Learning Geolocation by Accurately Matching Customer Addresses via Graph based Active Learning

Saket Maheshwary
Last Mile, Amazon
mahsaket@amazon.com

Saurabh Sohoney
Last Mile, Amazon
sohoneys@amazon.com

ABSTRACT

We propose a novel adaptation of graph-based active learning for customer address resolution or de-duplication, with the aim to determine if two addresses represent the same physical building or not. For delivery systems, improving address resolution positively impacts multiple downstream systems such as geocoding, route planning and delivery time estimations, leading to an efficient and reliable delivery experience, both for customers as well as delivery agents. Our proposed approach jointly leverages address text, past delivery information and concepts from graph theory to retrieve informative and diverse record pairs to label. We empirically show the effectiveness of our approach on manually curated dataset across addresses from India (IN) and United Arab Emirates (UAE). We achieved 9.3% absolute improvement in recall on average across IN and UAE while preserving 95% precision over the existing production system. We also introduce *delivery point (DP) geocode learning* for cold-start addresses as a downstream application of address resolution. In addition to offline evaluation, we also performed online A/B experiments which show that when the production model is augmented with active learnt record pairs, the delivery precision improved by 7.84% and delivery defects reduced by 12.32% on an average across shipments from IN and UAE.

CCS CONCEPTS

• **Computing methodologies** → **Active learning settings**; • **Information systems** → *Entity resolution*; • **Applied computing** → Transportation.

KEYWORDS

Active Learning, Entity Matching, Graph Theory, Geocoding

ACM Reference Format:
Saket Maheshwary and Saurabh Sohoney. 2023. Learning Geolocation by Accurately Matching Customer Addresses via Graph based Active Learning. In *Companion Proceedings of the ACM Web Conference 2023 (WWW '23 Companion), April 30-May 4, 2023, Austin, TX, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3543873.3584647

1 INTRODUCTION

Entity matching (EM), also known as entity resolution (ER), aims at identifying and linking different representations of the same real-world entities across databases. EM is a challenging task for real-world applications, particularly when entities are highly unstructured [33] and of low quality, for example, when there is lack of completeness and consistency in their descriptions. Further, real-world EM tasks [19, 29] have limited access to labeled data and require substantial labeling effort to learn accurate EM models. For delivery systems, customer addresses play an important role in delivery planning, as the address is the primary source of information provided by customers regarding their location. Customers provide their addresses in text fields at their own discretion, which may or may not follow a fixed pattern. Address writing styles and patterns are idiosyncratic in the same way as hand writing or signatures. This leads to a lot of variation in similar addresses and their components (unit, building, road, locality). Customers often use references to neighbourhoods, landmarks or points-of-interest (POI). For example, it is common for customers in India (IN) to provide colloquial addresses that use landmarks and other POI to denote the place, for example, *ABG Bank, Opp. Network Stone, Mahapurii.* Other customers may provide more structured addresses that intend to indicate the same place but also conform to local postal standards, for example, *Plot No. 438 Taj Towers, ABG Bank, Mahapuri.* In the aforementioned examples[1], both addresses refer to the same place in *Mahapuri* neighbourhood. The first example mentions *Network Stone* building as a landmark, refers *Opp.* for opposite and mentions *ABG Bank* as the place. In the second example, the number and name of the building *Taj Towers* is used to mention the same *ABG Bank* as the place. Further, neighborhood provided by a customer can also be known by other vernacular names or be a part of a larger neighborhood. For example, *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 customer address even more challenging to comprehend.

Address resolution aims to de-duplicate a query address against a set of candidate addresses in the database via pair-wise address matching. Creating a representative training set for pair-wise matching is challenging for customer addresses for multiple reasons — (1) Data distribution is skewed towards negative pairs, i.e. no-match. (2) Based on an analysis carried out by our data curation team, the average handle time (AHT) for an annotator to label a customer address pair is high; *four* times higher on average when compared to other EM tasks. (3) The component values in addresses are vernacular, redundant, noisy or missing, thus leading to unstructured data problems. (4) No appropriate pre-labeled data exists to bootstrap classifiers, nor rules to automatically label training data through weak supervision. In some real-world classification tasks, it is possible to create ground truth labels automatically at large scale via

[1] Examples in this paper are modified to preserve the privacy of customers.

WWW '23 Companion, April 30-May 4, 2023, Austin, TX, USA

ACM ISBN 978-1-4503-9419-2/23/04.
https://doi.org/10.1145/3543873.3584647

rules and heuristics using distant supervision [32] or data programming [41], while for a large number of real-world applications such as resolution of unstructured addresses, manual labeling of data cannot be avoided. Labeling a large volume of pairs for a variety of scenarios in EM does not scale, hence prior studies has adopted active learning (AL). AL aims to focus the labeling effort on the most informative records that will maximize the performance of the model, thus reducing annotation cost. But due to the variety of challenges posed by customer addresses, the existing AL techniques are insufficient and cannot be utilized in their current form.

