address text and sufficient neighbourhood information is available only after a few successful deliveries. This poses another question on how sensitive are the embeddings to the amount of neighbourhood



(a) (b) (c)

Figure 4: (a) CG Recall for various K values where K ranging from 10 to 1000 in the step size of 50. Legend in the figure presents AvgR@K values (in %) corresponding to each experiment. (b) Demonstrating the significance of effectively leveraging other geospatial signals and neighborhood information in learning place embeddings for incomplete addresses. Values in x-axis represents various buckets (ranging from 0 to 100 in the steps of 20) on which the average cosine similarity values are computed. (c) Evaluation of GraphSage++'s ability to identify neighbours with varying amount of geospatial/neighbourhood signals. The values in x-axis represents the percentage of neighborhood information masked during inference for generating place embeddings.

information or geospatial signal available during inference. We answer this through our next analysis.

6.2 Against cold-start addresses

Apart from the incomplete addresses, the second important test subset of interest is the cold-start addresses, specifically, addresses with limited or no historical delivery information available during inference. Our curated ground truth is constructed with addresses served in the past which do not fall in the cold-start category. Hence, we simulate the required behaviour by artificially masking X% (X varies from 0% to 100% with a step size of 20) of the neighbourhood information. Similar to the previous experiment, we sample a few thousand addresses, perform the masking and then compare the new embeddings' quality through average cosine similarity with two true neighbours for each of the address. Drop in average similarity from 0.94 to 0.74 in the blue curve (Figure 4c) shows the contribution of neighbourhood information which we already observed in the main results. We make similar observations from Table 2. The top four rows correspond to the cold-start addresses as none of these models use any geospatial signals at the time of inference. While we do not see a clear winner amongst the two, we observe significant improvements across the two tasks - pair-wise matching R@95P and candidate generation AvgR@K improve by 18% and 6%, respectively, over the baseline. It can be concluded that both Places-Bert and Places-GraphSage are able to learn useful geospatial knowledge during training and use it effectively even for addresses which do not have neighbourhood information available.

Table 3: Demonstrating the significance of various weak-supervision signals on Pair-wise Matching and Candidate Generation tasks.

Sr. No.	Model	Pair-wise Matching		Candidate Generation
		R@95	AUC-PR	AvgR@K
(a)	Stop-sharing	51.27	91.72	69
(b)	Geoproximity	**64.3**	93.51	68.44
(c)	(a) + (b)	62.57	**93.53**	**74.36**

6.3 Comparing weak-supervision signals

As mentioned in Section 4.4.1, we use two types of weak supervision signals to induce geospatial knowledge during graph construction - stop-sharing and geoproximity. To understand their individual contributions, we present results by masking the signals one by one as shown in the Figure 4c and Table 3. Stop-sharing signal alone seems to be lagging behind the other two settings and Geoproximity alone appears to be doing better for R@95P metric. We can attribute lower performance of stop-sharing signal to the noise introduced by driver behaviour and our ability to extract stop-sharing information effectively from historical delivery logs. While GPS noise and driver compliance issues are also applicable in case of geoproximity, we are able to overcome those with smaller geohash grids to define relatedness. It is also observed that the noise in stop-sharing signals only affects the matching task as we observe significantly better AvgR@K when using both the signals. This observation suggests the superiority of using geoproximity data when matching at a fine-grained level. We further validated this by experimenting with a pair-wise matcher that matches at a finer level than the community

Table 4: Time and Space complexity estimates of various place embeddings models. Training and Inference rate are presented as number of records processed per second.

Places Embedding Models	Places FastText	Places Bert	Places GraphSage
Training rate (#rec. per sec)	6000	350	7000
Inference rate (#rec. per sec)	12000	664	15000
Model size (in GB)	2	0.42	0.01
Embedding size	300	768	300

where we observe the gap between using geoproximity alone *v/s* in combination with stop-sharing widens further.

