FastAddr: Real-time Abnormal Address Detection via Contrastive Augmentation for Location-based Services

Zhiqing Hong
Rutgers University, Piscataway, USA
JD Logistics, Beijing, China
zhiqing.hong@rutgers.edu

Heng Yang
Beijing Institute of Computer Technology and Application
Beijing, China
008yheng@163.com

Haotian Wang
JD Logistics
Beijing, China
wanghaotian18@jd.com

Wenjun Lyu
Rutgers University
Piscataway, USA
wenjun.lyu@rutgers.edu

Yu Yang
Lehigh University
Bethlehem, USA
yuyang@lehigh.edu

Guang Wang
Florida State University
Tallahassee, USA
guang@cs.fsu.edu

Yunhuai Liu
Peking University
Beijing, China
yunhuai.liu@pku.edu.cn

Yang Wang
University of Science and Technology of China
Suzhou, China
angyan@ustc.edu.cn

Desheng Zhang
Rutgers University
Piscataway, USA
desheng@cs.rutgers.edu

ABSTRACT

An address, a textual description of a physical location, plays an important role in location-based services such as on-demand delivery and e-commerce. However, abnormal addresses (i.e., an address without detailed information representing a spatial location) have led to significant costs. In real-world settings like e-commerce, abnormal address detection is not trivial because it needs to be completed in real-time to support massive online queries. In this study, we design `FastAddr`, a fast abnormal address detection framework, which detects abnormal addresses among millions of addresses in a short time. By investigating and modeling the hierarchical structure of address data, we first design a novel contrastive address augmentation approach to generate training data via learning the entity transition probability matrix. We further design a lightweight multi-head attention model for learning compact address representation by modeling the address characteristics. We conduct a comprehensive three-phase evaluation. (i) We evaluate `FastAddr` on a real-world dataset and it yields the average F1 of 85.7% in 0.058 milliseconds, which outperforms the state-of-the-art models by 47.4% with similar detection time. (ii) An offline A/B test shows that `FastAddr` outperforms the previous deployed model significantly. (iii) We also conduct an online A/B test to compare `FastAddr` with the deployed model, which shows an improvement of F1 by more than 20%. Moreover, a real-world case study demonstrates both the efficiency and effectiveness of `FastAddr`.

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

ACM ISBN 978-1-4503-9529-8/22/11…$15.00
https://doi.org/10.1145/3557915.3560999

CCS CONCEPTS

• **Information systems** → **Location based services**; • **Applied computing** → **E-commerce infrastructure**.

KEYWORDS

real-time abnormal address detection, contrastive augmentation, multi-head attention, geocoding

ACM Reference Format:
Zhiqing Hong, Heng Yang, Haotian Wang, Wenjun Lyu, Yu Yang, Guang Wang, Yunhuai Liu, Yang Wang, and Desheng Zhang. 2022. FastAddr: Real-time Abnormal Address Detection via Contrastive Augmentation for Location-based Services. 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/10.1145/3557915.3560999

1 INTRODUCTION

Location-based services (LBS) provide convenient services such as on-demand delivery and map navigation for users according to spatial locations. Generally, users input a textual description of a location, i.e., address, to obtain the location-based functions. The service providers use a Geocoding system to convert the input address into geographic coordinates for later processes such as range queries and shortest distance calculations. Returning an accurate location within a short time is the ultimate goal of the Geocoding system. To achieve this goal, the Geocoding system typically consists of two steps, as shown in Fig. 1(a). *Address Matching:* The service providers usually maintain a large address database, e.g., with more than 100 million textual addresses and their corresponding coordinates. The input address will be first matched with the database to fetch the related information. *Address Recommendation:* if the input address is not found in the database, some other methods, such as approximate matching or similarity searching, will recommend the possible coordinate for the query. When the user

selects the correct coordinate based on a visualization platform, this coordinate and the textual address are stored in the database.

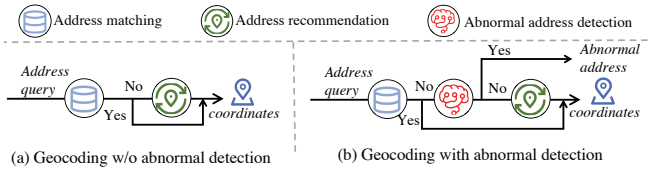


Figure 1: Geocoding system workflow.

In the real world, especially in fast-developing countries such as India and China, many low-quality and abnormal addresses [1] exist mainly due to weak address regulation, complex address structure [44], and the fraud of click farming [49] (see Sec. 2.1). For example, the address *"Anhui Province, Hefei City, deliver the parcel by the door"* is an abnormal address because no detailed street information is provided. These abnormal addresses are not maintained in the database, so they need to be checked in the recommendation process, wasting a lot of time to return an unconfident coordinate, which is not meaningful. To handle millions of address parsing requests each day and reduce latency, abnormal address detection should finish in a short time (e.g., less than one millisecond). Most existing Geocoding systems solve this problem implicitly [2], i.e., returning an error message if the confidence of Geocoding an address is lower than a threshold. These systems have two major limitations. (i) Efficiency: address recommendation based on machine learning algorithms is time-consuming to rank and recommend a possible coordinate from a large address database. (ii) Effectiveness: these systems emphasize the precision of address queries rather than the recall of abnormal addresses, making them less effective when the abnormal and low-quality input addresses are pervasive. Therefore, to improve the efficiency and effectiveness of the existing Geocoding systems, we aim to design a framework that automatically detects abnormal addresses before address recommendations, as shown in Fig. 1(b).

Admittedly, some works have studied problems related to abnormal address detection. Instead of directly addressing the abnormal address issue, some studies focus on applications related to address and Point-of-Interest (POI), such as coordinates generating for POIs and addresses [39, 41], and POI alias discovery [16]. Meanwhile, there are many anomaly detection works in different modalities, e.g., image [15], time series [4, 10], graph data [52], and text data [26, 32], which have applications such as pickpocket suspects detection [14] and malicious behavior detection on social media [34, 50]. However, in the abnormal address domain, existing studies are faced with three major challenges (see Sec. 2.2.3), including (i) the lack of large-scale labeled address data for model training, which makes it challenging to train expressive machine learning models; (ii) the complex address structure resulted from user habits and weak regulation in developing countries, which is significantly different from the well-regulated addresses in the U.S.; (iii) the requirement of real-time detection in real-world location-based services that receive millions of Geocoding requests every day, which puts a significant challenge for the detection algorithm to respond in a short time.

Most existing works that can be applied to abnormal address detection are categorized into four categories as in Fig. 2. It is challenging to have both high precision and short detection time because of

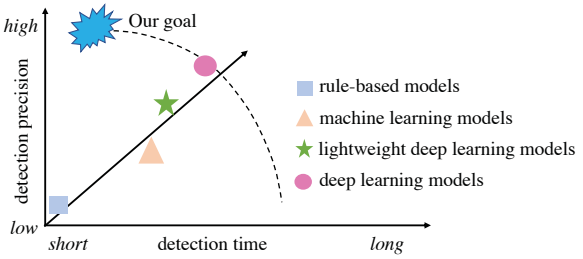


Figure 2: Design goal illustration.

the trade-off between model capability and model simplicity. However, we argue that we might achieve both high precision and short detection time by modeling the hierarchical structure of address data and focusing on important positions in addresses, which is the design goal of this study as in Fig. 2.

