Spatio-Temporal Dual Graph Attention Network for Query-POI Matching

Conference Paper · June 2020

DOI: 10.1145/3397271.3401159

7 authors, including:

Zixuan Yuan
Rutgers, The State University of New Jersey
26 PUBLICATIONS **272** CITATIONS
SEE PROFILE

Yanchi Liu
Rutgers, The State University of New Jersey
112 PUBLICATIONS **4,367** CITATIONS
SEE PROFILE

Denghui Zhang
Rutgers, The State University of New Jersey
17 PUBLICATIONS **174** CITATIONS
SEE PROFILE

Nengjun Zhu
Shanghai Jiao Tong University
28 PUBLICATIONS **174** CITATIONS
SEE PROFILE

Spatio-Temporal Dual Graph Attention Network for Query-POI Matching

Zixuan Yuan[1†], Hao Liu[2†*], Yanchi Liu[1], Denghui Zhang[1], Fei Yi[3], Nengjun Zhu[4], Hui Xiong[1*]
[1]Rutgers University, [2]Business Intelligence Lab, Baidu Research, National Engineering Laboratory of Deep Learning Technology and Application, China, [3]Northwestern Polytechnical University, [4]Shanghai Jiao Tong University.
{zy101,yanchi.liu,denghui.zhang,hxiong}@rutgers.edu,liuhao30@baidu.com,
yifei93219@mail.nwpu.edu.cn,zhu_nj@sjtu.edu.cn

ABSTRACT

In location-based services, such as navigation and ride-hailing, it is an essential function to match a query with Point-of-Interests (POIs) for efficient destination retrieval. Indeed, due to the space limit and real-time requirement, such services usually require intermediate POI matching results when only partial search keywords are typed. While there are numerous retrieval models for general textual semantic matching, few attempts have been made for query-POI matching by considering the integration of rich spatio-temporal factors and dynamic user preferences. To this end, in this paper, we develop a *spatio-temporal dual graph attention network* (STDGAT), which can jointly model dynamic situational context and users' sequential behaviors for intelligent query-POI matching. Specifically, we first utilize a semantic representation block to model semantic correlations among incomplete texts as well as various spatio-temporal factors captured by location and time. Next, we propose a novel dual graph attention network to capture two types of query-POI relevance, where one models global query-POI interaction and another one models time-evolving user preferences on destination POIs. Moreover, we also incorporate spatio-temporal factors into the dual graph attention network so that the query-POI relevance can be generalized to the sophisticated situational context. After that, a pairwise fusion strategy is introduced to extract the salient global feature representatives for both queries and POIs. Finally, several cold-start strategies and training methods are proposed to improve the matching effectiveness and training efficiency. Extensive experiments on two real-world datasets demonstrate the performances of our approach compared with state-of-the-art baselines. The results show that our model achieves significant improvement in terms of matching accuracy even with only partial query keywords are given.

KEYWORDS

Query-POI Matching, Spatio-temporal Analysis, Dual Graph Neural Network, User Modeling

ACM Reference Format:
Zixuan Yuan, Hao Liu, Yanchi Liu, Denghui Zhang, Fei Yi, Nengjun Zhu, Hui Xiong. 2020. Spatio-Temporal Dual Graph Attention Network for Query-POI Matching. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'20), July 25–30, 2020, Virtual Event, China.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3397271.3401159

1 INTRODUCTION

Query and Point-of-Interest (POI) matching has become an essential retrieval service in various location-based applications, e.g., Google Maps, Baidu Maps, and Uber. As illustrated in Figure 1, query-POI matching aims to retrieve the destination POI from a list of candidate POIs based on (incomplete) query keywords. Since the

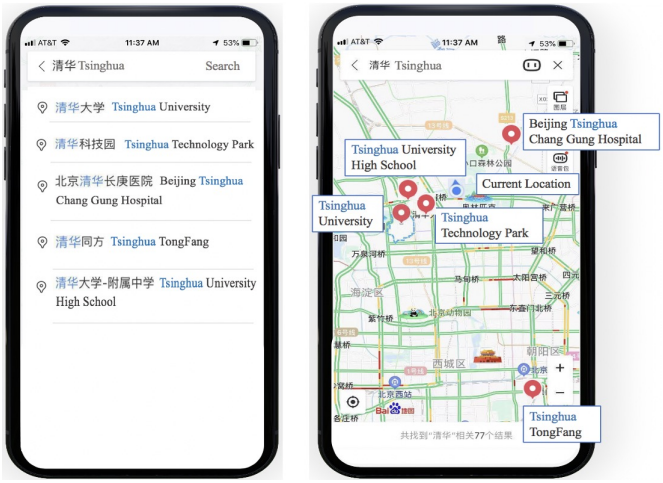


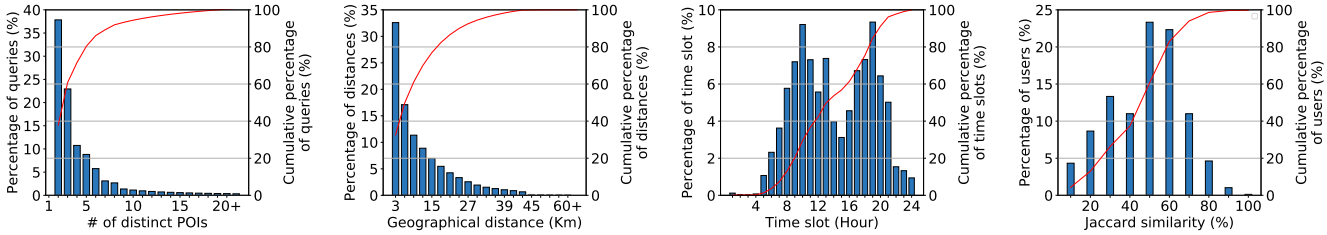
Figure 1: An illustrative example of query-POI matching on Baidu Maps. Left side shows the matched POI candidate list of given query, and right side depicts locations of these POIs.

retrieval result directly involves the user's travel decision process, accurate and predictive query-POI matching can significantly improve user experience and ultimately boost the commercial benefits for these location-based applications.

† Zixuan Yuan and Hao Liu contributed equally to this work.
* Hao Liu and Hui Xiong are corresponding authors.

SIGIR '20, July 25–30, 2020, Virtual Event, China

ACM ISBN 978-1-4503-8016-4/20/07…$15.00
https://doi.org/10.1145/3397271.3401159



(a) Matching uncertainty. **(b) Spatial effect.** **(c) Temporal effect.** **(d) Jaccard similarity.**

Figure 2: Characteristics of query-POI matching records: (a) Query-POI matching uncertainty distribution; (b) Distance distribution between current location and destination POI location; (c) Temporal distribution of queries; (d) Jaccard similarity distribution of user preference between consecutive weeks.

Despite its ubiquity and importance, only a few efforts have been made for dedicated query-POI matching. General information retrieval approaches usually project queries and POIs into a vector space [21] or link them via a probabilistic model [19] for matching. Recently, deep learning based methods have been introduced for search and retrieval systems, by learning more effective representations for both queries and items [17, 23]. For example, DSSM [9] maps queries and documents to the same latent semantic space and computes the relevance for each query-document pair. Based on DSSM, PALM [36] further models static geographical correlations between query-POI pair. Despite their effectiveness on general query-item matching, we argue these methods are not sufficient enough to deliver satisfactory user experience for dynamic query-POI matching. The main reason is that the destination POI of same query keywords may vary over different users under different spatio-temporal contexts. Take the query in Figure 1 for example, when typing “Tsinghua" on navigation apps, a college student search at morning rush hour may intend to go to *Tsinghua Univeristy* for class, while a patient search at midnight is with higher probability to go to *Beijing Tsinghua Chang Gung Hospital* for treatment.