6.4 Time and Space Complexity Evaluation

We evaluate various proposed models on their runtime and model sizes which plays a crucial role in model selection for production usage. From Table 4, we observe Bert to be dominated by the other two variants and this can be attributed to the deep transformer based architecture of Bert. On the other hand, GraphSage with less number of model parameters and a highly efficient parallel implementation of Graph convolutions which enables processing of as high as 15k records per second for inference which translates to a latency of 67 ms per record. Given the other observations on the recall gains, GraphSage combined with FastText (to initialize the node embeddings) turns out to be a clear winner for use in production environment.

7 CONCLUSION

In this work, we exposed the shortcomings of existing last mile delivery systems that learn places embeddings by relying only on address text information. To overcome their limitations, we adapted three SOTA models to geospatial domain using weak-supervision coming from different last-mile delivery signals like geocodes and stop-sharing. We presented results to prove superiority of using geospatial signals in place learning and moving from FastText to Bert and Graph Neural Network (GNN) based techniques. We particularly discussed the capability of GNN's to leverage neighborhood information and learn better place embeddings even for incomplete addresses. Our best performing model gives around 29% and 10% improvement in performance on pair-wise matching and candidate generation tasks, respectively. It is also faster and lighter making it the right choice for production environments and resulting in huge cost savings for multiple downstream tasks. We also simulated experiments to evaluate performance of our best performing model, Places-GraphSage++, on cold-start addresses and incomplete addresses. We observe that both Places-Bert and Places-GraphSage are able to learn geospatial information during training and use it effectively during inference.

Our work opens up the possibility for combining geospatial signals from other domains like Maps and Delivery Attributes to further learn better place embeddings. One such possible direction is to generate place embeddings from heterogenous graphs having edges representing relationships beyond geoproximity. Another promising direction is to use the learnings from our work and extend it to support other place relationships like semantic relationships between place types (e.g., embeddings for hospitals and pharmacies being closer in latent space as compared to schools and bakeries).

REFERENCES

[1] Nikolay Arefyev, Dmitry Kharchev, and Artem Shelmanov. 2021. NB-MLM: Efficient Domain Adaptation of Masked Language Models for Sentiment Analysis. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (Eds.). Association for Computational Linguistics, 9114–9124. https://doi.org/10.18653/v1/2021.emnlp-main.717

[2] T. Ravindra Babu, Abhranil Chatterjee, Shivram Khandeparker, A. Vamsi Subhash, and Sawan Gupta. 2015. Geographical address classification without using geolocation coordinates. In *Proceedings of the 9th Workshop on Geographic Information Retrieval, GIR 2015, Paris, France, November 26-27, 2015*, Ross S. Purves and Christopher B. Jones (Eds.). ACM, 8:1–8:10. https://doi.org/10.1145/2837689.2837696

[3] Jasmijn Bastings, Ivan Titov, Wilker Aziz, Diego Marcheggiani, and Khalil Sima'an. 2017. Graph Convolutional Encoders for Syntax-aware Neural Machine Translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, Martha Palmer, Rebecca Hwa, and Sebastian Riedel (Eds.). Association for Computational Linguistics, 1957–1967. https://doi.org/10.18653/v1/d17-1209

[4] Daniel Beck, Gholamreza Haffari, and Trevor Cohn. 2018. Graph-to-Sequence Learning using Gated Graph Neural Networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, Iryna Gurevych and Yusuke Miyao (Eds.). Association for Computational Linguistics, 273–283. https://doi.org/10.18653/v1/P18-1026

[5] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomás Mikolov. 2017. Enriching Word Vectors with Subword Information. *Trans. Assoc. Comput. Linguistics* 5 (2017), 135–146. https://transacl.org/ojs/index.php/tacl/article/view/999

[6] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. *CoRR* abs/2005.14165 (2020). arXiv:2005.14165 https://arxiv.org/abs/2005.14165

[7] Deng Cai and Wai Lam. 2020. Graph Transformer for Graph-to-Sequence Learning. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*. AAAI Press, 7464–7471. https://ojs.aaai.org/index.php/AAAI/article/view/6243

[8] Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net. https://openreview.net/forum?id=r1xMH1BtvB