Our proposed approach jointly leverages address text, GPS points from past deliveries, concepts from graph theory, namely graph partitioning, graph cuts and transitivity along with active learning to sample informative and diverse record pairs to minimize the cost of annotation. Our empirical evaluation show significant improvements in pair-wise matching and delivery point (DP) geocoding metrics compared to the existing production system and other state-of-the-art baselines. Further, it should be noted that the structure of addresses are quite different for IN and UAE, hence the improvements across both geographies confirm the wide applicability and generic nature of our approach. In summary, our main contributions are — (1) We propose a novel adaptation of graph-based active learning to tackle a real-world problem of customer address resolution, particularly pair-wise matching. (2) We jointly leverage address text, geospatial properties of addresses along with concepts from graph theory to retrieve informative record pairs. Our query strategy utilizes *disagreement* and *geospatial-diversity* to select record pairs to label in a data-efficient manner. (3) We deployed our approach to production and show its impacts on DP geocoding, a fundamental business problem that enables delivering packages in a cost-effective manner.

2 RELATED WORK

The widely used uncertainty-based methods leverage the prediction scores to select difficult examples for annotation [11, 22] whereas diversity-based sampling exploits heterogeneity in the feature space [2, 5]. Hybrid approaches that combine uncertainty and diversity [13, 42, 46] tackle the limitation of acquiring redundant and easy samples from earlier discussed sampling techniques, but some studies [30, 31, 38] found it to be less effective for EM tasks. Due to the lack of reusable EM models, crowd sourcing [21] is leveraged. But in real-world applications where data is domain-specific or confidential, crowdsourcing becomes challenging and incurs high costs. Many earlier works have also used AL with a variety of classifiers and explainable ER rules [39, 40]. Some studies have effectively used graph based techniques [3, 37] for various real-world applications. Different signals of the graph structure, such as the cluster representation and density are used for refining uncertainty-based sampling strategy [3]. Another work on multi-source resolution [37] used graph algorithms as well. But due to the variety of challenges posed by customer addresses, the existing techniques cannot be utilized in their current form for our problem space. Later, DTAL [19] tackled EM in low-resource settings and proposed learning a deep neural network via active learning with uncertainty sampling along with partitioning. None of the aforementioned studies consider transformer-based techniques for EM. Later, Mussmann et al. [34] proposed to use adaptive retrieval with BERT-based embedding model, namely COSINEBERT to greedily maximize the number of positive examples. Last year, DIAL [17] used an Index-By-Committee framework, where each committee member learns representations via RoBERTa based pre-trained models.

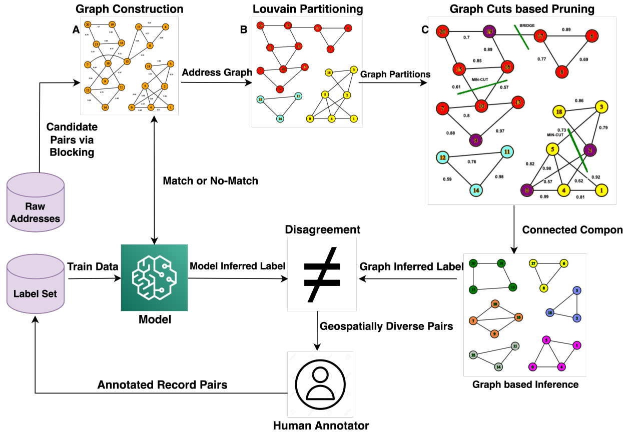


Figure 1: Demonstrates the workflow of our approach

3 METHODOLOGY

Problem Statement In our problem domain, a *match* represents an address pair belonging to the same physical building whereas *no-match* represents an address pair referring to different buildings. We consider a pool-based setting [17, 19] for active learning where unlabeled record pairs $\mathcal{P}$ are generated via blocking over customer address database $\mathcal{A}$. At each iteration of AL, it selects a batch of $\mathcal{K}$ instances and add them to the labeled corpus $\mathcal{L}$, removing them from $\mathcal{P}$. We perform AL for $\mathcal{T}$ iterations. Our task is to effectively sample $\mathcal{K}$ record pairs to be labeled by human annotators from $\mathcal{P}$, such that after re-training it on the acquired labeled pairs $\mathcal{L}$, the performance on an unseen test set is *maximized.* The details of graph theory concepts are discussed in Appendix A.1. Next, we discuss our approach as shown in Figure 1.

Blocking Comparing every address in the database to every other address (Cartesian product) is not scalable. We use ElasticSearch [15], with fastText embeddings [6, 16, 36] to index [28, 44] the addresses and then filter those addresses that are an obvious no-match (addresses that belong to a different district, state or postal code). We retrieve *top-k* candidates for every customer addresses to get a pool of unlabeled candidate record pairs $\mathcal{P}$ and apply the trained classifier to determine pair-wise matching. The details of the classification model are discussed in Appendix A.4.