To address the aforementioned challenges and realize the design goal, we design `FastAddr`, a Fast abnormal Address detection framework, which consists of two main modules: (i) a contrastive address augmentation module that generates large-scale labeled addresses to solve the data scarcity problem. The augmentation module learns a spatial entity transition matrix to generate addresses that align with the real-world address distribution, improving the model's generalizability even with unseen addresses. (ii) a lightweight multi-head attention model that captures the important segments of address sentences for abnormal address detection, which is motivated by the characteristics of addresses. The lightweight model significantly reduces the computational cost by squeezing the parameter space for attention calculation.

In summary, the major contributions are listed as follows:

- To the best of our knowledge, `FastAddr` is the first attempt to formulate and explicitly solve the abnormal address detection problem. We motivate `FastAddr` based on extensive data-driven analyses in a real-world e-commerce logistics platform. We show (i) the significance of the abnormal address problem in the platform and (ii) the challenges of addressing the abnormal address detection problem due to the lack of labeled data, address format complexity, and real-time detection requirements.
- We design a lightweight and effective framework `FastAddr` to detect abnormal addresses in real time. Based on an in-depth investigation of address data structure, `FastAddr` addresses the challenge of data scarcity by a novel contrastive address augmentation approach, generating labeled data for model training. In addition, `FastAddr` learns compact address representation by an efficient multi-head attention mechanism that captures important address segments with significantly fewer parameters than most state-of-the-art models.
- More importantly, we conduct a three-phase evaluation based on real-world datasets from an e-commerce logistics platform. The experimental results show that `FastAddr` achieves 99.3% of AUC, 92% of precision, 89.3% of F0.5-score, 85.7% of F1-score, and 0.058 millisecond of the detection time. With similar detection time, `FastAddr` outperforms the state-of-the-art models by 47.4%. Moreover, compared with a state-of-the-practice model deployed on the platform, `FastAddr` improves the detection precision by 21.5% and the detection rate by 25%.

2 BACKGROUND AND MOTIVATION

In this section, we introduce the preliminaries and motivation about the abnormal address problem, followed by three key challenges.

2.1 Preliminaries

Addresses are widely used in location-based services such as logistics and ridesharing to describe spatial locations. In this study, we define the address with low quality as *Abnormal Address* as in Def. 1. Abnormal addresses generally miss detailed road or building information, which is essential to locating the address by either Geocoding systems or humans, e.g., delivery couriers. Examples of both normal and abnormal addresses are given in Table 1.

Definition 1. *(**Abnormal Address**) is an address without detailed address information, e.g., road, building, and POI name, and thus can not be localized by humans (e.g., couriers) for certain purposes (e.g., delivery).*

Table 1: Examples of abnormal and normal addresses.

Index	Address Example	Reason Analysis
① Abnormal Address	Anhui Province Hefei City	No detailed location information, e.g., *road name*, *POI name*
② Abnormal Address	Anhui Province Hefei City deliver by the door	Oral language, no reachable location information, e.g., *POI name*
③ Normal Address	Anhui Province Hefei City Shangri-La Hotel	With detailed POI information, i.e., *Shangri-La Hotel*

The origin of abnormal addresses. There are two main scenarios resulting in abnormal addresses. (1) *Unintentional abnormal address.* Unlike addresses with clear and consistent structures in developed countries such as the U.S. and Japan, in developing counties such as India and China, addresses entered by users are complex and of diverse qualities. For example, low-quality addresses cannot be geocoded by commercial Geocoding services, e.g., Google Maps and Baidu Maps [35, 39, 41, 44]. Moreover, due to rapid development, new addresses and spatial entities are emerging every day. Therefore, it is common for customers to enter addresses with typing errors or ignored detailed addresses. (2) *Intentional abnormal address.* People may intentionally use abnormal addresses to make extra money or save costs. For example, abnormal addresses are used for the fraud of click farming [49]. Some people create abnormal addresses to place fake orders on e-commerce platforms to increase sales ranking; some may cheat for compensation because their packages cannot be delivered in time (to abnormal addresses).

2.2 Data and data-driven investigation

2.2.1 Data description. This study utilizes three datasets.

(i) Address data includes 20 million real-world individual addresses, business addresses (such as companies), schools, etc., which are accessed based on our collaboration with a large e-commerce logistics platform in China. The data is anonymized without personal identities such as names and phones.

(ii) Spatial entity data includes POI, road names, and all administrative partitions, including provinces, cities, districts, and towns of China. POI data includes multiple types (e.g., school and restaurant). Road data includes urban and country road names in different areas and scales, e.g., province-level roads, city-level roads, and town-level roads. In total, spatial entity data consists of more than 30 million POI and road names, 600 cities, and 30 provinces. In general, an address consists of multiple spatial entities.

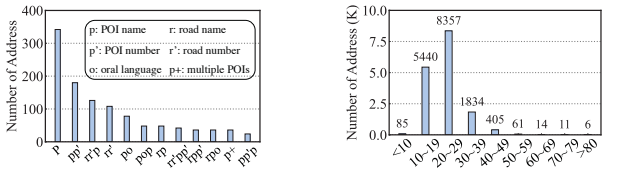
(iii) Natural language data includes high-frequency phrases that describe spatial orientation, parcel-related notes, articles, news, and novels. It is collected from customers' reviews on the e-commerce platform and open-source websites such as Wikipedia. We apply an Organization and Location Named Entity Recognition (OL-NER) model [11, 20] to extract and remove all *organization names* and *location names*, which are used to construct abnormal addresses.

2.2.2 Problem significance. We investigate the impact of abnormal addresses on the real-world platform through an in-depth data-driven analysis. Based on the data from the e-commerce and credit card platforms, we identify 1.5% and 3% abnormal addresses among randomly sampled 100,000 addresses on these two platforms, respectively. These abnormal addresses lead to order delivery failure for the e-commerce platform and high-risk malicious overdue for the credit card platform. Given millions of addresses geocoded each day, there are a significant number of abnormal addresses even though the ratio of abnormal addresses is low. For example, abnormal addresses cost 3 million U.S. dollars in delivery waste of the large e-commerce logistics platform in 2021.

2.2.3 Challenges. Three main challenges make large-scale abnormal address detection a non-trivial task.

(1) Lack of large-scale labeled data. Even though we can easily collect normal addresses from real-world location-based services, it is challenging to collect abnormal addresses due to two main reasons. Firstly, real-world abnormal addresses can only be identified by delivery workers or customer services, who rarely have the incentive to report or provide feedback. Secondly, we cannot collect all kinds of abnormal addresses due to the evolving locations in developing countries, making deep learning models challenging to generalize to unseen abnormal addresses well.

(2) Complexity of Chinese addresses. In developed countries such as the U.S., the urban structures are relatively stable, and addresses have a well-organized format (e.g., road number + road name). Therefore, abnormal addresses can be detected by straightforward methods, e.g., applying regular expression with a nationwide address table to detect whether an address contains required information such as road number and name. However, in some developing countries such as China and India, many new addresses appear each year due to economic development [12]. Moreover, these addresses have diverse formats, ambiguous address content, etc., due to the lack of a standard address format [39, 44]. Fig. 3(a) illustrates the ratio of different address component combinations, e.g., *rpo* represents *"road name + POI + oral language"*, which demonstrates the complexity of the address format. The lengths of address sentences also vary significantly, as shown in Fig. 3(b). Therefore, the Chinese address is usually the partial combination of entities such as province, city, road, road number, POI name, and natural language (e.g., "left to the corner") with diverse orders and lengths, which makes it nontrivial to detect abnormal addresses.