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *CoRR* abs/1810.04805 (2018). arXiv:1810.04805 http://arxiv.org/abs/1810.04805

[10] Patrick Fernandes, Miltiadis Allamanis, and Marc Brockschmidt. 2019. Structured Neural Summarization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net. https://openreview.net/forum?id=H1ersoRqtm

[11] William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. *CoRR* abs/1706.02216 (2017). arXiv:1706.02216 http://arxiv.org/abs/1706.02216

[12] Hanqi Jin, Tianming Wang, and Xiaojun Wan. 2020. SemSUM: Semantic Dependency Guided Neural Abstractive Summarization. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*. AAAI Press, 8026–8033. https://ojs.aaai.org/index.php/AAAI/article/view/6312

[13] Jiaqi Jin, Zhuojian Xiao, Qiang Qiu, and Jinyun Fang. 2019. A Geohash Based Place2vec Model. In *2019 IEEE International Geoscience and Remote Sensing Symposium, IGARSS 2019, Yokohama, Japan, July 28 - August 2, 2019*. IEEE, 3344–3347. https://doi.org/10.1109/IGARSS.2019.8898375

[14] Vishal Kakkar and T. Ravindra Babu. 2018. Address Clustering for e-Commerce Applications. In *The SIGIR 2018 Workshop On eCommerce co-located with the 41st*

International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2018), Ann Arbor, Michigan, USA, July 12, 2018 (CEUR Workshop Proceedings, Vol. 2319), Jon Degenhardt, Giuseppe Di Fabbrizio, Surya Kallumadi, Mohit Kumar, Andrew Trotman, Yiu-Chang Lin, and Huasha Zhao (Eds.). CEUR-WS.org. http://ceur-ws.org/Vol-2319/paper8.pdf

[15] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. https://doi.org/10.48550/ARXIV.1412.6980

[16] 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. *CoRR* abs/1907.11692 (2019). arXiv:1907.11692 http://arxiv.org/abs/1907.11692

[17] Zhibin Lu, Pan Du, and Jian-Yun Nie. 2020. VGCN-BERT: Augmenting BERT with Graph Embedding for Text Classification. In *Advances in Information Retrieval - 42nd European Conference on IR Research, ECIR 2020, Lisbon, Portugal, April 14-17, 2020, Proceedings, Part I (Lecture Notes in Computer Science, Vol. 12035)*, Joemon M. Jose, Emine Yilmaz, João Magalhães, Pablo Castells, Nicola Ferro, Mário J. Silva, and Flávio Martins (Eds.). Springer, 369–382. https://doi.org/10.1007/978-3-030-45439-5_25

[18] Shreyas Mangalgi, Lakshya Kumar, and Ravindra Babu Tallamraju. 2020. Deep Contextual Embeddings for Address Classification in E-commerce. *CoRR* abs/2007.03020 (2020). arXiv:2007.03020 https://arxiv.org/abs/2007.03020

[19] Diego Marcheggiani, Jasmijn Bastings, and Ivan Titov. 2018. Exploiting Semantics in Neural Machine Translation with Graph Convolutional Networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 2 (Short Papers)*, Marilyn A. Walker, Heng Ji, and Amanda Stent (Eds.). Association for Computational Linguistics, 486–492. https://doi.org/10.18653/v1/n18-2078

[20] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. *CoRR* abs/1310.4546 (2013). arXiv:1310.4546 http://arxiv.org/abs/1310.4546

[21] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Doha, Qatar, 1532–1543. https://doi.org/10.3115/v1/D14-1162

[22] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *CoRR* abs/1802.05365 (2018). arXiv:1802.05365 http://arxiv.org/abs/1802.05365

[23] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR* abs/1910.01108 (2019). arXiv:1910.01108 http://arxiv.org/abs/1910.01108

[24] Michael Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2017. Modeling Relational Data with Graph Convolutional Networks. https://doi.org/10.48550/ARXIV.1703.06103

[25] Shuangli Shan, Zhixu Li, Qiang Yang, An Liu, Lei Zhao, Guanfeng Liu, and Zhigang Chen. 2020. Geographical address representation learning for address matching. *World Wide Web* 23, 3 (2020), 2005–2022. https://doi.org/10.1007/s11280-020-00782-2

