Negative Sampling in Next-POI Recommendations: Observation, Approach, and Evaluation

Hong-Kyun Bae*
hongkyun@hanyang.ac.kr
Hanyang University
Seoul, South Korea

Yebeen Kim*
kyeb98@hanyang.ac.kr
Hanyang University
Seoul, South Korea

Hyunjoon Kim†
hyunjoonkim@hanyang.ac.kr
Hanyang University
Seoul, South Korea

Sang-Wook Kim†
wook@hanyang.ac.kr
Hanyang University
Seoul, South Korea

ABSTRACT

To recommend the *points of interest* (POIs) that a user would *check-in next*, most *deep-learning* (DL)-based existing studies have employed *random negative* (RN) sampling during model training. In this paper, we claim and validate that, as the training proceeds, such an RN sampling in reality performs as sampling *easy negative* (EN) POIs (*i.e.*, *EN sampling*) that a user was *highly unlikely* to check-in at her check-in time point. Furthermore, we verify that EN sampling is more *disadvantageous* in improving the accuracy than sampling *hard negative* (HN) POIs (*i.e.*, *HN sampling*) that a user was *highly likely* to check-in. To address this limitation, we present the novel concept of the *Degree of Positiveness* (DoP), which can be formulated by two factors: (i) the degree to which a POI has the *characteristics preferred* by a user; (ii) the *geographical distance* between a user and a POI. Then, we propose a new *model-training scheme based on HN sampling* by using DoP. Using real-world datasets (*i.e.*, NYC, TKY, and Brightkite), we demonstrate that all the state-of-the-art models trained by our scheme showed dramatic improvements in accuracy by up to about 82.8%.

CCS CONCEPTS

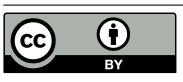
• **Information systems** → **Recommender systems**.

KEYWORDS

Next-POI recommender systems, hard negative sampling, dynamic negative sampling

ACM Reference Format:
Hong-Kyun Bae, Yebeen Kim, Hyunjoon Kim, and Sang-Wook Kim. 2024. Negative Sampling in Next-POI Recommendations: Observation, Approach, and Evaluation. In *Proceedings of the ACM Web Conference 2024 (WWW '24), May 13–17, 2024, Singapore, Singapore.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3589334.3645681

*Co-first authors
†Co-corresponding authors



WWW '24, May 13–17, 2024, Singapore, Singapore

ACM ISBN 979-8-4007-0171-9/24/05.
https://doi.org/10.1145/3589334.3645681

1 INTRODUCTION

Recently, *location-based social network services* (LBSNs) (*e.g.*, Brightkite, Foursquare) have been widely used all over the world [6, 36]. On LBSN platforms, users *check-in points-of-interest* (POIs), *i.e.*, locations or places that they have visited. The vast amounts of such check-in records obtained from users triggered the intensive research on *next-POI recommender systems* [13, 33].

A user's *POI check-in sequence* indicates the *chronological order* for POIs that the user has checked-in along with their *check-in time points*. Next-POI recommendations are based on the intuition that recommender models trained by a user's POI check-in sequence will be able to infer her check-in pattern, thereby finding the POIs that the user would visit next time (*i.e.*, next-POIs).

With the growth of *deep-learning* (DL) technology, models that can learn user preferences for items from sequences of users' feedback effectively, such as RNN [27], LSTM [10], and Transformer [31], have been mainly employed for next-POI recommendations [22, 24]. DL-based next-POI recommender systems train models by determining a user's positive/negative POIs at every check-in time point of her POI check-in sequence. The POI that she checked-in at that point is regarded as *positive*, while the (remaining) POIs that she did not check-in are regarded as *negative*. We note that there are many POIs not checked-in by a user at that time; this makes a lot of room for design choices on which POIs should be selected as negative POIs for the positive POI. Since this design choice can significantly influence the recommendation accuracy, extensive research has been conducted on *negative sampling* in various recommendation domains such as OTT and e-commerce [12, 18, 26, 28, 37, 39, 40]. However, in the domain of next-POI recommendations, most approaches simply employ the *random sampling* on non-visited POIs for each user during model training [5, 23, 25, 34, 38]. In this paper, we aim to conduct the first *exhaustive and comprehensive study on negative sampling in the next-POI domain.*