(a) Address components. (b) Address length distribution.

Figure 3: Address characteristic analysis.

(3) Strict detection time requirement. The real-world, large-scale application requirement poses another challenge, i.e., the detection time requirement. For example, an e-commerce company with 15 million daily orders requires the abnormal detection model to respond to each request within a strict time window. Therefore, the strict detection time requirement is another important factor that we need to consider in the model design.

In summary, based on our data-driven analysis, we found (i) the abnormal address problem is important and has significant real-world impact, and (ii) it is nontrivial to design an effective and efficient model due to the lack of labeled data, address complexity, and high-speed requirements in real-world applications.

3 ABNORMAL DETECTION FRAMEWORK

In this section, we introduce the detailed design of FastAddras shown in Fig. 4, which consists of two main components. (i) The contrastive address augmentation module for address data augmentation and thereby enabling the training of abnormal address detection model. The spatial entity data (including *city, road, POI,* etc.) and natural language data (including *novel, news,* etc.) are sampled to construct synthetic addresses by learning the spatial entity transition matrix from the address data. Then, the constructed normal and abnormal addresses are fed into (ii) the lightweight multi-head attention model for learning compact address representations, which is utilized to detect abnormal addresses.

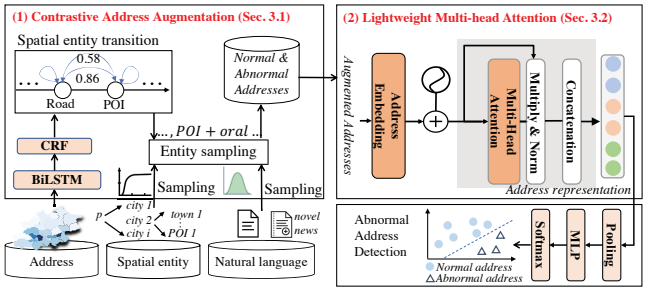
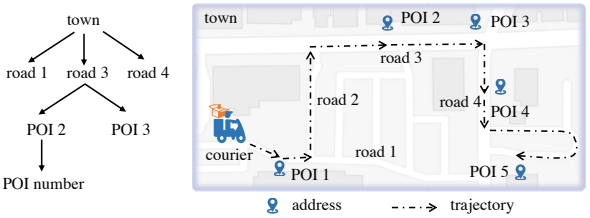


Figure 4: Overall framework.

3.1 Contrastive address augmentation

In this subsection, we introduce contrastive address augmentation. One of the critical challenges in the abnormal address detection task is the scarcity of large-scale labeled data to train neural network models. Different from general classification tasks where data in all categories are easy to obtain and can be labeled manually, few abnormal addresses are available in abnormal address detection, which prevents us from training a model with large-scale labeled data. However, a good opportunity is that we can access large amounts of normal addresses. Therefore, a research question is *can we augment existing addresses by designing a novel augmentation approach?* To answer this question, we design a contrastive address augmentation approach by learning a spatial entity transition matrix that augments existing addresses for model training.

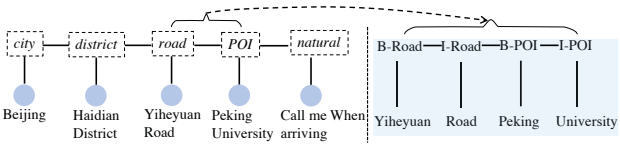
3.1.1 Contrastive spatial sample augmentation. Existing data augmentation approaches such as cropping-based augmentation, span cutoff, and mix up [33, 42, 48], significantly boost the performance of many tasks such as text classification. However, we argue that we cannot directly apply these existing approaches to our problem due to a major challenge, i.e., address data has an apparent hierarchical structure as shown in Fig. 5(a). For example, an address might consist of *province, city, district, town, road, POI, and POI number*, which exhibits hierarchy in terms of spatial granularity. Fig. 5(b) illustrates a concrete example of a four-level hierarchy in a real-world logistics delivery scenario. Therefore, directly applying existing augmentation approaches cannot retain the hierarchical address structure and may not result in satisfactory performance. Motivated by the characteristics of address data and the success of contrastive learning models [8], we design a novel augmentation approach. This approach constructs contrastive *<normal, abnormal>* address pairs, which enforce the downstream deep learning models to learn the difference between normal and abnormal addresses.



(a) Spatial entity relationship (b) Logistics delivery in four-level entities

Figure 5: Illustration of spatial entity hierarchy.

3.1.2 Spatial entity transition matrix. A key challenge in augmenting address data is maintaining the hierarchical structure and the transition patterns among spatial entities within an address. Each address consists of a sequence of entities (e.g., *Peking University* and *Yiheyuan Road*), which represent the hidden labels (e.g., entity class *POI* and *road*). Moreover, the entity labels within one address correlate with each other, violating the Markov assumption. Therefore, we model the address by Conditional Random Field (CRF), where all spatial entities constitute observation sequences in CRF and labels for these entities constitute state sequences as shown in Fig. 6. Each entity might have multiple states.



(a) Spatial entity transition example (b) State transition example

Figure 6: Entity transition and state transition in addresses.

State transition and entity transition. To learn the transition patterns of spatial entities, we design and learn the entity transition matrix of addresses. We design nine entities to represent an

address, i.e., *province, city, district, town, road, POI, road number, POI number, natural language*. For each entity, we have two types of labels *B* and *I*. For example, as in Fig. 6(b), the word *Yiheyuan* is labeled with *B-Road*, representing it is the beginning of a *Road* entity, while the following word *Road* is labeled as *I-Road* to represent it is not the beginning of a a *Road* entity (intermediate). Thus, we have 18 types of labels in total. Intuitively, we can calculate the *state transition matrix* by identifying the labels for each word in addresses, representing the semantic meaning in each entity. However, we do not need fine-grained state transition probability to construct addresses because we already have structured spatial entity data such as *road* and *POI*. Moreover, due to labeling errors or model capacity issues, the error rate of identifying the correct state sequence is intuitively higher than identifying the correct spatial entity sequence because the error of entity boundary will not impact the entity sequence. Therefore, as shown in Fig. 7, we aim to learn *spatial entity transition matrix* from real address data.

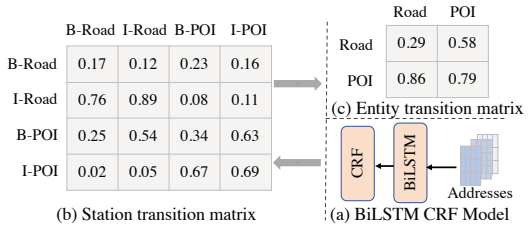


Figure 7: State transition and spatial entity transition matrix.

Fig. 7(a) shows the process of calculating the *state transition matrix* of a real-world address dataset. Specifically, we implement a BiLSTM-CRF Model [18, 20] to extract spatial entities (e.g., *city, road, and POI*) in the address dataset. The BiLSTM-CRF Model consists of three LSTM layers and a CRF layer, which is trained by a labeled dataset. Thus, the *state transition matrix* is calculated based on extracted spatial entities, as shown in the example of Fig. 7(b). We then derive the *spatial entity transition matrix* from the state transition matrix by adding transition probabilities belonging to the same entity. As an example in Fig. 7(c), the transition probability between entity *Road* to *POI* equals the sum of the transition probability between *B-Road, I-Road* and *B-POI, I-POI*.