Graph Construction An undirected and weighted graph G with no self-loops, vertices V, and edges E is a pair (V, E), that is $E \subseteq \{\{u, v\} \mid u, v \in V \wedge u \neq v\}$. Each address is represented via a node and the edge between two nodes is determined based on prediction of a trained model $\mathcal{M}$. Given $\mathcal{P}$ retrieved via blocking and its corresponding predictions $\psi(\mathcal{P})$ inferred via $\mathcal{M}$, we construct an address graph G. We add an edge for every matching record pair, while we skip the edge for every non-matching pair. The weights assigned to an edge is equal to the predicted probability score learnt by $\mathcal{M}$. We leverage *transitivity* of an address graph G to discover *false negatives* from base model predictions. However, given that the edges of the graph are derived from the predictions of $\mathcal{M}$, and $\mathcal{M}$ is not always accurate, a wrongly predicted match edge can lead to a series of *false positive* record pairs.

Algorithm 1 Retrieve informative records via graph based AL

```
Require: Labeled set $\mathcal{L}$, Customer Address database $\mathcal{A}$, Query budget $\mathcal{T}$, Record
    pairs per query $\mathcal{K}$, Classification Model $\mathcal{M}$, $CC_{G_{final}} = \{\}$
1:  $\mathcal{P} \leftarrow blocking(\mathcal{A})$                         ▷ Retrieve unlabeled record pairs
2:  for $i = 1, ..., \mathcal{T}$ do
3:      $\psi(\mathcal{P}) \leftarrow$ Train classifier $\mathcal{M}$ on $\mathcal{L}$ to infer for $\mathcal{P}$
4:      $G \leftarrow build\_graph(\psi(\mathcal{P}))$
5:      $G_{louvain} \leftarrow louvain\_partition(G)$
6:      $CC_{G_{louvain}} \leftarrow connected\_components(G_{louvain})$
7:      for $component$ $in$ $CC_{G_{louvain}}$ do
8:          $V_{s-t} \leftarrow \{\{x, y\} : x, y \in component \wedge (x \neq y) \wedge (\exists path(x, y) \in
    G_{louvain}) \wedge (geospatial\_proximity(x, y) > \mathcal{N})\}$
9:          $E_{mincut} \leftarrow min\_cut(V_{s-t})$
10:         $G_{mincut} \leftarrow$ Remove all edges in $E_{mincut}$ from $G_{louvain}$
11:         $G_{pruned} \leftarrow$ Remove "bridge" edges from $G_{mincut}$
12:         $CC_{G_{pruned}} \leftarrow connected\_components(G_{pruned})$
13:     $CC_{G_{final}} \leftarrow CC_{G_{final}} \cup CC_{G_{pruned}}$
14:     $\psi(\mathcal{G}) \leftarrow$ "Match" label to node-pairs within same CC else "No-Match"
15:     $X_{disagreement} \leftarrow \psi(\mathcal{G}) \neq \psi(\mathcal{P})$
16:     $\mathcal{X} \leftarrow MGRS(X_{disagreement})$
17:     $\mathcal{L} \leftarrow \mathcal{L} \cup \mathcal{X}$ , $\mathcal{P} \leftarrow \mathcal{P} \setminus \mathcal{X}$          ▷ Update labeled set and unlabeled pool
18: return final classification model trained on updated $\mathcal{L}$
```

Graph Partitioning and Graph Cuts We use graph partitioning and graph cuts to find and remove likely false positive edges from the graph and obtain smaller connected components (CC) so that the set of nodes within the same CC represent addresses from the same physical building. After constructing an address graph G, we apply a single pass of Louvain algorithm [4] to separate the nodes into multiple mutually exclusive graph partitions $G_{louvain}$. Louvain was preferred as it does not require us to input the number of partition sizes before execution. Further, a single pass of Louvain is a *linear operation* in terms of the number of edges of the graph, thus allowing it to scale across millions of edges in a graph. We then determine the CC of $G_{louvain}$. For each component in $CC_{G_{louvain}}$, we use graph cuts to prune weak links and isolated components (Appendix A.1.2). We leverage minimum cut [9] and bridges [1] as graph cut techniques to prune the *likely false positive* edges from the graph. For a given CC, we iterate over all the node pairs and retrieve those node pairs V_{s-t} where the *geospatial proximity* is greater than a pre-defined threshold $\mathcal{N}$ (Appendix A.1.3 and A.1.4). This ensures that the pair of extracted nodes are likely to belong to two different physical buildings. We compute the min-cut for all node pairs V_{s-t} and then remove min-cut edges from the graph $G_{louvain}$ to get a graph G_{mincut}. Further, we identify and remove *bridge edges* from the graph G_{mincut} to finally get a pruned graph G_{pruned}. In order to avoid creating too many small components, we only remove the bridge edges connecting nodes that have at least three neighbours each. (Algorithm 1, line 5-11).