While it is intuitively useful to incorporate spatio-temporal factors and user preference, dynamic query-POI matching is a non-trivial task because of the following three challenges: 1) **Matching uncertainty**. In practice, a POI name may correspond to multiple POIs, according to large-scale data analysis, over a half queries correspond to multiple POIs, as reported in Figure 2(a). In fact, the distribution of query-POI matching uncertainty nearly follows the power law distribution with $\alpha = 0.56$ [3]. More severely, most query inputs are incomplete POI names, which further increases the uncertainty of the destination POI. Therefore, the first challenge is how to reduce the matching uncertainty between (incomplete) query keywords and POIs. 2) **Situational context complicacy**. In query-POI matching, we identify strong dependencies between the destination POI intention and the situational context [39]. For example, as shown in Figure 2(b), for the majority queries, the current location and destination POI location is within a relative short distance [16]. Figure 2(c) depicts the bimodal distribution of query-POI interactions, where most query-POI interactions happen in the morning and evening rush hours. Besides, since the context factors may appear to be extremely high dimensional (e.g., locations and time), nearly a half queries are under unseen situational context (e.g., search a POI from a new location). How to model and generalize the destination POI intention under complex situational context is the second challenge. 3) **User preference dynamicity**. Due to the locality and temporal dependency of human mobility [33], the intended POI is user-specific and evolving over time [31]. For illustration, we randomly sample 5,000 anonymized users and partition their historically clicked POIs to eight chronically ordered sets (each set represent one week data). Figure 2(d) reports the distribution of Jaccard similarities between two consecutive sets of each user. As can be seen, the Jaccard similarity of most users' preference falls in a moderately high percentage interval, which conveys the temporal dependent and evolving user preference drift. As a result, it is challenging to model time-evolving user preference from limited user historical query data.

To tackle the above challenges, we propose the *Spatio-temporal Dual Graph Attention Network* (STDGAT) for intelligent query-POI matching. First, we introduce a semantic representation block that projects sparse query words, POI names as well as multiple spatio-temporal factors into a unified latent space. After that, we propose a dual graph attention network to collaboratively model query-POI relevance. Specifically, the generic query-POI graph attention captures the global query-POI correlation from the bipartite graph connecting all queries and POIs. While the user-specific graph attention captures the time-evolving user preference on destination POIs from the bipartite graph structure connecting user-specific queries and POIs. Spatio-temporal factors such as geographical location and time slot are also incorporated to generalize the query-POI relevance modeling under unseen situational contexts. Furthermore, a pairwise neuron fusion method fuses learned representations through a feed-forward neural network. Last but not least, STDGAT adopts a simple yet effective strategy to handle cold start problems for new users, queries and POIs.

Our contributions are summarized as follows:

- We propose a novel framework for intelligent query-POI matching. By collaboratively incorporating various textual information and situational contexts, our framework is capable of matching incomplete queries and POIs by considering dynamic situational context.
- We develop a novel dual graph attention network to model query-POI relevance from both the generic perspective and

the user-specific perspective, with consideration of the sophisticated situational context.
- We address the cold start problem in query-POI matching and propose several training techniques to improve matching effectiveness and training efficiency.
- We evaluate STDGAT on two real-world datasets, the results demonstrate the effectiveness of our approach compared with six state-of-the-art baselines.

2 PRELIMINARIES AND PROBLEM STATEMENT

In this section, we first introduce some important notations and definitions, then formally define the query-POI matching problem. Notations frequently used in this paper are listed in Table 1.

Table 1: Table of notations.

Notations	Description
m, n	The number of queries and POIs.
Q, P	The sets of all queries and all POIs.
$q = (l_q, qk)$, $q \in Q$	A query is a 2-tuple: location l_q, query word qk.
$p = (l_p, pn, c)$, $p \in P$	A POI is a triplet: location l_p, POI name pn, and POI category c.
$l_q = \{lng_1, lat_1\}$	The location of query q is represented by longitude lng_1 and latitude lat_1.
$l_p = \{lng_2, lat_2\}$	The location of POI p is represented by longitude lng_2 and latitude lat_2.
$U = \{u_1, u_2, \ldots, u_N\}$	The set of all users.
$mq = \{u, \tau, q, p\}$	A map search query event.
G^g	The generic query-POI graph.
$G^s_{u,t}$	The user-specific query-POI graph for user u at the t-th time period.
gv	The geographical embedding for a query-POI pair.
τ	The time slot/hour zone.
$\mathbf{X}_Q$, $\mathbf{X}_P$	The semantic representations of queries and POIs.
$\mathbf{X}_{\hat{Q}}$, $\mathbf{X}_{\hat{P}}$	The generic embeddings of queries and POIs.
$\mathbf{X}_{\hat{Q}^s}$, $\mathbf{X}_{\hat{P}^s}$	The user-specific embeddings of queries and POIs.
$\mathbf{X}_{Q^*}$, $\mathbf{X}_{P^*}$	The ultimate embedding matrices of queries and POIs respectively.

Let u denote an individual user, τ denote a timestamp. We define the map query $q \in Q$ as a 2-tuple (l_q, qk), where l_q is the location of query, qk is the query keyword, and define the Point-of-Interest (POI) $p \in P$ as a 3-tuple (l_p, pn, c), where l_p is the location of POI, pn is the POI name, and c is the category of p. Furthermore, the location l, e.g., l_q and l_p, is represented by a geographical coordinate (lng, lat).

Definition 1. ***Map query event***. *A map query event is defined as a 4-tuple $mq = \{u, \tau, q, p\}$, such that a user u issues a map query q at time slot τ, and clicks on a candidate POI p.*

Note that given a map query, there may exist multiple POI candidates and a user may click on multiple different POIs, which will generate multiple map query events for each click on different POIs. Given a set of map queries $Q = \{q_1, q_2, \cdots, q_n\}$ and a set of POIs $P = \{p_1, p_2, \cdots, p_m\}$, we construct two query-POI interaction graphs as follow. Specifically, given a map query event mq, we call the click action as a *query-POI interaction.*

Definition 2. ***Generic query-POI graph*** *is defined as $G^g = (V, E)$, where g is the generic symbol, $V = Q \cup P$ and E is a set of edges, indicating all query-POI interactions. Formally, we define $e_{ij} \in E$ as*

$$e_{ij} = \begin{cases} 1, & v_i \in Q, v_j \in P, freq(v_i, v_j) \geq \delta \\ 0, & otherwise \end{cases}, \quad (1)$$

where $freq(v_i, v_j)$ is the overall frequency of query-POI interactions between query $v_i \in Q$ and POI $v_j \in P$, δ is a threshold.

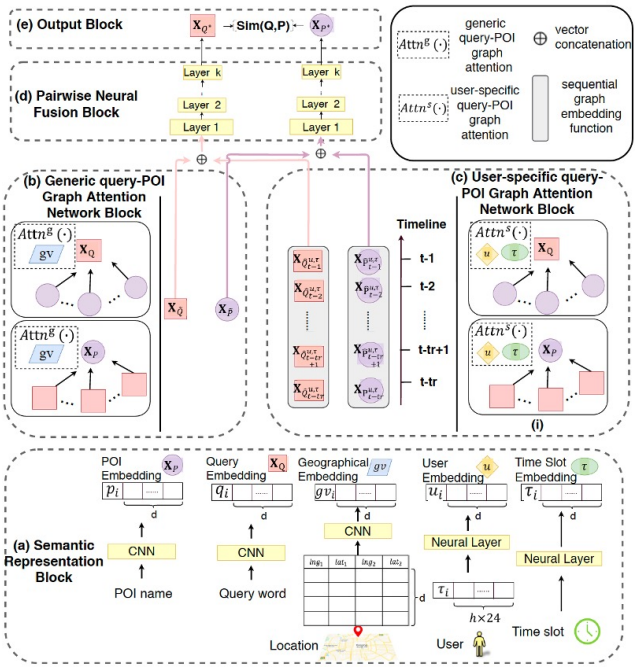


Figure 3: Framework overview.

In practice, δ filters out low-frequency query-POI interactions, which will induce minor discrepancy among vertex representations since most of the frequencies fall in a small range of value [32]. Note that the generic query-POI graph is an unweighted, undirected bipartite graph.