First, we categorize the negative POIs into three groups: (i) POIs that a user was *highly likely* to check-in at that point but did not; (ii) POIs that a user was *highly unlikely* to check-in at that point; and (iii) POIs that belong to neither group (i) nor group (ii). POIs in group (i) are likely to have *characteristics* that she prefers at that point, making it *difficult* for the model to distinguish them from her corresponding positive POI at that point. In this sense, the POIs in group (i) can be regarded as *hard negative* (HN) POIs of the user at that point. In contrast, the POIs in group (ii) can be regarded as

easy negative (EN) POIs since they are *easily* distinguished from the positive POI.

Random negative (RN) sampling employed in existing studies [5, 23, 25, 34, 38] is to sample negative POIs without distinguishing between HN POIs and EN POIs. In this paper, however, we claim that sampling HN POIs rather than EN POIs as negative POIs for the corresponding positive POI, *i.e.*, *HN sampling*, can be more effective in improving the accuracy of next-POI recommendations. By HN sampling, the models will be trained toward capturing precisely the characteristics of POIs that the user prefers more, which makes the ranking of the positive POI be predicted correctly (*i.e.*, higher than any other HN POIs of the user). To verify this claim in Section 2, we empirically investigate how the ranking of the positive POI predicted by the model can be changed depending on whether it is learned together with the user's HN POIs or EN POIs. Furthermore, by demonstrating that RN sampling behaves in reality as *EN sampling* as model training progresses, we validate why RN sampling will be less effective than HN sampling.

To address these limitations, we propose a new *model-training scheme based on HN sampling for the next-POI recommendation*. To this end, we define the *Degree of Positiveness* (DoP) that a user has for a POI as the degree to which the user is likely to check-in the POI at a time point. The following two factors can determine a user's DoP for a POI at the time point: (i) the degree to which a POI has the *characteristics preferred* by the user, based on her previous check-in sequence up to the given point; (ii) the *geographical distance* between a user and a POI at the point, which is crucial in the next-POI recommendation. With these two factors, we formulate DoP, thereby determining the negative POIs with *high DoPs* as the HN POIs of the user. In detail, our proposed model-training scheme based on DoP is designed in two steps below.

Step 1. Filtering POIs by the preferred characteristics. Basically, the HN items in the general recommendation domain indicate the items *positioned very close* to the positive item in the *latent feature space* [26, 39]. In the next-POI recommendation, the distance of a negative POI from the positive POI in the latent space will likely become larger as the negative POI has fewer characteristics preferred by a user based on her previous check-in sequence. Therefore, we *filter out* POIs with a *low degree* of having the preferred characteristics, *i.e.*, POIs *positioned far away* from the positive POI in the latent feature space, from the candidates for HN POIs.

Step 2. Sampling HN POIs via DoP. Unlike other recommendation domains such as OTT and e-commerce where considering only the distance in the latent feature space is *sufficient* to find the HN items, we should consider the *geographical distance* between POIs and a user as well in the next-POI domain; POIs *geographically located far away* from the user would be unlikely to be checked-in by her at that time point. Therefore, by Step 2, we aim to find the n HN POIs among the POIs obtained by Step 1 by considering not only the degree of having the characteristics preferred by her but also the *geographical closeness* to the user at that point. As a result, we can effectively identify the user's HN POIs that are located close to the user's positive POI, *both in the latent feature space and in the geographical space*, through the concept of DoP.