Query Strategy To learn a graph label $\psi(\mathcal{G})$, we first compute all the CC in G_{pruned}. For all node pairs belonging to the same CC, we assign a *match* label else *no-match* label is assigned. The address pairs with graph label $\psi(\mathcal{G})$ *different (disagreement)* from the model prediction $\psi(\mathcal{P})$ hint towards nuanced matching patterns that are not yet learnt by the model. This can occur under the following two scenarios. First, if the record pair has been predicted as no-match by $\mathcal{M}$ but due to graph transitivity, the graph inferred label is match. Second, if the record pair has been predicted as a match by $\mathcal{M}$ but the corresponding edge was removed via graph partitioning and graph-cuts. From the record pairs in disagreement, we select an equal number of likely false-negatives and likely false-positives to prevent skewness in the data distribution. To ensure *geospatial-diversity* across pairs in disagreement, we sample across a grid based on the Military Grid Reference System (MGRS) [35] to capture samples across different regions. In each iteration, we select $\mathcal{K}$ informative pairs to be labeled by the human annotators. We augment labeled data to the initial manually curated training data. The augmented train set is used to re-train $\mathcal{M}$ and is evaluated on the same unseen test data. (Algorithm 1, line 12-18).

4 EXPERIMENTAL EVALUATION

Curated Ground Truth (CGT) We did stratified sampling of addresses to cover all the address writing styles and abbreviations across the country. The selection also ensured to consider medium and low address volume districts, thus accounting for the varied density of addresses, i.e. probable urban vs rural split. We generate close to $15K$ unique address pairs each for IN and UAE, which are then manually labeled by the annotation team.

Evaluation Settings In blocking, we use fastText model trained on customer addresses. For each address, we pick *top-k* similar records, with $k = 200$ and generated *tens of millions* of unlabeled pairs $\mathcal{P}$ as the output of blocking step across the geography. The number of AL iterations $\mathcal{T}$ is set to 10 and number of records $\mathcal{K}$ to sample per query is 500. On an average, the graph constructed for each geography had close to $3MM$ nodes and around $15MM$ edges.

Evaluation Metrics We split the CGT data in 70-10-20 for training, validation and testing. All the models in Table 1 were evaluated on same CGT test set. To align with downstream application, a high precision (atleast 95% precision of match class) pairwise-matching model is required. We evaluate the matching model across two metrics, namely — (1) Overall pair-wise accuracy (Accuracy), and (2) Recall at 95% Precision (R@95P). Table 1 reports the performance of all the models across these metrics on CGT test dataset. The R@95P numbers are corresponding to the match class to align the performance of the model with the downstream application.

Learning Model Baselines The details of *Production Model* are discussed in Section 3. *Ditto* [23] is a BERT-based model fine-tuned on the CGT train set. Table 1 reports the performance of these models on the same CGT test set for IN and UAE.

Active Learning Baselines *Random Sampling* retrieve random pairs from unlabeled pool $\mathcal{P}$ whereas *Confidence Sampling* select uncertain [45] record pairs. *ER Rules* proposed use of tree based models with rules [20, 27, 38]. We used our domain knowledge, and historic delivery data to design the rules. *ALMSER* [37] used graph algorithms for multi-source matching. *Deep Transfer AL* (DTAL) [19] used transfer learning in low-resource settings and proposed to learn a deep neural network via active learning with uncertainty sampling along with partitioning. *Adaptive Retrieval* with COSINEBERT [34] was used to greedily maximize the number of positive record pairs. Further, we leveraged geospatial properties

Table 1: Comparison with baselines on fixed CGT test set

Model	Accuracy (%)		R@95P (%)	
	IN	UAE	IN	UAE
Production Model	80.49	84.24	23.78	71.87
Ditto	79.89	82.49	21.65	70.59
Random Sampling	81.05	84.75	23.89	72.11
Confidence Sampling	82.97	84.97	24.92	73.89
ER Rules	83.32	85.49	25.33	75.14
ALMSER	83.81	85.78	25.78	75.66
Deep Transfer AL	84.14	86.39	26.04	76.08
Adaptive Retrieval (COSINEBERT)	84.01	86.91	26.96	75.98
Adaptive Retrieval (GEOBERT)	85.18	86.79	26.97	76.75
DIAL	85.08	87.13	27.68	76.88
Our Approach (Ditto)	85.89	88.77	29.92	79.94
Our Approach	**87.42**	**90.54**	**32.33**	**81.91**
Our Approach (No bridges)	87.09	89.83	31.27	80.19
Our Approach (No partitioning)	85.21	87.95	29.64	79.38
Our Approach (No min-cut)	84.74	86.81	28.07	77.68

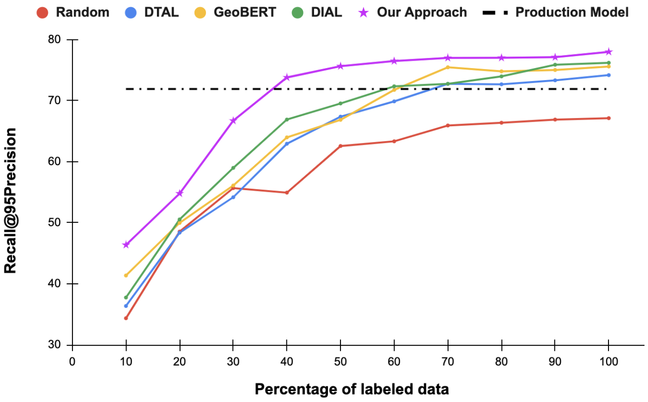


Figure 2: Performance against baselines wrt data size for UAE

and GPS scan information with GEOBERT [24] and utilized adaptive retrieval [34] to generate a stronger baseline. We fine-tuned [25, 26] the pre-trained COSINEBERT [34], and GEOBERT [24] models on CGT train set. *DIAL* [17] used index-by-committee with RoBERTa models. We fine-tuned pre-trained RoBERTa as described in DIAL [17] on CGT train set.