3.1.3 Contrastive address sampling. Based on the *spatial entity transition matrix*, we then design a sampling algorithm to construct contrastive address samples, i.e., positive (abnormal) addresses, and negative (normal) addresses. We start by constructing a label sequence for an address via sampling from the *spatial entity transition matrix*, e.g., *<province, city, district, town, road, POI, natural language>*. We then construct negative samples following the label sequence. Note that we still need an augmentation even though we have enough negative addresses because we aim to introduce randomness in the negative addresses to increase model robustness in the normal addresses. Seven entities are sampled, i.e., province, city, district, town, road, POI, and natural language data. Except for natural language data, all other entities can be sampled uniformly from the spatial entity database. For the natural language data, we sample it from the natural language database as follows. We first sample the document index uniformly from $idx \sim \mathrm{U}[1,\mathrm{N}]$, where N is the total number of documents. We define the length of natural language as ℓ_{min}=1 and ℓ_{max}=32, the length of natural language entity (*natural*) is $\ell_{natural}$:

$$\ell_{natural} \sim \{\ell_{min}, \ell_{min}+1, ..., \ell_{max}\} \tag{1}$$

The start position o_i^{start} and end position of the natural language entity are calculated as below:

$$o_i^{start} \sim \{1, ..., L^{idx} - \ell_{natural}\}, o_i^{end} = o_i^{start} + \ell_{natural} \tag{2}$$

where L^{idx} is the length of the *idx-th* document. Given the document set d, the sampled natural language entity is:

$$o_i = d^{idx}_{o_i^{start}:o_i^{end}} \tag{3}$$

After we have a negative (normal) address, we construct the contrastive positive (abnormal) address by removing the detailed address entities, i.e., *road* and *POI*. In total, we construct 8 million normal and abnormal addresses for the downstream models.

3.2 Lightweight Multi-head Attention Model

Based on the labeled addresses from the contrastive address augmentation, we train our model to learn effective and efficient address representations. In this subsection, we introduce our lightweight multi-head attention model, as shown in Fig. 8, which is driven by two main design goals: (i) short detection time for real-time detection, and (ii) high detection performance.

Intuition. Intuitively, entities in an address are not equally important for abnormal address detection. For example, when an address consists of a POI and a piece of natural language, focusing on the POI will probably yield better abnormal detection performance because containing valid POIs represents the address is normal. Therefore, we design a multi-head attention mechanism to assign higher importance to important entities of an address.

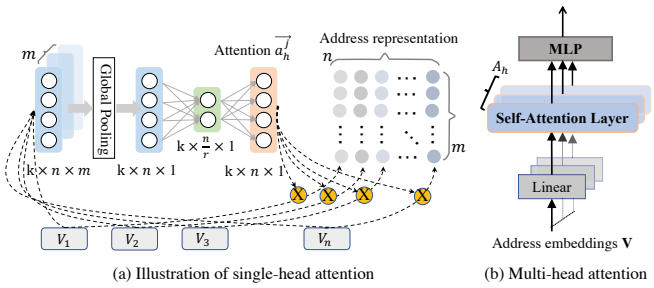


Figure 8: Lightweight attention model.

Address Representation Module. Even though the main characteristics (i.e., the hierarchical structure) of the address data are different from the open-domain data, there are still similar patterns, e.g., there are natural languages in the Chinese textual address data. Therefore, we train the model based on the pre-trained Chinese word embedding [40]. The embedding of *j-th* address is described in Eqn. (4).

$$\mathbf{V}^j = [\mathbf{V}_1^j, ..., \mathbf{V}_i^j, ..., \mathbf{V}_n^j] \tag{4}$$

where $\mathbf{V}_i^j$ is the embedding for *i-th* word in an address sentence and $j \in [1, k]$, k is the number of addresses in a batch, n is the maximum length of all addresses. The address embedding $\mathbf{V}$ is a

three-dimensional matrix, $\mathbf{V} \in \mathbb{R}^{k \times n \times m}$, where m is the embedding dimension for a word embedding vector $\mathbf{V}_i^j$.

The address embeddings are then utilized as input for address representation learning. Intuitively, due to the spatial entity hierarchy of addresses, the importance of different address entities varies significantly, i.e., only some words play an important role in determining whether an address is normal or abnormal. For example, for address *"Haitian District, Peking University, better deliver before 11:00 AM", "Peking University"* is an important entity, which implies this address is a normal address with high confidence.

A straightforward approach is to define an importance vector and assign this vector to addresses, i.e., components at different locations have pre-defined fixed importance. However, due to the address structure and length variance, fixed importance vectors will not bring a satisfactory performance. Another approach is to design a self-attention mechanism with Q, K, and V matrices such as in Transformer [45]. This approach does not fit our design requirement well due to the high computational cost (see Table 2).

Inspired by the attention mechanism in computer vision community [19], which assigns importance to different channels in images, we view each word in an address as a *channel* and assign importance to each word by learning the positional importance among all entities. Fig. 8(a) describes the process of calculating address representation based on a single-head attention mechanism.

Specifically, the dimension of each word embedding in $\mathbf{V}^j$ is reduced from m to 1 by a Global Pooling operation, which compresses the embedding information and reduces the computation overhead significantly, i.e., the embedding dimension reduced from $k \times n \times m$ to $k \times n \times 1$.

$$\mathbf{T}_{n\times 1}^j = \texttt{Pooling}(\mathbf{V}_{n\times m}^j) \tag{5}$$

Then, the compressed embedding $\mathbf{T}_{n\times 1}^j$ is followed by a `Linear` layer with learnable parameters W_1. The `Linear` layer encourages the information flows between different words in an address by compressing the number of words from n to $\frac{n}{r}$, where r is the compression ratio and we set it as 4. The output of The `Linear` layer is $\mathbf{L}_{\frac{n}{r}\times 1}^j$.

$$\mathbf{L}_{\frac{n}{r}\times 1}^j = W_1 \mathbf{T}_{n\times 1}^j \tag{6}$$

After a `ReLU` activation, the dimension of channels is increased back to n by another `Linear` layer with parameters W_2. The embedding with dimension $n \times 1$ is transformed to an attention score for the *j-th* address by a `Sigmoid` layer.

$$a^j = \left(1 + exp(-W_2 \texttt{ReLU}(\mathbf{L}_{\frac{n}{r}\times 1}^j))\right)^{-1} \tag{7}$$

Multi-head attention. To capture importance at different granularities, we further extend our model to a multi-head attention model, as shown in Fig. 8(b), which increases the expressiveness of learned address representations. We use $\vec{a_i^j}$ to represent the attention of the *i-th* attention head. The multi-head attention $\mathbf{A}^j$ is calculated as follows in Eqn. (8).

$$\mathbf{A}^j = [\vec{a_1^j}, ..., \vec{a_i^j}, ..., \vec{a_h^j}] \tag{8}$$

where h is the number of attention head in the model, and $\mathbf{A^j} \in \mathbb{R}^{h \times n}$.

After the multi-head attention matrix $\mathbf{A^j}$ has been calculated for all addresses through the forward process, we derive the weighted address embedding $\mathbf{H}$ through multiplication and concatenation as in Eqn. (9).

$$\mathbf{H}^j = F_{cat}(\mathbf{A}^j, \mathbf{V}^j) = [\vec{a_1^j} \times \mathbf{V}^j || , ..., || \vec{a_h^j} \times \mathbf{V}^j] \tag{9}$$

where $\times$ is the multiplication between vectors and matrices, $||$ represents the vector concatenation operation. The obtained weighted address embedding has a larger embedding space, i.e., $\mathbf{H} \in \mathbb{R}^{k \times n \times m \times h}$.