Definition 3. ***User-specific query-POI graph*** *is defined as $G^s_{u,t} = (V, E)$, where s is the user-specific symbol, E is a set of edges among $V = Q_{u,t} \cup P_{u,t}$, indicating query-POI interactions made by user u in the time-period t. Formally, we define $e_{ij} \in E$ as $e_{ij} = freq_{u,t}(v_i, v_j)$, where $freq_{u,t}(v_i, v_j)$ is the frequency of query-POI interactions between query $v_i \in Q_{u,t}$ and POI $v_j \in P_{u,t}$.*

Note that for each distinct user, there exists a corresponding user-specific query-POI graph. The user-specific query-POI graph models the user's preference in recent map query behaviors. Slightly different from the generic query-POI graph, a user-specific query-POI graph is a weighted, undirected bipartite graph. We preserve the edge weight to ease the subsequent learning process of user preference and therefore alleviate the data sparsity problem in each user-specific graph.

Problem 1. ***Query-POI matching problem***. *Given a map query q made by user u at time slot τ and location l, we aim to estimate the most relevant POI p that the user may click through based on historical map query events.*

3 FRAMEWORK OVERVIEW

Figure 3 shows the overall framework of STDGAT. It consists of five components, i.e., the *Semantic representation* block, the *Generic query-POI graph attention network* block, the *User-specific query-POI graph attention network* block, the *Pairwise neural fusion* block, and the *Output* block.

Given a set of map query events, the *Semantic representation* block first generates low dimensional embedding vectors for five types of input elements, including: 1) tokenized queries, 2) tokenized POIs, 3) geographical locations, 4) time slots, and 5) anonymized user IDs. Then, the *Generic query-POI graph attention network* block and the *User-specific query-POI graph attention network* block generate the generalized geographical aware and user-specific temporal aware representation embeddings based on the generic query-POI graph and the user-specific query-POI graph, respectively. After that, for each query-POI pair, the *Pairwise neural fusion* block fuses their corresponding generalized and user-specific representation embeddings. Finally, the *Output* block computes the matching scores between each query and corresponding candidate POIs.

4 METHODOLOGY

In this section, we introduce each component of STDGAT in detail.

4.1 Semantic representation

As shown in Figure 3, for each map query event, we first extract five types of input features (i.e., query words, POI names, geographical locations, time slots, and users). The semantic representation block projects high dimensional categorical features into low dimensional dense vectors, which preserves latent semantic correlations among input features. Note that these input features are treated as model input. Rather than training a separate embedding model such as word2vec [18], we adopt a convolutional neural network (CNN) [13] for dimension reduction, the semantic representation block is optimized with other components simultaneously, along with the supervision of the query-POI matching task.

For queries and POIs, we split each query words and POI names into tokenized characters and words [15, 38], transform each character and word into randomly-initiated vectors, and finalize the embeddings of queries and POIs with these term-level vectors through CNN. Conceptually, the character-level representation conserves inherent connections of incomplete query words and POI names, while the word-level representation captures semantic correlations among queries and POIs under various contexts. The textual semantic embedding produced by CNN operation is shared by both query and POI, which enhances the matching accuracy between incomplete/full query keyword and POI name.

Then, we extract the current location $\{lng_1, lat_1\}$ of the user, the destination POI location $\{lng_2, lat_2\}$ of clicked POI, and time slot τ in each map query event. All locations are projected to a set of grid-based regions [25] and encoded into randomly-initiated encoded vectors. Instead of simply embedding each longitude and latitude, for each location, we further incorporate their neighboring region longitude embeddings and latitude embeddings to capture local dependencies and mitigate the boundary conflicts [36]. Specifically, we derive the embedding of a given longitude/latitude ϕ_i as follow:

$$\phi_i' = w_{i-1}\phi_{i-1} + w_i\phi_i + w_{i+1}\phi_{i+1}, \tag{2}$$

where ϕ_i is the embedding of current longitude/latitude, ϕ_{i-1} and ϕ_{i+1} are its neighboring longitude/latitude embeddings, ϕ_i' is the updated embedding for ϕ_i, w_{i-1}, w_i and w_{i+1} are learnable parameters. Then, we apply the convolution operation to derive the geographical embedding from the aggregated embeddings of longitude and latitude. In this way, the geographical information in each map query event is projected to a d-dimensional dense vector. Similarly, for time slot τ, we first map it into predefined time slots, then project it into a d-dimensional vector by incorporating its neighboring time slot embeddings.

For each user, we first partition POIs into h predefined POI categories (e.g., residential, transport, education, etc.), and divide the whole day into 24 time slots. Then, we generate a $h\times24$-dimensional vector for each user, where the $i \times j$-th dimension is the portion of the user's clicks on the i-th category POIs in the j-th time slot. Finally, we feed the user vector into a fully connected layer to obtain the d-dimensional user embedding. Note that the user embeddings of the same user in different map query events are identical.

4.2 Dual graph attention network

Then we introduce the dual graph attention network, which captures pairwise correlation among queries and POIs based on both the generic query-POI graph and the user-specific query-POI graph. In past years, graph neural network (GNN) [12] has shown its superiority on processing non-Euclidean correlated graph structures [26, 28, 40]. In a word, for each vertex (i.e., query or POI), GNN applies aggregation and transformation operation on its neighbors to obtain new representations. Since the correlation of neighboring vertex may vary non-linearly, we adopt the graph attention network (GAT) [24], an attention based variant of GNN to capture correlations among queries and POIs in same latent space. Specifically, we propose the generic query-POI graph attention operation as well as the user-specific query-POI graph attention operation to capture the query-POI relevance from their corresponding graph structures. Additionally, we incorporate spatiotemporal dynamics from the semantic representation block into the dual graph attention network to guide the optimization of the attention coefficient. In this way, the learned representations of queries and POIs capture generic and user-specific query-POI relevance and are generalizable under dynamic situational context.

4.2.1 Generic query-POI graph attention. Based on learned semantic representations, we first introduce the generic query-POI graph attention operation, which captures the global relevance among queries and POIs. Consider the semantic representations of query q and POI p, we update the representation vectors based on the generic query-POI graph G^g as follow:

$$\mathbf{X}_{\tilde{Q}} = \sigma(A^Q(G^g)W^Q\mathbf{X}_P + b), \tag{3}$$

$$\mathbf{X}_{\tilde{P}} = \sigma(A^P(G^g)W^P\mathbf{X}_Q + b), \tag{4}$$

where $\mathbf{X}_{\tilde{Q}}$ and $\mathbf{X}_{\tilde{P}}$ are the generic embeddings of queries and POIs. σ is the activation function. W^Q and W^P are respectively weight matrices for queries Q and POIs P, b is the bias. $A^Q(G^g)$ and $A^P(G^g)$ are proximity matrices derived from an attention mechanism based on the generic graph G^g, by incorporating the geographical influence. Each $a^Q_{qp} \in A^Q(G^g)$ and $a^P_{pq} \in A^P(G^g)$ are defined as:

$$a^Q_{qp} = \frac{Attn^g(\mathbf{x}_q, \mathbf{x}_p, e_{qp}, gv)}{\sum_{v_{u_p}\in N(v_q)} Attn^g(\mathbf{x}_q, \mathbf{x}_{u_p}, e_{qu_p}, gv)}, \tag{5}$$

$$a^P_{pq} = \frac{Attn^g(\mathbf{x}_p, \mathbf{x}_q, e_{qp}, gv)}{\sum_{v_{u_q}\in N(v_p)} Attn^g(\mathbf{x}_p, \mathbf{x}_{u_q}, e_{u_qp}, gv)}, \tag{6}$$

where gv is the geographical embedding for the corresponding query-POI pair. For a given edge e_{qp}, $\mathbf{x}_q$, $\mathbf{x}_p$, $\mathbf{x}_{u_p}$ and $\mathbf{x}_{u_q}$ are the input embeddings of node v_q, node v_p and their 1-hop neighbours $v_{u_p} \in N(v_q)$, $v_{u_q} \in N(v_p)$, respectively. $Attn^g(\cdot)$ is the attention function defined as:

$$Attn^g(a, b, c, d) = LeakyRelu(W_{abc}(W_{ab}(a \parallel b) \otimes W_c c) \parallel W_d d), \quad (7)$$

where W_{ab}, W_{abc}, W_c, W_d denote learnable weighted matrices. Specifically, the weighted matrix W_{ab} represents weighted matrix of a after combined with b, W_{abc} stands for the joint weighted matrix of a, b, and c. The weighted matrix W_c is used to adjust the edge weight of edge e_{qp}, and W_d is denoted as the weighted matrix to measure the geographical influence. $\otimes$ and $\parallel$ represent the element-wise multiplication operation and concatenation operation, respectively. Note that for any query-POI pair, both static semantic similarities and geographical correlations in generic query-POI attention operation jointly determine the query-POI attention weights a^Q_{qp} and a^P_{pq}.