Results and Ablation Study We start with CGT train set as the initial labeled set for all AL baselines. The record pairs retrieved by AL baselines are labeled by the data annotation team. We retrain after augmenting the active learnt pairs to CGT train set and evaluate the performance on the same CGT test set. The Production model and Ditto baselines are trained on CGT train set only. Table 1 show that our approach outperformed all baselines across IN and UAE by a significant margin, thus confirming its wide applicability and generic nature. In comparison to the production model the accuracy improved by 6.62%, and R@95P by 9.3% on average in absolute terms. In comparison to the second best performing approach, an average improvement of 4.84% is observed in R@95P across IN and UAE. The ablation study results in the last three rows of Table 1 highlight the importance of graph partitioning and graph cuts. Also, the improvement with our AL approach across metrics using *Ditto* as classification model (Our Approach (Ditto)) show its effectiveness for deep learning models as well. As a separate experiment, for UAE, we compared top AL strategies to identify the approach that takes the smallest possible subset of CGT train set to reach the R@95P performance observed using 100% of train set (Production Model). We start with identical 10% of the CGT train set, and sampled 10% data in each iteration from the remaining pool using different AL strategies. Figure 2 show that to achieve any R@95P, our approach outperforms all baselines in terms of the percentage of labels requested. Our approach required 28.7% less train data in comparison to the next best method to reach the same performance as production model. Similar trend was observed for IN as well. The details around qualitative analysis of our approach is discussed in Appendix A.2.

5 REAL WORLD APPLICATION

DP Geocoding converts free-form address text to a geocode (pair of latitude-longitude). Having sufficient number of deliveries to an address allows us to learn reasonable quality geocodes by aggregating the past delivery locations [14]. In this work, we limit the scope of learning DP geocodes for cold-start addresses, a particularly challenging task because of lack of historical geocode data. To learn a DP, we match the new address against existing addresses in the database (reference set) for which geocode information is available. We then aggregate the geocodes of matched addresses to learn a single DP geocode using KDE [43]. The key metrics used are — (1) *Delivery Precision* is the percentage of total shipments for which the actual delivery happened within a threshold distance $\mathcal{Z}$ from the planned delivery location. (2) *Delivery Defects* is the percentage of total shipments for which the actual delivery happened outside of the threshold distance $\mathcal{Y}$ from the planned delivery location. Hence, lower the value of outliers, better the metric. For business reasons, we cannot reveal $\mathcal{Z}$ and $\mathcal{Y}$. The details of the offline evaluation are discussed in Appendix A.3.

Online A/B Experiment After observing significant improvements during offline evaluation, we launched an online A/B experiment in the third quarter of 2022 on live traffic for IN geography. We dialled-up the model in a phased manner — 10%, 50%, *and* 100% delivery stations. We observed statistically significant improvements during one week of dial-up in each phase. During the A/B test period, our active learnt model has learned DP geocodes for a few hundred thousand shipments, where we observed *7.84%* improvement in delivery precision and *12.32%* reduction in delivery defects. Following the success in IN, we will launch an online experiment in UAE.

6 CONCLUSION

We propose a novel adaptation of graph-based active learning for customer address resolution. In comparison to existing production system, experiments on manually curated dataset show that our approach is highly effective. After observing significant improvements during offline evaluation for DP geocoding, we successfully deployed our approach and performed online A/B experiments which show that when the production model is augmented with active learnt record pairs, the DP geocoding metrics improved significantly. These improvements lead to better delivery planning, significant decrease in operation costs, and customer satisfaction.

REFERENCES

[1] Chris Biemann, Irina Matveeva, Rada Mihalcea, and Dragomir Radev. 2007. Proceedings of the Second Workshop on TextGraphs: Graph-Based Algorithms for Natural Language Processing. In *Proceedings of the Second Workshop on TextGraphs: Graph-Based Algorithms for Natural Language Processing*.

[2] Mikhail Bilenko and Raymond J Mooney. 2003. Adaptive duplicate detection using learnable string similarity measures. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*. 39–48.

[3] Mustafa Bilgic, Lilyana Mihalkova, and Lise Getoor. 2010. Active learning for networked data. In *ICML*.

[4] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment* 10 (2008), P10008.