Detection Module. The address representation from the multi-head attention module is fed into a `MLP` to calculate an anomaly score, representing the abnormal probability of an address. Specifically, the address representation is first fed into a `Linear` layer (with parameters W_3) to map the representation to a scalar value, which is then passed to a `Softmax` layer to calculate the probability of being an abnormal address, i.e., anomaly score Λ in Eqn. (10).

$$\Lambda^j = \sigma(W_3 \mathbf{H}^j) \tag{10}$$

where Λ^j is the anomaly score for the *j-th* address in a batch, σ is the `Softmax` layer. $\mathbf{H}$ is the feature vector obtained from the attention module. By setting up a threshold α ($0.5 \leq \alpha < 1$), the address with anomaly score Λ greater than α is detected as an abnormal address as shown in Eqn. (11).

$$\hat{Y}^j = \begin{cases} normal, & \Lambda^j \leq \alpha, \\ abnormal, & \Lambda^j > \alpha. \end{cases} \tag{11}$$

Ideally, the model will detect all abnormal addresses (*true positive*) and will not detect any normal address as abnormal addresses (*false positive*). However, noises and diversities of real-world data make the design of such a perfect model challenging. Thus, we need to balance between *true positive* and *false positive*. In the real world, *false positive* will significantly damage user experience and the company's reputation because it prevents a user from placing orders for possible services on the company's platform (e.g., shopping on an e-commerce platform). It leads to a more severe impact than ignoring an abnormal address. Therefore, we aim to punish more when the model classifies a normal address as an abnormal one during the training process. Most of the training samples are relatively easier to be classified. We want the model to focus more on the challenging task, i.e., samples easier to be classified as wrong labels. Therefore, We adapt Focalloss [28] as the loss function as shown in Eqn. (12).

$$Loss(p) = -(1-p)^{\lambda} log(p) \tag{12}$$

where p represents the probability of an address being abnormal, and $1 - p$ is the probability of an address being a normal address. λ is the focusing parameter, which decreases the importance of samples that are easier to classify. Therefore, the model can focus on the hard and important samples to improve performance.

4 EVALUATION

In this section, we first introduce the evaluation settings such as datasets and metrics. Then, we conduct three-phase evaluations to verify the effectiveness of `FastAddr` comprehensively. **Phase-1: Offline comparison** with state-of-the-art baselines, which is to demonstrate the effectiveness of `FastAddr` and analyze how each component in `FastAddr` impacts the performance. **Phase-2: Offline A/B test** compares `FastAddr` with a deployed abnormal

address detection model in the industry. **Phase-3: Online A/B test** evaluates FastAddr in a real-world online environment.

4.1 Evaluation Setting

Datasets. (i) Training and validation datasets: We have two training configurations, i.e., training with real data and contrastive address augmentation, and training with real data only. The real dataset consists of 45,000 normal addresses and 5,000 abnormal addresses. (ii) Phase-1 evaluation dataset: We utilize a real-world dataset from an e-commerce logistics platform to evaluate model performance. In total, this dataset contains 98,600 normal addresses and 1,400 abnormal addresses. (iii) Phase-2 evaluation: We compare FastAddr with the platform's abnormal address detection model using 300,000 addresses, consisting of normal and abnormal addresses. (iv) Phase-3 evaluation: We compare FastAddr with the platform's abnormal address detection model based on two-week streaming addresses.

Ground truth. All labeled addresses in real data are collected by the e-commerce logistics platform from accumulated low-frequency reporting data. The ground truth addresses in phase-2 and phase-3 are sampled and manually labeled.

Metrics. The goal is to detect abnormal addresses from massive streaming addresses, which have strict requirements for effectiveness and efficiency. Thus, we utilize five metrics for evaluation.

(i) Area Under the ROC Curve (AUC). As a commonly used metric for abnormal detection, a high AUC indicates a good model in general.

(ii) Precision. Precision is the fraction of true anomalies out of detected anomalies. Let α be the threshold of anomaly score, $Precision(\alpha) = \frac{|\hat{A} \cap A|}{|\hat{A}|} \times 100\%$, $\hat{A}$ is the set of detected anomalies, and A is the ground truth set of anomalies.

(iii) F1. $F1(\alpha) = 2 \times \frac{Precision(\alpha) \times Recall(\alpha)}{Precision(\alpha) + Recall(\alpha)}$, where $Recall(\alpha) = \frac{|\hat{A} \cap A|}{|A|} \times 100\%$. $F1(\alpha)$ is an important metric we focus on because $F1(\alpha)$ is a balance between $Precision(\alpha)$ and $Recall(\alpha)$ [3] and it can better measure the quality of abnormal address detection [43].

(iv) F0.5. $F0.5(\alpha) = 1.25 \times \frac{Precision(\alpha) \times Recall(\alpha)}{0.25 Precision(\alpha) + Recall(\alpha)}$. $F0.5(\alpha)$ is a popular metric where the importance of precision is higher than recall, which satisfies our problem setting.

(v) Detection time. Detection time is the time cost of a model to output a detection result, which measures the model's capability of handling a massive number of address queries.

Baselines. We compare FastAddr with 8 baseline models, i.e., AddrLens (based on length of address sentences), AddrSimi (measures address similarities based on pretrained model MiniLM) [46], TextCNN [23] (consists of three Convolution layers, a Pooling layer, and a Linear layer), FastText [5], EntityDet (based on named entity recognition of spatial entities such as roads and POIs) [20], Transformer [45] (consists of two encoders and each encoder has five attention heads), TextRNN [31] (consists of a bi-directional LSTM layer and a Linear layer), TextRNN_Att (TextRNN with attention mechanism) [53].

Configurations. We train all deep learning models on NVIDIA RTX A4000. We evaluate the model inference speed on Intel(R) Xeon(R) CPU E5-2650 v4 @ 2.20GHz, 12 cores. We set the dimension of word embedding as 100, batch size as 20, and training epoch as 15. The learning rate *lr* is 0.0001, decay step of *lr* *lr_decay_rate* is 0.95, and *lr_dacay_steps* is 1,000,000.

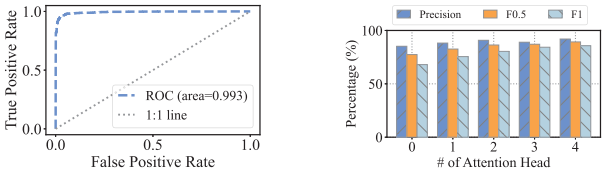
Table 2: Comparison with the state-of-the-art models

Model	AUC	Precision	F1	F0.5	detection time
AddrLens	-	6.3	8.2	9.2	**0.003**
AddrSimi	-	8.2	3.2	5.1	34.2
TextCNN	96.5	11	19.7	13.3	0.12
FastText	98.1	24.2	38.3	28.4	0.058
EntityDet	-	7.59	6.7	9.2	5.87
Transformer	98.4	48.7	62.7	53.5	0.229
TextRNN	99	72.3	75.4	73.5	3.44
TextRNN_Att	99.1	80.3	77.2	79	3.48
FastAddr	**99.3**	**92**	**85.7**	**89.3**	0.058

* The unit for AUC, Precision, F1, and F0.5 is %; for detection time is ms.