4.2.2 User-specific query-POI graph attention. Besides the global correlation, the relevance among queries and POIs is also user-dependent and temporal-aware. We further introduce the user-specific query-POI graph attention operation to model the time-evolving user preference.

Similar to the generic query-POI graph attention operation, a graph attention operation can be applied on each user-specific query-POI graph to learn different attention weights to measure the relevance of query-POI pair w.r.t. different users. Consider a set of user-specific query-POI graphs, $\{G^s_{u,t-1}, \ldots, G^s_{u,t-tr}\}$, where tr is the number of involved time periods. We further incorporate temporal influence to model dynamic user preferences under different time periods. Inspired by successful applications of autoregressive moving average (ARMA) principle [27] on processing sequential graphs [14], we describe the linear temporal correlations between current POIs and previous click sequences from the following two aspects: (1) AR(t_1) considers linear dependency between current destination POI and historical sequential query-POI interactions. (2) MA(t_2) measures the effects of white noise, i.e., external variations that is only seen indirectly, via regression based on user's historical sequential query-POI interactions.

$$\mathbf{X}_{\hat{P}^{u,\tau}_t} = \sum_{i=1}^{t_1} \mathbf{X}_{\hat{P}^{u,\tau}_{t-i}} Z_i + \sum_{i=1}^{t_2} \varepsilon^{u,\tau}_{t-i} Z'_i, \quad (8)$$

$$\mathbf{X}_{\hat{Q}^{u,\tau}_t} = \sum_{i=1}^{t_1} \mathbf{X}_{\hat{Q}^{u,\tau}_{t-i}} Z_i + \sum_{i=1}^{t_2} \varepsilon^{u,\tau}_{t-i} Z'_i, \quad (9)$$

where $\mathbf{X}_{\hat{P}^{u,\tau}_{t-i}}$ and $\mathbf{X}_{\hat{Q}^{u,\tau}_{t-i}}$ are learned user-specific representation vectors of POIs and queries made by user u in the time slot τ at the $t-i$-th time period. $\varepsilon^{u,\tau}_{t-i}$ stands for the matrix at i-step ahead of t, which is generated from white noise. In addition, Z_i, Z'_i are the i-th weight matrices. The first term in both Equation 8 and Equation 9 serves as autoregressive factor to take the sequential influence from past self terms into account, while the second one acts as moving average function by absorbing the effects from contextual noise.

To further improve model interpretability and non-linear temporal dependency, we propose the user-specific graph attention operation by extending Equation 8 and Equation 9 from two aspects. Due to space limit, we explain the devised learning process of POIs below, the learning process of queries is identical.

For the first aspect, we replace the white noise term to representations of corresponding queries and POIs,

$$\mathbf{X}_{\hat{P}^{u,\tau}_t} = Attn^s(u, \tau, \mathbf{X}_{P_t}), \quad (10)$$

where $Attn^s(x, y, z) = softmax(\frac{xy^T}{\sqrt{d}}z)$ is the attention function that collaboratively quantifies the influence of user attribute and time slot onto the query-POI embeddings. d is denoted as the dimension of representation vector, and $\mathbf{X}_{P_t}$ represents the semantic representations of user's searched POIs at time period t.

For the second aspect, we define a *sequential graph embedding function* to model the temporal dependency of current query-POI interaction with previous query-POI interactions. Specifically, we first devise the white noise term with the corresponding POI representation vectors,

$$\mathbf{Y}^{u,\tau}_{t+1} = \sum_{k=0}^{K_1-1} \psi_k(A_t)\mathbf{Y}^{u,\tau}_t W_k + \mathbf{X}_{\hat{P}^{u,\tau}_t} Z_0. \quad (11)$$

Since A_t is the adjacent matrix that reflects the connectivity among nodes, $\psi_k(A_t) = A^k_t$ records the k-path reachable nodes and is utilized to compute a k-neighbour (or k-scale) influence effect for a given node. K_1 is the kernel scale of neighborhood which is taken into account. Z_0 and $W_k, k \in [1, K_1 - 1]$ are weight matrices. Compared with the white noise, the hidden state $\mathbf{Y}$ memorizes the representations of POIs in previous steps. Then, the 1-step ahead predicted embedding of POI can be obtained as follow,

$$\mathbf{X}_{\hat{P}^{u,\tau}_{t+1}} = \mathbf{Y}^{u,\tau}_{t+1} + \sum_{k=1}^{K_2-1} \psi_k(A_t)\mathbf{X}_{\hat{P}^{u,\tau}_t} Z_k, \quad (12)$$

where K_2 is another kernel scale of neighborhood. $Z_k, k \in [1, K_2-1]$ is denoted as weight matrix. Moreover, $\mathbf{Y}^{u,\tau}_{t+1}$ and $\mathbf{X}_{\hat{P}^{u,\tau}_{t+1}}$ are respectively regarded as the hidden state and output state of predicted personalized t-th POI embedding at time slot τ. Finally, as shown in Figure 3, a temporal convolution operation is applied on a set of predicted POI embeddings:

$$\mathbf{X}_{\hat{P}^{u,\tau}} = Conv(\mathbf{X}_{\hat{P}^{u,\tau}_{t-tr}}, \mathbf{X}_{\hat{P}^{u,\tau}_{t-tr+1}}, \ldots, \mathbf{X}_{\hat{P}^{u,\tau}_{t-1}}), \quad (13)$$

where $Conv(\cdot)$ is the convolution operation on the sequence of 1-step forward predicted POI embeddings. Such convolution operation captures the non-linear dependency of current query-POI interactions on previous query-POI interactions, which further enhances the matching performance by incorporating user's historical preference. Overall, the *sequential graph embedding function* is a combination of Equation 11, 12, and 13.

4.3 Pairwise neural fusion and ranking

We further propose a pairwise neural fusion block to ensemble both generic and user-specific effects and feed the combined representation to the relevance ranking block. Specifically, we employ a dual-tower structure to extract the salient global feature representatives for both queries and POIs. The learned generic representations of POIs $\mathbf{X}_{\tilde{P}}$ and queries $\mathbf{X}_{\tilde{Q}}$, along with user-specific representations of POIs $\mathbf{X}_{\hat{P}^{u,\tau}}$ and queries $\mathbf{X}_{\hat{Q}^{u,\tau}}$ are fused through two different

neural networks [29]:

$$\mathbf{X}_{P^*} = \phi_S^P(\dots\phi_2^P(\phi_1^P(\mathbf{X}_{\hat{P}^{u,\tau}} \parallel \mathbf{X}_{\tilde{P}}))), \quad (14)$$

$$\mathbf{X}_{Q^*} = \phi_S^Q(\dots\phi_2^Q(\phi_1^Q(\mathbf{X}_{\hat{Q}^{u,\tau}} \parallel \mathbf{X}_{\tilde{Q}}))), \quad (15)$$

$$\phi_s^P(\mathbf{x}) = \sigma(W_s^P\mathbf{x} + b_s^P), s \in [1, S], \quad (16)$$

$$\phi_s^Q(\mathbf{x}) = \sigma(W_s^Q\mathbf{x} + b_s^Q), s \in [1, S], \quad (17)$$

where $\parallel$ is the vector concatenation operation. σ is the sigmoid activation function. $\phi_s^P, \phi_s^Q \in \mathbb{R}^{d_s \times d_{s-1}}$ denote the neural networks for POIs and queries on the s-th layer respectively, in which W_s^P, $W_s^Q \in \mathbb{R}^{d_s \times d_{s-1}}$ denote weight matrices of POIs and queries, b_s^P, $b_s^Q \in \mathbb{R}^{d_s \times d_s}$ are bias vectors of POIs and queries.