[5] Zalán Bodó, Zsolt Minier, and Lehel Csató. 2011. Active learning with clustering. In *Active Learning and Experimental Design workshop In conjunction with AISTATS 2010*. JMLR Workshop and Conference Proceedings, 127–139.

[6] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics* 5 (2017), 135–146.

[7] Aydın Buluç, Henning Meyerhenke, Ilya Safro, Peter Sanders, and Christian Schulz. 2016. Recent advances in graph partitioning. *Algorithm engineering* (2016), 117–158.

[8] Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*. 785–794.

[9] Xiaojun Chen, Joshua Zhexue Haung, Feiping Nie, Renjie Chen, and Qingyao Wu. 2017. A self-balanced min-cut algorithm for image clustering. In *Proceedings of the IEEE International Conference on Computer Vision*. 2061–2069.

[10] Nitin R Chopde and Mangesh Nichat. 2013. Landmark based shortest path detection by using A* and Haversine formula. *International Journal of Innovative Research in Computer and Communication Engineering* 1, 2 (2013), 298–302.

[11] David A Cohn, Zoubin Ghahramani, and Michael I Jordan. 1996. Active learning with statistical models. *Journal of artificial intelligence research* 4 (1996), 129–145.

[12] 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 (2019), 334–348.

[13] Melanie Ducoffe and Frederic Precioso. 2018. Adversarial active learning for deep networks: a margin based approach. *arXiv preprint arXiv:1802.09841* (2018).

[14] George Forman. 2021. Getting Your Package to the Right Place: Supervised Machine Learning for Geolocation. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 403–419.

[15] Clinton Gormley and Zachary Tong. 2015. *Elasticsearch: the definitive guide: a distributed real-time search and analytics engine*. " O'Reilly Media, Inc.".

[16] Govind and Saurabh Sohoney. 2022. Learning Geolocations for Cold-start and Hard-to-Resolve Addresses via Deep Metric Learning. In *Proceedings of the The 2022 Conference on Empirical Methods in Natural Language Processing: Industry Track*. Association for Computational Linguistics, Abu Dhabi, UAE.

[17] Arjit Jain, Sunita Sarawagi, and Prithviraj Sen. 2021. Deep indexed active learning for matching heterogeneous entity representations. *arXiv preprint arXiv:2104.03986* (2021).

[18] Zhanming Jie and Wei Lu. 2019. Dependency-guided LSTM-CRF for named entity recognition. *arXiv preprint arXiv:1909.10148* (2019).

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

[20] Ambika Kaul, Saket Maheshwary, and Vikram Pudi. 2017. Autolearn—automated feature generation and selection. In *2017 IEEE International Conference on data mining (ICDM)*. IEEE, 217–226.

[21] Asif R Khan and Hector Garcia-Molina. 2016. Attribute-based crowd entity resolution. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. 549–558.

[22] David D Lewis and Jason Catlett. 1994. Heterogeneous uncertainty sampling for supervised learning. In *Machine learning proceedings 1994*. Elsevier, 148–156.

[23] 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* (2020).

[24] 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*. 2209–2214.

[25] Rishabh Maheshwary, Saket Maheshwary, and Vikram Pudi. 2021. A context aware approach for generating natural language attacks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 15839–15840.

[26] Rishabh Maheshwary, Saket Maheshwary, and Vikram Pudi. 2021. Generating natural language attacks in a hard label black box setting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 13525–13533.

[27] Saket Maheshwary, Ambika Kaul, and Vikram Pudi. 2017. Data driven feature learning.

[28] Saket Maheshwary and Hemant Misra. 2018. Matching resumes to jobs via deep siamese network. In *Companion Proceedings of the The Web Conference 2018*. 87–88.

[29] Saket Maheshwary and Vikram Pudi. 2017. Mining keystroke timing pattern for user authentication. In *New Frontiers in Mining Complex Patterns: 5th International Workshop, NFMCP 2016, Held in Conjunction with ECML-PKDD 2016, Riva del Garda, Italy, September 19, 2016, Revised Selected Papers 5*. Springer, 213–227.

[30] Katerina Margatina, Giorgos Vernikos, Loïc Barrault, and Nikolaos Aletras. 2021. Active learning by acquiring contrastive examples. *arXiv preprint arXiv:2109.03764* (2021).

[31] Venkata Vamsikrishna Meduri, Lucian Popa, Prithviraj Sen, and Mohamed Sarwat. 2020. A comprehensive benchmark framework for active learning methods in entity matching. In *Proceedings of the 2020 ACM SIGMOD Conference on Management of Data*. 1133–1147.

[32] Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*. 1003–1011.

[33] 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*. 19–34.

[34] Stephen Mussmann, Robin Jia, and Percy Liang. 2020. On the importance of adaptive data collection for extremely imbalanced pairwise tasks. *arXiv preprint arXiv:2010.05103* (2020).

[35] Tan Ningsheng, Yang Chongjun, Yang LiuZhong, and Liu Yuan. 2015. An address regional tessellation method for spatial subdivision and geocoding in digital earth system. *International Journal of Digital Earth* 8, 10 (2015), 825–839.