4.2 Phase-1 evaluation: Offline Experiments

4.2.1 Overall Performance. To measure the model's effectiveness, we compare FastAddr with the recent state-of-the-art models in five dimensions. The result is described in Table 2. Even though FastAddr has the highest AUC, we argue that AUC alone is not enough to select the best model. As in Table 2, most models have similar high AUC values but different precision, F0.5, and F1. Since precision and recall are both important metrics, we consider F1 (the balance between precision and recall) as the indicator of model effectiveness and speed as the indicator of model efficiency. We conclude several important results: (i) FastAddr and FastText have the shortest detection time because of the model's simplicity. However, FastAddr outperforms FastText by 47.4% of F1 due to its better address representation capability. (ii) Transformer is a representative attention-based large language model. Even with good AUC and F1, the detection time prevents it from being deployed to a CPU environment to support hundreds of millions of daily requests. Moreover, the short length of addresses also limits the advantage of Transformer.



(a) ROC curve of FastAddr. **(b) Impact of attention head.**

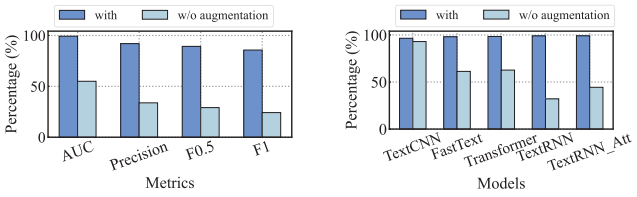
Figure 9: ROC (a) and ablation study on attention (b).

4.2.2 Ablation Study. We conduct extensive ablation studies to evaluate the contribution of each module in FastAddr.

Impact of attention. We investigate how the attention module impacts the abnormal address detection performance. For the AUC value, the model with no attention head achieves 97.3% while all other models have an AUC value of 99.3%, which is not reported in Fig. 9(b) due to space limitations. Fig. 9(b) illustrates the performance in three metrics of models with a different number of attention heads. For all three metrics, the performance improves

with the increase of attention heads when the number of attention heads is less than 4. However, increasing the number of attention heads from 3 to 4 does not bring a significant improvement. The possible reason is that 3 attention heads are good enough to model the important segments in an address sentence, and more attention heads will not boost the performance significantly. Therefore, we adapt 4 attention heads to balance the model complexity and performance. Compared with no attention head, the model with 4 attention heads witnesses an improvement of more than 17% in F1. It demonstrates the effectiveness of the lightweight multi-head attention mechanism in our model.

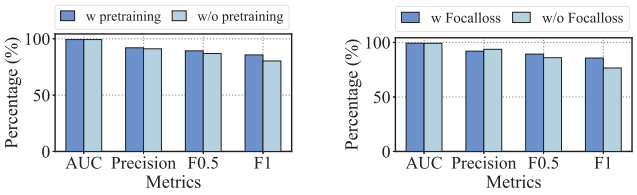
Impact of contrastive address augmentation. We use a manually labeled dataset *w/o augmentation* that consists of 45,000 normal addresses and 5,000 abnormal addresses to train `FastAddr`, and compare it with `FastAddr` trained with contrastive address augmentation (*with augmentation*). The result is shown in Fig. 10(a). The model trained with augmentation significantly outperforms the model trained without augmentation. We also evaluate baseline models that are trained without our augmentation approach as shown in Fig. 10(b). The AUC of all models is significantly decreased except TextCNN, which performs the worst when trained with augmentation. The reason might be that TextCNN has limited representation capability and thus cannot utilize the benefit of augmentation, which leads to a similar performance with or without augmentation. These results demonstrate the effectiveness of our contrastive address augmentation approach.



(a) Impact on `FastAddr`. **(b) Impact on baselines.**

Figure 10: Impact of contrastive address augmentation.

Impact of pre-trained word embedding. In `FastAddr`, we initialize the model with pre-trained word embedding from Tencent [40]. To verify the effectiveness of pre-trained word embedding, we initialize `FastAddr` randomly. Intuitively, utilizing pre-trained embedding would boost the model's performance. In Fig. 11(a), we observe that the model has lower precision, F0.5, and F1, and the same AUC without pre-trained embedding. One possible reason is that there are natural languages in addresses, which benefit the model by using embedding trained on the open-domain corpus.



(a) Impact of pre-training. **(b) Impact of Focalloss.**

Figure 11: Ablation study on pre-training and Focalloss.

Table 3: Offline A/B test result.

Model	True	False	Accuracy
Previous Deployed Model	72	528	12%
Our `FastAddr`	522	78	87%

Impact of Focalloss. We conduct an ablation study to evaluate the impact of Focalloss. Fig. 11(b) compares the model using Focalloss with the model using Cross-Entropy loss. With similar AUC and precision, the model without Focalloss has lower F0.5 and F1, which demonstrates the importance of Focalloss in `FastAddr`.

4.3 Phase-2 evaluation: Offline A/B Test

To further evaluate the model performance, we conduct the phase-2 evaluation to compare `FastAddr` with the online deployed model of the platform. We aim to compare the model performance on the same set of addresses. We randomly sample 300,000 streaming addresses from the online environment and feed them into `FastAddr` to get abnormal address detection results. Among 300,000 addresses, two models have different detection results over 1,750 addresses. We then randomly sample 600 of those addresses and evaluate them manually. The result is shown in Table 3 where *True* represents the model that makes the proper judgment, and *False* represents the model that makes the wrong judgment. `FastAddr` (87%) outperforms the deployed model (12%) by 75% in detection accuracy.

4.4 Phase-3 evaluation: Online A/B Test

After offline A/B test evaluation, we compare `FastAddr` with the deployed model in an online environment by designing a large-scale A/B test. We evaluate the precision, the number of detected abnormal addresses, and detection time over two weeks. We define AD_{rate} to measure the ratio of detected abnormal addresses because *Recall* cannot be calculated in the real-world environment due to the unknown number of abnormal addresses.

$$AD_{rate} = \frac{|\hat{A}|}{|\hat{A}| + |\hat{N}|} \times 100\% \tag{13}$$

where $|\hat{A}|$ is the number of detected abnormal addresses, $|\hat{N}|$ is the number of normal addresses judged by the model. Thus, AD_{rate} represents the percentage of abnormal addresses (detected by the model) in the whole dataset.

Table 4: Online A/B test result.

Model	Precision	AD_{rate}	TP99
Ours	21.5% ↑	25% ↑	4ms (same)

We report the average result in Table 4. We also evaluate the inference speed of `FastAddr` in the real-world environment. The speed is quantified by a standard speed metric, 99% Top Percentile ($TP99$) [30]. $TP99$ consists of model inference time, network delay time, etc. The model is called thousands of times in a time window, and 99% of model inference time is less or equal to $TP99$. Therefore, a low $TP99$ value can ensure a fast inference speed in a real-world environment with 99% of confidence.

4.5 Case study

Credit Card Fraud Detection. Nowadays, credit card companies accept online applications from applicants with information such

as an address, to represent work or home location. An essential feature for determining the reviewing decision is the quality of user-entered addresses because abnormal agents are utilizing abnormal addresses for malicious credit card overdue. Therefore, we can apply FastAddr to detect these abnormal addresses to reduce the potential fraud risk. We evaluate FastAddr on a dataset from an online credit card platform and results show that FastAddr detected 1,208 abnormal addresses among 120,000 addresses, i.e., 1% detection rate AD_{rate}, with a precision of 99%, which demonstrates the real-world impact of FastAddr.