After that, we employ the simple yet effective cosine similarity to calculate the relevance between the query and the POI candidate,

$$S(Q, P) = cos(\mathbf{X}_{Q^*}, \mathbf{X}_{P^*}) = \frac{\mathbf{X}_{Q^*}\mathbf{X}_{P^*}}{\parallel \mathbf{X}_{Q^*} \parallel\parallel \mathbf{X}_{P^*} \parallel}. \quad (18)$$

Finally, we rank each POI candidate based on the computed scores and return the ranked list as the matching result. The refined query embeddings Q^* and POI embeddings P^* can also be used for future query-POI matching.

4.4 Handling cold start problem

In this subsection, we discuss the cold-start issue in query-POI matching. Based on the node type, the cold start problem can be categorized into three classes, namely new users, new queries, and new POIs.

For cold start users, STDGAT can naturally handle new coming users solely based on the generic query-POI graph and the situational context. For cold start queries, we propose a two-step K-nearest neighbors (KNN) based approach to generate a corresponding representation that contains both semantic and situational context information. First, we generate the semantic representation of the new query based on character and word level semantic embeddings, and retrieve top-k semantically similar queries in the generic query-POI graph according to the Pearson Correlation [1]:

$$PearSim(x, y) = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^n (x_i - \bar{x})}\sqrt{\sum_{i=1}^n (y_i - \bar{y})}}, \quad (19)$$

where $x = (x_1, x_2, \dots, x_n)$ and $y = (y_1, y_2, \dots, y_n)$ are query semantic embeddings, $\bar{x} = \frac{1}{n}\sum_{i=1}^n x_i$, and $\bar{y} = \frac{1}{n}\sum_{i=1}^n y_i$. Second, we aggregate the learned query representation from STDGAT through a linear averaging function as the final representation of the cold start query. For cold start POIs, we apply a similar method for cold start queries, but restrict the KNN neighbors to existing POIs.

5 TRAINING AND OPTIMIZATION

In this section, we introduce the training techniques to improve the model training process.

First, STDGAT aims to minimize the error of predicted similarity score [36] obtained from Equation 18.

To further improve the discriminative power, we employ negative sampling [4] for training set augmentation. For each query-POI pair, let $\{P_k^+\}$ denote a clicked positive sample, we randomly select four unclicked POIs $\{P_k^-\}$ as negative samples, and move the embeddings of query away from the ones of POI. Then, given a query, the probability of a POI to be clicked is calculated by

$$O_1 = -log \prod_{Q,P^+} Pr(P^+|Q), \quad (20)$$

$$Pr(P|Q) = \frac{exp(S(Q, P))}{\sum_{P' \in \{P^+\} \cup \{P_j^-\}} exp(S(Q, P'))}. \quad (21)$$

Additionally, a $L1$ regularizer is applied to avoid overfitting by constraining the parameter space to be sparse. Since we observe most queries are correlated with a small range of POIs and have no interactions with the rest, we further downscale and segment the generic query-POI graph into multiple independent query-POI subgraphs. In this way, the computational complexity of the regularization loss is reduced from $O(N^2)$ to $O(Kn^2)$, where K is the number of subgraphs and $n \ll N$. However, since we need to calculate the gradient of regularization over every query and POI node in the subgraph, the computation complexity still remains high. Inspired by the success of mini-batch regularization [6] that constrains the embedding parameters of users and item nodes with their neighbor nodes, we define the approximated mini-graph user-based version of the $L1$ regularization as follow,

$$O_2 = \sum_{g \in G}(\sum_{u \in U}(\sum_{Q} \parallel \mathbf{X}_{Q^*} \parallel + \sum_{P} \parallel \mathbf{X}_{P^*} \parallel)), \quad (22)$$

where G represents the collection of independent query-POI subgraphs. Overall, the optimization objective function is defined as

$$O = O_1 + \lambda O_2, \quad (23)$$

where λ is the hyper-parameter controls the importance of the $L1$ regularizer.

6 DISCUSSIONS

In this section, we discuss model deployment issues as well as the key insight and limitations of STDGAT.

6.1 Model deployment

It is crucial to provide efficient and scalable online matching service to users. In our scenario, the online service can be partitioned into the offline phase and the online phase. In the offline phase, all map query events are stored in an offline data warehouse, and the model is trained and updated daily. To exclude seasonal user preference and situational context change, we define a two-month sliding window for training data selection. In the online phase, we employ BRPC[1], a scalable web service framework used throughout Baidu for online service. Once the model training is finished, the model is duplicated to multiple data centers in different regions to reduce the network latency from different geographical locations and balance the workload. In the online phase, the model takes 67.11MB memory space, and the averaged matching latency is $1.99ms$.

[1] https://github.com/apache/incubator-brpc

6.2 Model insights and limitations

STDGAT captures pairwise correlations among queries and POIs based on both generic query-POI graph and user-specific query-POI graph. Specifically, given a query, the generic query-POI graph attention assigns larger weights to POI candidates that have strong semantic similarity as well as follow the global situational context distribution. Such generic attention measures the relations among queries and POIs from a global perspective. On the other hand, in the user-specific graph, the weights assigned by user-specific attention capture users' time-varying preferences. For each time period, it quantifies the query-POI correlation with user information and temporal information. Besides, the sequential graph embedding function that consists of ARMA-based model and convolution operations simultaneously captures the higher-order temporal correlations among the query-POI pair. Since current model is updated by day, as a result, the model may exclude short-term user preference (e.g., in-session user interest [10]) and unexpected event influence, which we left as future work.

7 EXPERIMENTS

In this section, we conduct extensive experiments on two real-world large-scale datasets to evaluate: (1) overall performance of STDGAT, (2) parameter sensitivity, (3) influence of query incompleteness, and (4) performance on handling cold start problem.

7.1 Experimental setup

Data description. We use two real-world large-scale datasets, Beijing and Shanghai, to evaluate our model. All data are randomly sampled from 60 consecutive days in 2019. The averaged length of POI names are 10.23 and 10.36, respectively. We chronologically order each dataset, split the training set and validation set by 80% and 10%, and left the rest as the test set.

Implementation details. We use the PaddlePaddle platform to implement STDGAT. All locations are projected to $1,000 \times 1,000$ grid, and all timestamps are projected to 24 time slots. The dimension of all representation vectors d is fixed to 64, and we use 1-layer CNN and 1-layer neural network in semantic representation block. We set the regularization coefficient $\lambda = 0.001$, the learning rate to 0.001, the length of time period T to 1 (day), the number of time periods tr to 5, the slope in the LeakyRelu activation function to 0.2, weight coefficient γ to 0.4. The kernel scale K_1 and K_2 are set to 2 and 4, respectively. We employ a three-layer fully-connected neural network in the pairwise neural fusion block, and the number of neurons in each layer is 128, 64, 32, respectively. The threshold δ in the user-specific graph is set to 3.

Baseline algorithms. We compare our full approach with two statistical methods, four deep neural network based methods, and three variants of STDGAT.