[36] Vamsi Krishna Penumadu, Nitesh Methani, and Saurabh Sohoney. 2022. Learning geospatially aware place embeddings via weak-supervision. In *Proceedings of the 30th International Conference on Advances in Geographic Information Systems*. 1–10.

[37] Anna Primpeli and Christian Bizer. 2021. Graph-boosted active learning for multi-source entity resolution. In *International Semantic Web Conference*. Springer, 182–199.

[38] Kun Qian, Lucian Popa, and Prithviraj Sen. 2017. Active learning for large-scale entity resolution. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. 1379–1388.

[39] Kun Qian, Lucian Popa, and Prithviraj Sen. 2017. Active learning for large-scale entity resolution. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. 1379–1388.

[40] Kun Qian, Lucian Popa, and Prithviraj Sen. 2019. Systemer: A human-in-the-loop system for explainable entity resolution. (2019).

[41] Alexander Ratner, Sen Bach, Stephen H, and Christopher Ré. 2017. Snorkel: Rapid training data creation with weak supervision. In *Proceedings of the International Conference on Very Large Data Bases*, Vol. 11. NIH Public Access, 269.

[42] Dongyu Ru, Jiangtao Feng, Lin Qiu, Hao Zhou, Mingxuan Wang, Weinan Zhang, Yong Yu, and Lei Li. 2020. Active sentence learning by adversarial uncertainty sampling in discrete space. *arXiv preprint arXiv:2004.08046* (2020).

[43] David W Scott. 1992. Multivariate density estimation: Theory, practice and visualisation. John Willey and Sons. *Inc., New York* (1992).

[44] Isabel Segura-Bedmar, Adrián Carruana, and Paloma Martínez. 2016. LABDA at the 2016 BioASQ challenge task 4a: Semantic Indexing by using ElasticSearch. In *Proceedings of the Fourth BioASQ workshop*. 16–22.

[45] Burr Settles. 2009. Active learning literature survey. (2009).

[46] Dan Shen, Jie Zhang, Jian Su, Guodong Zhou, and Chew Lim Tan. 2004. Multi-criteria-based active learning for named entity recognition. In *Proceedings of the 42nd annual meeting of the Association for Computational Linguistics (ACL-04)*. 589–596.

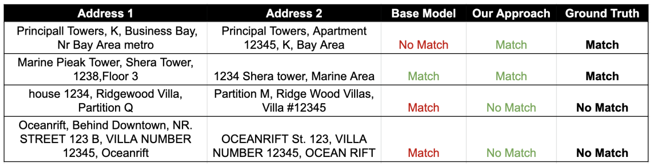
A APPENDIX

A.1 Background Details

This section introduces necessary background on concepts from graph theory and geospatial properties of customer addresses. An undirected and weighted graph G with no self-loops is a pair (V, E), where V is a set of vertices or nodes and E is a set of edges between the nodes. Each distinct text from customer address database is represented via a node and the edge between two nodes is determined based on a match or no-match prediction of a trained machine learning model. We use $n = |V|$ to denote the number of nodes and $m = |E|$ to denote the number of edges.

A.1.1 Graph Partitioning. It [7] refers to a class of problems that deals with reducing a graph to multiple smaller graphs by partitioning its set of nodes into mutually exclusive groups. The nodes that are much more linked to nodes within the groups compared to nodes in the other groups are said to form communities. We use the Louvain algorithm [4], a graphical method to partition a graph based on network structure and edge relationships. Louvain is an unsupervised algorithm and consists of two important phases, *modularity optimization* and *community aggregation* [4]. These steps are executed until there are no more changes in the network and *maximum modularity* of the graph is achieved. In comparison to other graph partitioning techniques, Louvain was preferred as it does not require us to input the number of communities, or the partition sizes before execution. Further, a single pass of Louvain is a *linear operation* in terms of the number of edges of the graph, thus allowing it to scale across millions of edges in a graph.

A.1.2 Graph Cuts. These techniques have been successfully applied to a number of real-world applications, for example, designing flow networks, computer vision, graphics and image processing problems [9]. We leverage *minimum cut* and *bridge* as graph cut techniques for the task of EM across customer addresses. For a pair of given nodes, source s and target t, the $s - t$ min-cut of a weighted graph is defined as the minimum sum of weights of (at least one) edges that when removed from the graph makes s and t disconnected. Bridge in an undirected graph G is an edge that when removed from a graph increases the number of connected components. In other words, if we remove an edge which is a bridges, the graph will no longer remain connected.