5 DISCUSSIONS

5.1 Lessons learned

(i) Data-driven findings. We have two important findings from the in-depth investigation of real-world datasets. (i) The abnormal address problem has caused significant losses for location-based services (Sec. 2.2.2), which is rarely investigated by existing address and POI studies. (ii) The abnormal address problem is challenging due to data scarcity, the detection speed requirement, and address data complexity in developing countries (Fig. 3), which is significantly different from addresses in developed countries.

(iii) Generalization of FastAddr. Even though FastAddr is designed for location-based services, we believe it has the potential to be generalized to other applications. Our contrastive address augmentation approach can benefit data augmentation in various domains such as medical data. Our lightweight abnormal detection model can be utilized for real-time user query analysis in recommender systems, which can suggest a possible query for users in real-time if it detects an abnormal user query.

5.2 Limitations and future work

Currently, FastAddr detects abnormal addresses without detailed location information successfully. However, FastAddr cannot verify the physical existence of an address, i.e., whether an address truly represents a spatial location in the real world. For example, *Princeton University, Boston City* is not a valid address because there is no *Princeton University* in *Boston*. This is mainly because FastAddr focuses on semantic information of addresses, which does not identify untruthful POI-city mappings. In the future, we plan to detect these addresses by combining spatial knowledge with semantic knowledge in a spatial entity knowledge graph.

6 RELATED WORK

We organize related works into two categories.

Text data Anomaly Detection. Anomaly detection has been a widely studied research topic for years [6, 13, 29]. Even though the research on anomaly detection dates from decades ago and has evolved from machine learning models to [37, 38] deep learning models [7], the study of anomaly detection on text data has not received enough attention. Motivated by the growing importance of text data, a few studies have started working on anomaly detection for text data, which were partially inspired by anomaly detection in the computer vision community [15, 17]. Kannan et al. [22] designed a non-negative matrix factorization method to detect text anomalies. Ruff et al. [36] utilized a pre-trained language model and a vector representation of words to detect text anomalies. Such a one-class anomaly detection model utilized a self-attention mechanism to capture multiple modes in data to detect anomalies. Manolache et al. proposed DATE [32], a Transformer-based model with Replaced Mask Detection to detect anomalies in text. Meanwhile, text anomaly detection can also be transformed into a classification task and solved by classification models [5, 11, 21, 23–25, 45]. These models detected abnormal text data by outputting an anomaly score, which represents the possibility for a sample to be abnormal.

Address and Point-of-Interest (POI). As important components in spatio-temporal data, addresses and POIs have received lots of research interests [9, 16, 35, 39, 41, 44, 51]. One of the most popular research areas is next POI recommendation [9, 51]. Zhao et al. [51] designed a Spatial-temporal Gated Network to capture the spatio-temporal relationship between consecutive check-ins and make the next POI recommendation. Chen et al. [9] proposed a novel transfer model to recommend the next POI in a new city by solving the cold-start problem. Another research direction in this area is address and POI information inference. Ruan et al. [35] proposed a novel method to infer delivery time in logistics. He et al. [16] designed a location-based method to discover POI alias. To obtain structured elements in addresses, some studies focus on the address segmentation problem [27, 47]. Li et al. [27] studied the Chinese address segmentation task and proposed a neural network-based model to identify different address components.

Summary. Our work differs from previous studies in three dimensions. (i) We study the abnormal aspect of addresses, whereas most existing address-related studies assume the addresses are normal. (ii) We focus on abnormal addresses and spatial-related text, whereas most studies detect text anomalies in open domains. (iii) We have designed an efficient and effective abnormal detection framework for detecting abnormal addresses with constrained resources and strict time requirements while most existing studies do not consider resource limitations.

7 CONCLUSIONS

In this paper, we formulate and investigate the abnormal address detection problem based on real-world data. We design FastAddr, which consists of (i) a contrastive address augmentation module to construct large-scale training data by modeling the hierarchical structure of addresses, and (ii) a Lightweight Multi-head Attention Model to learn both effective and efficient address representations. Results show that FastAddr has good performance in offline experiments, online comparison, and a real-world case study.

8 ACKNOWLEDGMENTS

This work is partially supported by NSF 1849238, 1932223, 1951890, 1952096, 2003874, and 2047822. We thank all the reviewers for their insightful feedback to improve this paper.

REFERENCES

[1] 2021. Abnormal address in E-commerce. 2021. http://finance.china.com.cn/roll/20211102/5685321.shtml.
[2] 2022. Gaode Maps. 2022. https://ditu.amap.com/.
[3] Ricardo A. Baeza-Yates and Berthier A. Ribeiro-Neto. 2011. *Modern Information Retrieval - the concepts and technology behind search, Second edition.*

[4] Ane Blázquez-García, Angel Conde, Usue Mori, and Jose A. Lozano. 2021. A Review on Outlier/Anomaly Detection in Time Series Data. *ACM Comput. Surv.* 54, 3, Article 56 (April 2021), 33 pages.

[5] 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.

[6] Varun Chandola, Arindam Banerjee, and Vipin Kumar. 2009. Anomaly detection: A survey. *ACM computing surveys (CSUR)* 41, 3 (2009), 1–58.

[7] Jinghui Chen, Saket Sathe, Charu Aggarwal, and Deepak Turaga. 2017. Outlier detection with autoencoder ensembles. In *Proceedings of the 2017 SIAM international conference on data mining*. SIAM, 90–98.

[8] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *ICML'20*. PMLR, 1597–1607.

[9] Yudong Chen, Xin Wang, Miao Fan, Jizhou Huang, Shengwen Yang, and Wenwu Zhu. 2021. *Curriculum Meta-Learning for Next POI Recommendation.* ACM, 2692–2702.

[10] Ailin Deng and Bryan Hooi. 2021. Graph neural network-based anomaly detection in multivariate time series. In *AAAI'21*, Vol. 35. 4027–4035.

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, 4171–4186.

[12] Yi Ding, Ling Liu, Yu Yang, Yunhuai Liu, Desheng Zhang, and Tian He. 2021. From Conception to Retirement: a Lifetime Story of a 3-Year-Old Wireless Beacon System in the Wild. In *NSDI'21)*. 859–872.

[13] Jose R Dorronsoro, Francisco Ginel, C Sgnchez, and Carlos S Cruz. 1997. Neural fraud detection in credit card operations. *IEEE transactions on neural networks* 8, 4 (1997), 827–834.

[14] Bowen Du, Chuanren Liu, Wenjun Zhou, Zhenshan Hou, and Hui Xiong. 2016. Catch me if you can: Detecting pickpocket suspects from large-scale transit records. In *ACM SIGKDD*. 87–96.

[15] Izhak Golan and Ran El-Yaniv. 2018. Deep anomaly detection using geometric transformations. *arXiv preprint arXiv:1805.10917* (2018).

[16] Tianfu He, Guochun Chen, Chuishi Meng, Huajun He, Zheyi Pan, Yexin Li, Sijie Ruan, Huimin Ren, Ye Yuan, Ruiyuan Li, et al. 2021. POI Alias Discovery in Delivery Addresses using User Locations. In *SIGSPATIAL'21*. 225–228.

[17] Dan Hendrycks, Mantas Mazeika, and Thomas Dietterich. 2018. Deep anomaly detection with outlier exposure. *arXiv preprint arXiv:1812.04606* (2018).

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

[19] Jie Hu, Li Shen, and Gang Sun. 2018. Squeeze-and-excitation networks. In *CVPR'18*. 7132–7141.

[20] Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *arXiv preprint arXiv:1508.01991* (2015).