[26] Danqing Wang, Pengfei Liu, Yining Zheng, Xipeng Qiu, and Xuanjing Huang. 2020. Heterogeneous Graph Neural Networks for Extractive Document Summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel R. Tetreault (Eds.). Association for Computational Linguistics, 6209–6219. https://doi.org/10.18653/v1/2020.acl-main.553

[27] Lingfei Wu, Yu Chen, Kai Shen, Xiaojie Guo, Hanning Gao, Shucheng Li, Jian Pei, and Bo Long. 2021. Graph Neural Networks for Natural Language Processing: A Survey. *CoRR* abs/2106.06090 (2021). arXiv:2106.06090 https://arxiv.org/abs/2106.06090

[28] Bo Yan, Krzysztof Janowicz, Gengchen Mai, and Song Gao. 2017. From ITDL to Place2Vec: Reasoning About Place Type Similarity and Relatedness by Learning Embeddings From Augmented Spatial Contexts. In *Proceedings of the 25th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, GIS 2017, Redondo Beach, CA, USA, November 7-10, 2017*, Erik G. Hoel, Shawn D. Newsam, Siva Ravada, Roberto Tamassia, and Goce Trajcevski (Eds.). ACM, 35:1–35:10. https://doi.org/10.1145/3139958.3140054

[29] Liang Yao, Chengsheng Mao, and Yuan Luo. 2018. Graph Convolutional Networks for Text Classification. *CoRR* abs/1809.05679 (2018). arXiv:1809.05679 http://arxiv.org/abs/1809.05679

[30] Wei Zhai, Xueyin Bai, Yu Shi, Yu Han, Zhong-Ren Peng, and Chaolin Gu. 2019. Beyond Word2vec: An approach for urban functional region extraction and identification by combining Place2vec and POIs. *Comput. Environ. Urban Syst.* 74 (2019), 1–12. https://doi.org/10.1016/j.compenvurbsys.2018.11.008

A APPENDIX

A.1 Abalation Study on Bert

Table 5: Abalation study on Bert models. Average of last layer embedding strategy from Public Bert finetuned on Address Pair Similarity task outperforms all other variants.

Sr.No	Places-Bert Variants	Place Embeddings Extraction Strategies	Pair-wise Matching				Candidate Generation
			Campus Level		Building Level		
			R@95	AUC-PR	R@95	AUC-PR	Avg_R@K
(a)	Public Bert	CLS token embedding	20.88	88.53	89.52	40	10.22
		avg of last layer embeddings	33.56	89.51	90.27	41.42	35.77
		avg of last four layers	29.58	88.88	89.98	38.16	32.78
(b)	Public Bert finetuned on MLM task	CLS token embedding	40.34	89.9	90.42	41.27	59.79
		avg of last layer embeddings	30.82	89.43	90.41	43.79	52.56
		avg of last four layers	29.64	88.75	89.85	41.73	51.59
(c)	Public Bert finetuned on Address Pair Similarity task (Places-Bert)	CLS token embedding	26.07	89.44	90.45	39.86	26.23
		avg of last layer embeddings	**51.1**	**91.34**	**91.86**	**53.31**	**71.48**
		avg of last four layers	46.66	91.08	91.76	49.1	72.9

We evaluate three different variants of Bert against Pair-wise matching and candidate generation tasks in three different settings. The performance of all 3x3 experiments is shown in Table 5. It can be observed that general Bert as it is does not perform well on geo-proximity tasks. Although finetuning Bert using Masked Language Modelling (MLM) shows slight improvement over general Bert, it is not well-suited for addresses (refer Section 4 in the main paper). Hence, we rely on Address Pair Similarity task where we generate address pairs from stop-sharing dataset and finetune Bert on UAE addresses. We refer to this model as Places-Bert. In order to generate sentence level embeddings, we take the average of token embeddings from last layer as it is clearly a winner when compared with CLS embeddings and average embeddings from last four layers.