Address 1	Address 2	Base Model	Our Approach	Ground Truth
Principall Towers, K, Business Bay, Nr Bay Area metro	Principal Towers, Apartment 12345, K, Bay Area	No Match	Match	**Match**
Marine Pieak Tower, Shera Tower, 1238,Floor 3	1234 Shera tower, Marine Area	Match	Match	**Match**
house 1234, Ridgewood Villa, Partition Q	Partition M, Ridge Wood Villas, Villa #12345	Match	No Match	**No Match**
Oceanrift, Behind Downtown, NR. STREET 123 B, VILLA NUMBER 12345, Oceanrift	OCEANRIFT St. 123, VILLA NUMBER 12345, OCEAN RIFT	Match	No Match	**No Match**

Figure 3: Demonstrates the quality of predictions of the base (Production) model against our approach

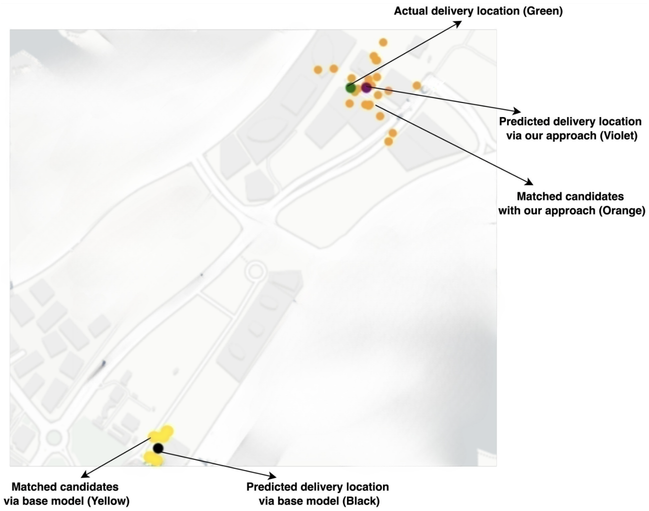

Actual delivery location (Green)
Predicted delivery location via our approach (Violet)
Matched candidates with our approach (Orange)
Matched candidates via base model (Yellow)
Predicted delivery location via base model (Black)

Figure 4: Demonstrates the comparison of quality of actual and predicted delivery locations

A.1.3 Geospatial Properties. For each node V in the graph G, two main modalities of information are available namely: the free-form address text and multiple GPS points based on successful past deliveries. GPS points are sometimes noisy as it depends on driver compliance. We use the GPS points associated with each address to learn a single delivery point (DP) to direct future deliveries. A brute force approach would be to compute the centroid of GPS points from past deliveries. Unfortunately, this can direct delivery associates to the middle of the street or to a different building. Centroids and medoids are prone to outliers, hence proving inaccurate in estimating delivery points [14]. We use density-based methods to accurately approximate a single delivery point from historical deliveries for each address via Kernel Density Estimation (KDE) [43]. KDE maximized delivery points are further used to determine the geospatial proximity between a pair of graph nodes using haversine distance [10]. The haversine distance, also called as great circle distance, is the shortest distance between two points on the surface of a sphere (Earth) with each point denoted by its latitude and longitude.

A.1.4 Geospatial Proximity. We use Gaussian based Kernel Density Estimation (KDE) [43] with 25 meters bandwidth to approximate a single delivery point from historical deliveries for each address. KDE maximized delivery points are then used to determine the geospatial proximity between a pair of graph nodes using haversine distance [10].

A.2 Qualitative Analysis

We observe the quality of predictions generated by the model with and without our proposed approach. To observe the quality of pairwise matching, we analysed address pairs from some neighbourhoods. We replace the personally identifiable information with dummy values (e.g 1234, 12345, 1238) and retain the public components of addresses to not reveal actual customers. The outcomes of the base model and our approach are shown in Figure 3. It is evident from these examples that our approach handles false positives and false negatives effectively.

Further, we analysed the DP geocodes predicted by the base model and our approach against the actual delivery location. The quality of predictions is highlighted through the following real-world scenario. *Downtown, Diablo Dum Mall St., 1234, Downtown*

Sky View is a newly created address. Figure 4 shows that the base (Production) model incorrectly match the new address against multiple addresses from the adjoining streets (yellow points), hence learning an inaccurate DP (black point), resulting in a delivery defect when compared to the actual delivery location (green point). With our approach, the model accurately identified addresses from the same building (orange points) to learn an accurate DP (violet point) within the threshold $\mathcal{Z}$ of the actual delivery location.

A.3 Offline Evaluation

The deliveries that happened from April 2020 to April 2022 across all delivery stations were considered for creating the reference set. Deliveries across the span of first three weeks of May 2022 were used as test set. During the test period, a few hundred thousand deliveries were done on cold-start addresses against which we evaluated our approach. We re-trained the production model by augmenting the initial CGT train set with record pairs sampled via our approach and observed the impact on DP geocoding. On an average across IN and UAE, we observed 6.74% improvement in delivery precision and 11.39% reduction in delivery defects compared to the system in production.

A.4 Classification Model

Following Comber et al. [12], we first parse the address text using BiLSTM-CRF [18] into address fields (unit, building, road, locality, etc.). Further we engineer features, such as cosine similarity and fuzzy match score of record pairs for all the parsed address fields to perform pair-wise matching (binary classification) using the XGBoost [8] classifier. This classifier $\mathcal{M}$, serves real-time traffic in our production system and was trained on manually curated ground truth data.