[21] Prabhu Kaliamoorthi, Sujith Ravi, and Zornitsa Kozareva. 2019. PRADO: Projection Attention Networks for Document Classification On-Device. In *EMNLP-IJCNLP*. Association for Computational Linguistics, 5012–5021.

[22] Ramakrishnan Kannan, Hyenkyun Woo, Charu C Aggarwal, and Haesun Park. 2017. Outlier detection for text data: An extended version. *arXiv preprint arXiv:1701.01325* (2017).

[23] Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. In *EMNLP*. Association for Computational Linguistics, 1746–1751.

[24] Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Recurrent Convolutional Neural Networks for Text Classification. In *AAAI'15* (Austin, Texas) *(AAAI'15)*. AAAI Press, 2267–2273.

[25] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. In *ICLR'20*.

[26] Dongha Lee, Jiaming Shen, SeongKu Kang, Susik Yoon, Jiawei Han, and Hwanjo Yu. 2022. TaxoCom: Topic Taxonomy Completion with Hierarchical Discovery of Novel Topic Clusters. *arXiv preprint arXiv:2201.06771* (2022).

[27] Hao Li, Wei Lu, Pengjun Xie, and Linlin Li. 2019. Neural Chinese address parsing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. 3421–3431.

[28] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2020. Focal Loss for Dense Object Detection. *IEEE PAMI* 42, 2 (2020), 318–327.

[29] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. 2008. Isolation forest. In *ICDM'08*. IEEE, 413–422.

[30] Hu Liu, Jing Lu, Hao Yang, Xiwei Zhao, Sulong Xu, Hao Peng, Zehua Zhang, Wenjie Niu, Xiaokun Zhu, Yongjun Bao, et al. 2020. Category-Specific CNN for Visual-aware CTR Prediction at JD. com. In *KDD'20*. 2686–2696.

[31] Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2016. Recurrent Neural Network for Text Classification with Multi-Task Learning. In *IJCAI'16*. AAAI Press, 2873–2879.

[32] Andrei Manolache, Florin Brad, and Elena Burceanu. 2021. DATE: Detecting Anomalies in Text via Self-Supervision of Transformers. In *NAACL-HLT'21*. 267–277.

[33] Yu Meng, Chenyan Xiong, Payal Bajaj, Paul Bennett, Jiawei Han, Xia Song, et al. 2021. Coco-lm: Correcting and contrasting text sequences for language model pretraining. *Advances in Neural Information Processing Systems* 34 (2021).

[34] Guansong Pang, Chunhua Shen, Longbing Cao, and Anton Van Den Hengel. 2021. Deep learning for anomaly detection: A review. *ACM Computing Surveys (CSUR)* 54, 2 (2021), 1–38.

[35] Sijie Ruan, Zi Xiong, Cheng Long, Yiheng Chen, Jie Bao, Tianfu He, Ruiyuan Li, Shengnan Wu, Zhongyuan Jiang, and Yu Zheng. 2020. Doing in One Go: Delivery Time Inference Based on Couriers' Trajectories. In *KDD'20*. 2813–2821.

[36] Lukas Ruff, Yury Zemlyanskiy, Robert Vandermeulen, Thomas Schnake, and Marius Kloft. 2019. Self-Attentive, Multi-Context One-Class Classification for Unsupervised Anomaly Detection on Text. In *ACL*. Association for Computational Linguistics, 4061–4071.

[37] Bernhard Schölkopf, John C Platt, John Shawe-Taylor, Alex J Smola, and Robert C Williamson. 2001. Estimating the support of a high-dimensional distribution. *Neural computation* 13, 7 (2001), 1443–1471.

[38] Bernhard Schölkopf, Robert C Williamson, Alexander J Smola, John Shawe-Taylor, John C Platt, et al. 1999. Support vector method for novelty detection.. In *NIPS*, Vol. 12. Citeseer, 582–588.

[39] Yatong Song, Jiawei Li, Liying Chen, Shuiping Chen, Renqing He, and Zhizhao Sun. 2021. A Semantic Segmentation Based POI Coordinates Generating Framework for On-Demand Food Delivery Service. In *SIGSPATIAL'21*. ACM, 379–388.

[40] Yan Song, Shuming Shi, Jing Li, and Haisong Zhang. 2018. Directional Skip-Gram: Explicitly Distinguishing Left and Right Context for Word Embeddings. In *NAACL-HLT'18, Volume 2 (Short Papers)*. Association for Computational Linguistics, 175–180.

[41] Vishal Srivastava, Priyam Tejaswin, Lucky Dhakad, Mohit Kumar, and Amar Dani. 2020. A Geocoding Framework Powered by Delivery Data. In *Proceedings of the 28th International Conference on Advances in Geographic Information Systems (SIGSPATIAL '20)*. ACM, 568–577.

[42] Lichao Sun, Congying Xia, Wenpeng Yin, Tingting Liang, Philip Yu, and Lifang He. 2020. Mixup-Transformer: Dynamic Data Augmentation for NLP Tasks. In *ACL*. International Committee on Computational Linguistics, 3436–3440.

[43] Nesime Tatbul, Tae Jun Lee, Stan Zdonik, Mejbah Alam, and Justin Gottschlich. 2018. Precision and recall for time series. *arXiv preprint arXiv:1803.03639* (2018).

[44] Qin Tian, Fu Ren, Tao Hu, Jiangtao Liu, Ruichang Li, and Qingyun Du. 2016. Using an optimized Chinese address matching method to develop a geocoding service: a case study of Shenzhen, China. *ISPRS International Journal of Geo-Information* 5, 5 (2016), 65.

[45] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NeurIPS'17*. 5998–6008.

[46] Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. MiniLM: Deep Self-Attention Distillation for Task-Agnostic Compression of Pre-Trained Transformers. In *NeurIPS 2020, December 6-12, 2020, virtual*.

[47] Vedang A Waradpande, Petchetti Vinay Surya Prakash, Nikhil Jhaveri, and Shashank Agarwal. 2021. Predicting Completeness of Unstructured Shipping Addresses Using Ensemble Models. (2021).

[48] Seonghyeon Ye, Jiseon Kim, and Alice Oh. 2021. Efficient Contrastive Learning via Novel Data Augmentation and Curriculum Learning. In *EMNLP*. Association for Computational Linguistics, 1832–1838.

[49] Mengxi Yu, Ziyu Liu, Yuhang Tang, and Jianfeng Jiang. 2021. Recognition algorithm of e-commerce click farming based on K-means technology. In *2021 6th International Conference on Intelligent Computing and Signal Processing (ICSP)*. 103–106.

[50] Rose Yu, Huida Qiu, Zhen Wen, ChingYung Lin, and Yan Liu. 2016. A survey on social media anomaly detection. *ACM SIGKDD Explorations Newsletter* 18, 1 (2016), 1–14.

[51] Pengpeng Zhao, Anjing Luo, Yanchi Liu, Fuzhen Zhuang, Jiajie Xu, Zhixu Li, Victor S Sheng, and Xiaofang Zhou. 2020. Where to go next: A spatio-temporal gated network for next poi recommendation. *TKDE'20* (2020).

[52] Li Zheng, Zhenpeng Li, Jian Li, Zhao Li, and Jun Gao. 2019. AddGraph: Anomaly Detection in Dynamic Graph Using Attention-based Temporal GCN.. In *IJCAI*. 4419–4425.

[53] Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. 2016. Attention-Based Bidirectional Long Short-Term Memory Networks for Relation Classification. In *ACL'16 (Volume 2: Short Papers)*. Association for Computational Linguistics, 207–212.