- **Frequency-based matching** is a statistical method based on the query-POI interaction frequency. Specifically, given a query, we rank the POIs based on the query-POI co-occurrence (i.e., number of clicks of the POI candidate).
- **Distance-based matching** is another statistical method based on the distance between POI location and query location. Specifically, given a query, we rank the POI candidates based on their distance with the underlying query location.
- **DSSM** [9] is a widely used semantic matching model. A deep neural network is employed to predict the relevance between keywords and documents. In our experiments, for all DSSM based models, we treat POIs as documents and queries as keywords. We fine-tune all parameters based on settings in the original paper.
- **C-DSSM** [22] extends DSSM by adding extra convolutional-pooling layers to extract sentence-level features from n-gram word representations.
- **LSTM-DSSM** [20] incorporates LSTM [8] with DSSM to capture the temporal effect for semantic matching.
- **PALM** [36] proposes an attention-based neural network to incorporate semantic similarity and geographical correlation to quantify the query-POI relevance. Similarly, we fine-tune the parameters based on the default settings in original paper.
- **STDGAT-B** is the basic version of STDGAT, without considering the situational context factors and user attributes during the learning process.
- **STDGAT-St** is a variant of STDGAT without the user-specific query-POI graph attention block in learning and prediction.
- **STDGAT-Dy** is another variant of STDGAT, but it excludes the generic query-POI graph attention block.

Evaluation metrics. We adopt $Hits@k$ [30] and $NDCG@k$ [11] for evaluation. For $Hits@k$, it computes what percentage of POIs among the $top-k$ matched POIs based on queries has been clicked by a given user,

$$Hits@k = \frac{P_{u,q} \cap R_{u,q}(k)}{k}, \tag{24}$$

where $P_{u,q}$ is a set of clicked POIs based on query q for a user u, and $R_{u,q}(k)$ records the top-k matched POIs based on query q for user u. For $NDCG@k$, it takes both relevance score and the orders of all potential destination POIs into account and demonstrates the ranking quality of matching list,

$$NDCG@k = \frac{1}{IDCG} \sum_{i=1}^{M} \frac{2^{rel_i} - 1}{log(1+i)}, \tag{25}$$

where IDCG stands for the maximum possible DCG for a given POI recommendation list, and we set rel_i as 1 if the POI at position i is clicked and 0 otherwise. M denotes the number of correctly recommended POIs.

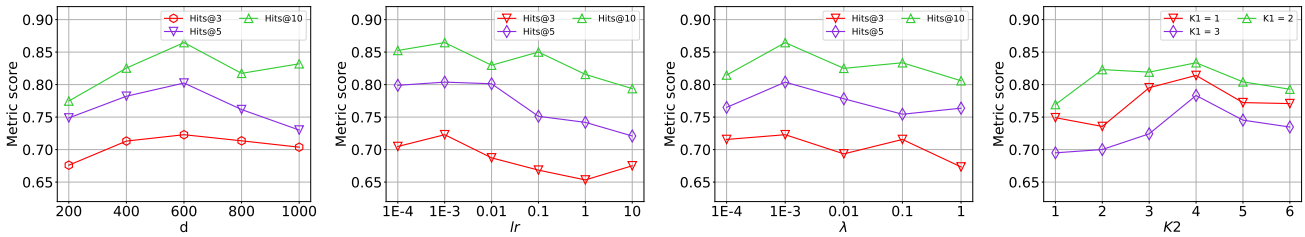
7.2 Overall performance

We evaluate the overall performance of our model as well as all baselines on Beijing and Shanghai Datasets. Specifically, we use $Hits@3$, $Hits@5$, $Hits@10$, $NDCG@3$, $NDCG@5$, $NDCG@10$.

All results are reported in Table 2. As can be seen, STDGAT achieves the best performance compared with all baselines on both datasets using all metrics (all the p-values between our model and each baseline are much smaller than 0.05, indicating the statistical significance of improvements). Specifically, we average the improvements on both datasets, and STDGAT outperforms the state-of-the-art baseline, PALM, by (0.0621, 0.0643, 0.0493, 0.0644, 0.0526, 0.0336) in terms of six metrics. In addition, we observe the advance of STDGAT reduces when we evaluate on a larger k. For example, STDGAT achieves 0.0486 improvement compared with PALM on Hit@3 on Beijing, whereas the improvement reduces to

Table 2: Overall performance.

Algorithm	Beijing							Shanghai						
	Hits@3	Hits@5	Hits@10	NDCG@3	NDCG@5	NDCG@10	p-value	Hits@3	Hits@5	Hits@10	NDCG@3	NDCG@5	NDCG@10	p-value
Frequency-based search	0.2938	0.4593	0.5809	0.2685	0.3893	0.4749	5.76e-18	0.2863	0.4474	0.5796	0.2732	0.4145	0.5193	6.19e-17
Distance-based search	0.2492	0.3670	0.4399	0.2283	0.3115	0.3669	1.76e-18	0.2294	0.2974	0.3659	0.2122	0.2658	0.3148	3.26e-20
DSSM	0.6016	0.6889	0.7337	0.5982	0.6687	0.7024	6.77e-8	0.6217	0.7039	0.7475	0.6169	0.6851	0.6928	9.43e-7
C-DSSM	0.6243	0.6910	0.7647	0.6134	0.6705	0.7293	4.02e-5	0.6384	0.7292	0.7789	0.6255	0.7090	0.7535	6.57e-7
LSTM-DSSM	0.6441	0.7311	0.7860	0.6317	0.7025	0.7436	3.23e-9	0.6145	0.7371	0.7883	0.6236	0.7114	0.7562	1.92e-8
PALM	0.6743	0.7382	0.8251	0.6685	0.7022	0.7653	5.57e-4	0.6588	0.7531	0.8046	0.6327	0.7295	0.7689	1.55e-9
STDGAT-B	0.5973	0.6691	0.7231	0.5772	0.6219	0.6689	8.62e-13	0.6153	0.6970	0.7332	0.5939	0.6668	0.6921	4.05e-12
STDGAT-St	0.6377	0.7186	0.7795	0.6211	0.6729	0.7157	4.61e-6	0.6422	0.7690	0.7709	0.6313	0.7249	0.7467	1.58e-6
STDGAT-Dy	0.6573	0.7524	0.8026	0.6397	0.7244	0.7590	2.76e-7	0.6854	0.7792	0.8125	0.6672	0.7482	0.7643	7.54e-5
STDGAT	**0.7229**	**0.8038**	**0.8646**	**0.7034**	**0.7733**	**0.8042**	–	**0.7343**	**0.8161**	**0.8537**	**0.7266**	**0.7635**	**0.7971**	–



(a) Dimension of latent vector d. **(b) Learning rate lr.** **(c) Regularization coefficient λ.** **(d) Kernel scale K_1, K_2.**

Figure 4: Parameter sensitivity on Beijing.

0.0395 on Hit@10. This makes sense because most of the correctly predicted POIs are placed in the front ranks, which demonstrates the high quality of the matching list. Moreover, we observe significant improvement by applying neural network based models on query-POI matching problem, which demonstrates the effectiveness of deep learning models. As Table 2 shows, the performances of both STDGAT-St and STDGAT-Dy are better than our basic model STDGAT-B, but STDGAT-St performs slightly worse than STDGAT-Dy, which indicates the user's unique search preference has more impacts than geographical correlations on the POI selection. Finally, our full approach STDGAT outperforms STDGAT-B, STDGAT-St, ADGA-Dy by (0.1223, 0.1269, 0.1310, 0.1295, 0.1241, 0.1202), (0.0887, 0.0662, 0.0840, 0.0889, 0.0695, 0.0694) and (0.0573, 0.0442, 0.0516, 0.0795, 0.0321, 0.0390) respectively in average. Indeed, the introduction of geographical information, time slot, and user preference does exert positive influences on query-POI matching.

7.3 Parameter sensitivity

Then we report the parameter sensitivity of STDGAT on Beijing dataset, including the representation vector dimension d, the learning rate lr, the regularization coefficient λ, and kernel scales K_1 and K_2. The results on Shanghai are similar, and we omit them due to space limit.

First, we vary the representation vector dimension d from 16 to 256. As shown in Figure 4(a), we observe a performance improvement when we increase d from 16 to 64 and performance degrade when we further increase d from 64 to 256. These results illustrate 64 dimension representation vector is powerful enough to capture semantic information.

Second, we vary the learning rate lr from 0.0001 to 10. As shown in Figure 4(b), the performance is relatively stable when lr is smaller than 0.01, and we observe consistent performance degradation when lr increases from 0.01 to 10, probably because large learning rate results in divergent weight update, which may oscillate the model performance.

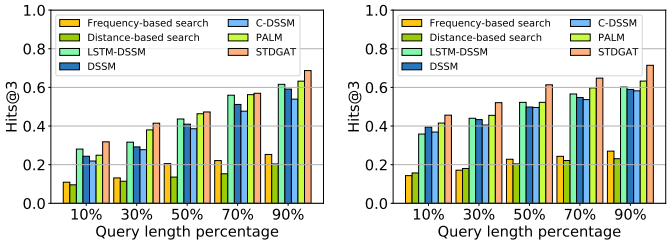
Third, we vary the regularization coefficient λ from 0.0001 to 1. The results are reported in Figure 4(c). As can be seen, the model achieves optimal performance when set $\lambda = 0.001$, and the performance degrades with λ is either too small or too large. One possible reason is that if the interference of the regularization term is negligible, it will make little contribution to the performance. In contrast, if too much attention is paid to regularization terms, the model performance will be underestimated.

Fourth, we evaluate the impact of scales K_1 and K_2. As shown in Figure 4(d), we vary K_1 from 1 to 3 and K_2 from 1 to 6. Overall, the performance reaches optimal when we set $K_1 = 2$ and $K_2 = 4$. We observe remarkable performance degradation when we increase or decrease K_1 and K_2.

7.4 Influence of query incompleteness

We further compare the effectiveness of STDGAT and existing baselines on handling incomplete queries. The average lengths of POI name in Beijing and Shanghai datasets are 10.23 and 10.36, whereas the average lengths of query keywords are only 5.12 and 5.03, respectively. Specifically, we partition each dataset into subsets according to the percentage of query completeness. The results on $Hits@3$ are shown in Figure 5.

As can be seen, STDGAT achieves the best performance compared with all baselines on all subsets of different percentages of query completeness. Specifically, the performance of STDGAT more than doubles two statistical methods (Frequency-based and Distance-based search) and greatly outperforms the results of DSSM-based models by (0.0851, 0.0726, 0.0840, 0.0821, 0.1069) for different query length percentages. Such results validate our expectation



(a) Beijing. (b) Shanghai.

Figure 5: Influence of query incompleteness.

that incorporating textual semantics, pairwise activity relevance, and time-evolving user preference does have positive influences on incomplete query-POI matching. Look into more details, the performances of Frequency-based and Distance-based search models remain low (between 0.0772 and 0.2709) mainly because traditional methods have become incompatible with query-based POI prediction when the query keyword is ambiguous. Similarly, while the family of deep structured semantic models (DSSM) has managed to generate a series of appropriate semantic representatives for latent semantics from a query-POI pair, the incomplete query physically blurs the previously obvious semantic patterns. Furthermore, DSSM-based models overlook factors from non-semantic domains such as user preference and situational context, thus the scores are relatively lower than PALM and STDGAT. In contrast, PALM preserves the semantic similarity and geographical correlation, therefore performs relatively better among other baselines.

7.5 Handling cold start problem

Finally, we evaluate the performance of STDGAT on handling the cold start problem. For three classes of cold start problems, we randomly remove 5% vertices (i.e., users, queries, and POIs, respectively) and their corresponding map query events from the Beijing dataset. The results of *Hits*@3 on removed records are reported in Table 3. As can be seen, the performances on records with cold start users, cold start queries, and cold start POIs are marginally worse than those on records with existing users, queries, and POIs. However, the performance of STDGAT on cold start records is still significantly better than all baselines. We observe the influence of unseen users is the largest, which is because the user-specific query-POI graph attention is not applicable. On the other hand, however, the above observations also validate the effectiveness of the user-specific query-POI graph attention.

Table 3: *Hits*@3 performance on handling cold-start problems on Beijing.

Algorithm	New user	New POI	New query	p-value
Frequency-based search	0.3143	0.3414	0.3355	3.26e-14
Distance-based search	0.2857	0.3043	0.3158	1.23e-16
DSSM	0.3389	0.3576	0.4259	5.02e-8
C-DSSM	0.3537	0.3820	0.4097	3.41e-8
LSTM-DSSM	0.3807	0.4265	0.4465	2.85e-7
PALM	0.3902	0.3899	0.4635	1.78e-5
STGDAT	**0.4023**	**0.4347**	**0.4728**	–

8 RELATED WORK

Deep learning based semantic matching. In recent years, deep learning has demonstrated its effectiveness in learning higher-order features for various information retrieval tasks [7, 9, 17, 20, 22, 36]. Among them, the DSSM [9], its extensions (C-DSSM [22], LSTM-DSSM [20] and latest PALM are the most related to our work. DSSM uses a deep neural network architecture to map a bag of letter-trigrams from search queries and documents to low-dimensional semantic embeddings. The conditional likelihood of clicks among query-item pairs is computed as the cosine similarity of their corresponding embeddings. However, bag-of-words representations cannot keep the contextual structure and long-term contextual influence within the query or documents. Subsequently, C-DSSM and LSTM-DSSM are proposed to bridge the research gap accordingly. In order to compensate the limitations brought by one-sided semantic source, PALM introduces external geographical information with semantic similarity for measuring query-POI relevance. It uses pre-generated word embeddings to present four types of variables (POI name, POI address, geographical location and query word), and stacks multiple convolutional and self-attention layers to capture query-POI correlations for matching. Compared with our approach, the above approaches have two major limitations: (1) they only capture static representations and structures of queries and items, therefore take the risk of losing important situational context for query-POI matching. (2) they ignore the effects of time-evolving user preference in the query-POI relevance learning.

Graph neural network. GNN has shown its power on modeling non-Euclidean graph structures [12]. Specifically, GCN [5] learns node representations by considering their neighbor nodes through a predefined aggregation function. GAT [24] only explicitly sets the adjacent matrix and employs the attention mechanism to learn the edge weights automatically. Recently, GNN based models have been proposed to tackle various problems, such as social recommendation [6], user behaviour modeling [37], and spatiotemporal forecasting [35]. For relevance matching, Zhang et al. [34] adopt GAT to capture structural information for both query and documents to obtain better representations, Wang et al. [26] employ GCN for cross-modal language-to-vision matching. Different from the above works, to the best of our knowledge, we first apply GAT to the query-POI matching problem.

Dual paradigm. In real-world life, the essence of the dual paradigm inspires the dual structural design in many tasks. For example, DELF [2] learns static embeddings for both users and items in recommendation systems. DANSER [29] proposes a dual graph attention network to model social effects in recommendation tasks. DGCN [40] uses a dual structural to ensure the global and local consistency in the semi-supervised learning of graph-structured data. MDAL [28] utilizes the dual effects of label and content information for structural knowledge learning. In this paper, we follow the dual paradigm and propose a novel dual GAT architecture to jointly learn the generic and user-specific correlations between queries and POIs under complex situational contexts.

9 CONCLUSION

In this paper, we proposed STDGAT, user preference and situational context aware intelligent query-POI matching framework.

Our contributions lie in four aspects. First, we collaboratively modeled different textual information (i.e., queries and POI names) and situational contexts (i.e., geographical location, time information, and user) when quantifying the query-POI similarity. Second, we investigated the inherent interactions among dynamic spatio-temporal factors and proposed a dual graph attention network to capture query-POI relevance from both generic perspective and user-specific perspective. Specifically, the generic graph attention captures global query-POI correlation, while the user-specific graph attention captures the time-evolving user preference on destination POIs. Third, we addressed the cold start problem in query-POI matching task, and introduced several training techniques to improve matching effectiveness and training efficiency. Finally, we conducted extensive experiments on two real-world map search query datasets to evaluate the model, and the experimental results demonstrate that the performance of STDGAT significantly outperforms six baselines.

REFERENCES

[1] Jacob Benesty, Jingdong Chen, Yiteng Huang, and Israel Cohen. 2009. Pearson correlation coefficient. In *Noise reduction in speech processing*. Springer, 1–4.

[2] Weiyu Cheng, Yanyan Shen, Yanmin Zhu, and Linpeng Huang. 2018. DELF: A Dual-Embedding based Deep Latent Factor Model for Recommendation.. In *IJCAI*. 3329–3335.

[3] Aaron Clauset, Cosma Rohilla Shalizi, and Mark EJ Newman. 2009. Power-law distributions in empirical data. *SIAM review* (2009), 661–703.

[4] Zeyu Cui, Zekun Li, Shu Wu, Xiao-Yu Zhang, and Liang Wang. 2019. Dressing as a Whole: Outfit Compatibility Learning Based on Node-wise Graph Neural Networks. In *The World Wide Web Conference*. ACM, 307–317.

[5] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in neural information processing systems*. 3844–3852.

[6] Wenqi Fan, Yao Ma, Qing Li, Yuan He, Eric Zhao, Jiliang Tang, and Dawei Yin. 2019. Graph Neural Networks for Social Recommendation. In *The World Wide Web Conference*. ACM, 417–426.

[7] Geoffrey Hinton, Li Deng, Dong Yu, George Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Brian Kingsbury, et al. 2012. Deep neural networks for acoustic modeling in speech recognition. *IEEE Signal processing magazine* (2012).

[8] Sepp Hochreiter and Jürgen Schmidhuber. 1997. LSTM can solve hard long time lag problems. In *Advances in neural information processing systems*. 473–479.

[9] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*. ACM, 2333–2338.

[10] Priit Järv. 2019. Predictability limits in session-based next item recommendation. In *Proceedings of the 13th ACM Conference on Recommender Systems*. 146–150.

[11] Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems (TOIS)* (2002), 422–446.

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

[13] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*. 1097–1105.

[14] Chaolong Li, Zhen Cui, Wenming Zheng, Chunyan Xu, and Jian Yang. 2018. Spatio-temporal graph convolution for skeleton based action recognition. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

[15] Hao Liu, Ting Li, Renjun Hu, Yanjie Fu, Jingjing Gu, and Hui Xiong. 2019. Joint Representation Learning for Multi-Modal Transportation Recommendation. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence*. 1036–1043.

[16] Hao Liu, Yongxin Tong, Panpan Zhang, Xinjiang Lu, Jianguo Duan, and Hui Xiong. 2019. Hydra: A personalized and context-aware multi-modal transportation recommendation system. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2314–2324.

[17] Grégoire Mesnil, Xiaodong He, Li Deng, and Yoshua Bengio. 2013. Investigation of recurrent-neural-network architectures and learning methods for spoken language understanding.. In *Interspeech*. 3771–3775.

[18] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[19] Kyosuke Nishida, Hiroyuki Toda, Takeshi Kurashima, and Yoshihiko Suhara. 2014. Probabilistic identification of visited point-of-interest for personalized automatic check-in. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. ACM, 631–642.

[20] Hamid Palangi, Li Deng, Yelong Shen, Jianfeng Gao, Xiaodong He, Jianshu Chen, Xinying Song, and R Ward. 2014. Semantic modelling with long-short-term memory for information retrieval. *arXiv preprint arXiv:1412.6629* (2014).

[21] Tatjana Scheffler, Rafael Schirru, and Paul Lehmann. 2012. Matching points of interest from different social networking sites. In *Annual Conference on Artificial Intelligence*. Springer, 245–248.

[22] Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. 2014. A latent semantic model with convolutional-pooling structure for information retrieval. In *Proceedings of the 23rd ACM international conference on conference on information and knowledge management*. ACM, 101–110.

[23] Gokhan Tur, Li Deng, Dilek Hakkani-Tür, and Xiaodong He. 2012. Towards deeper understanding: Deep convex networks for semantic utterance classification. In *2012 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 5045–5048.

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

[25] Pengfei Wang, Yanjie Fu, Guannan Liu, Wenqing Hu, and Charu Aggarwal. 2017. Human mobility synchronization and trip purpose detection with mixture of hawkes processes. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 495–503.

[26] Peng Wang, Qi Wu, Jiewei Cao, Chunhua Shen, Lianli Gao, and Anton van den Hengel. 2019. Neighbourhood watch: Referring expression comprehension via language-guided graph attention networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1960–1968.

[27] Peter Whitle. 1951. *Hypothesis testing in time series analysis*. Almqvist & Wiksells.

[28] Longcan Wu, Daling Wang, Shi Feng, Yifei Zhang, and Ge Yu. 2019. MDAL: Multi-task Dual Attention LSTM Model for Semi-supervised Network Embedding. In *International Conference on Database Systems for Advanced Applications*. Springer, 468–483.

[29] Qitian Wu, Hengrui Zhang, Xiaofeng Gao, Peng He, Paul Weng, Han Gao, and Guihai Chen. 2019. Dual Graph Attention Networks for Deep Latent Representation of Multifaceted Social Effects in Recommender Systems. In *The World Wide Web Conference*. ACM, 2091–2102.

[30] Xiwang Yang, Harald Steck, Yang Guo, and Yong Liu. 2012. On top-k recommendation using social networks. In *Proceedings of the sixth ACM conference on Recommender systems*. ACM, 67–74.

[31] Hongzhi Yin, Xiaofang Zhou, Bin Cui, Hao Wang, Kai Zheng, and Quoc Viet Hung Nguyen. 2016. Adapting to user interest drift for poi recommendation. *IEEE Transactions on Knowledge and Data Engineering* (2016), 2566–2581.

[32] Yanwei Yu, Hongjian Wang, and Zhenhui Li. 2018. Inferring Mobility Relationship via Graph Embedding. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* (2018), 147.

[33] Quan Yuan, Gao Cong, and Aixin Sun. 2014. Graph-based point-of-interest recommendation with geographical and temporal influences. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*. ACM, 659–668.

[34] Ting Zhang, Bang Liu, Di Niu, Kunfeng Lai, and Yu Xu. 2018. Multiresolution Graph Attention Networks for Relevance Matching. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. ACM, 933–942.

[35] Weijia Zhang, Hao Liu, Yanchi Liu, Jingbo Zhou, and Hui Xiong. 2020. Semi-Supervised Hierarchical Recurrent Graph Neural Network for City-Wide Parking Availability Prediction. *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence* (2020).

[36] Ji Zhao, Dan Peng, Chuhan Wu, Huan Chen, Meiyu Yu, Wanji Zheng, Li Ma, Hua Chai, Jieping Ye, and Xiaohu Qie. 2019. Incorporating Semantic Similarity with Geographic Correlation for Query-POI Relevance Learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*. 1270–1277.

[37] Chang Zhou, Jinze Bai, Junshuai Song, Xiaofei Liu, Zhengchao Zhao, Xiusi Chen, and Jun Gao. 2018. ATRank: An attention-based user behavior modeling framework for recommendation. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

[38] Jingbo Zhou, Shan Gou, Renjun Hu, Dongxiang Zhang, Jin Xu, Xuehui Wu, Airong Jiang, and Hui Xiong. 2019. A Collaborative Learning Framework to Tag Refinement for Points of Interest. In *KDD*. 1752–1761.

[39] Jingbo Zhou, Hongbin Pei, and Haishan Wu. 2018. Early warning of human crowds based on query data from Baidu maps: Analysis based on Shanghai stampede. In *Big data support of urban planning and management*. Springer, 19–41.

[40] Chenyi Zhuang and Qiang Ma. 2018. Dual graph convolutional networks for graph-based semi-supervised classification. In *Proceedings of the 2018 World Wide Web Conference*. International World Wide Web Conferences Steering Committee, 499–508.