Along with the n HN POIs and the corresponding positive POI, we train a recommender model for the check-in point. With the model trained in this way for every user's all check-in points, we recommend the *next-POIs* that the user would check-in. We note that any existing models for the next-POI recommendation (*e.g.*, PLSPL [34], STAN [25]) can be incorporated with our proposed scheme, *i.e.*, *our scheme is model-agnostic*. In Section 4, we empirically show that state-of-the-art models benefit *significantly* with respect to accuracy from being trained by our scheme.

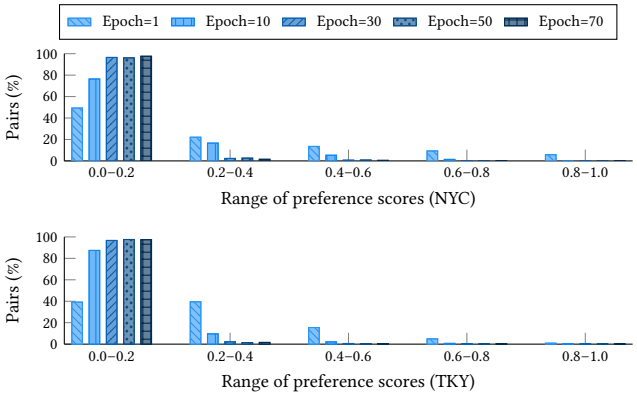


Figure 1: Percentage of user-POI pairs according to the range of preference scores predicted by GeoSAN for each epoch.

To the best of our knowledge, this is the first work to construct an *in-depth* study for the negative sampling issue in the next-POI recommendation that considers domain characteristics carefully. Our proposed scheme *differs* from the previous HN sampling methods proposed for the typical recommendation domains (*e.g.*, OTT, e-commerce) [26, 28, 37, 39, 40] in the following ways:

- We find the HN POIs by using not only the *distance between POIs in the latent feature space* but also their *geographical distance*.
- Rather than employing a *set of items* a user has used for HN sampling as in typical recommendation domains, we consider a *sequence of POIs* a user has checked-in to infer user preferences in the next-POI recommendation.

Our work contributes to providing an important insight that HN sampling can be more effective than RN sampling for *sequence-based recommender systems*.

2 MOTIVATION

In this section, first, we examine RN sampling methods employed in existing studies for the next-POI recommendation and highlight that they are close to EN sampling. Then, we show their limitations in terms of ranking prediction.

2.1 Negative sampling in existing studies

Negative POIs, which have not been checked-in by a user at a time point, can be grouped as follows, depending on *how probable* she would have checked-in at that point: (i) *hard negative* (HN) POIs with *high* probability; (ii) *easy negative* (EN) POIs with *low* probability; and (iii) the others. However, most existing studies for the next-POI recommendation have *not considered* these various types of negative POIs in model training. CatDM [38], STAN [25], and STKGRec [5] *randomly sample* negative POIs among *all* POIs that were not checked-in by a user. In addition, GeoSAN [23] and

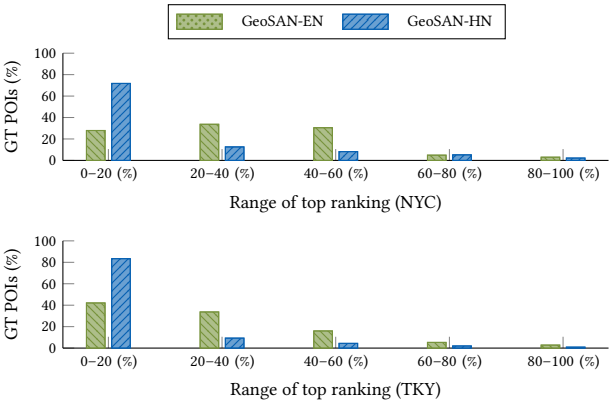


Figure 2: Percentage of ground truth (GT) POIs according to the range of top ranking predicted by GeoSAN-EN/HN.

TGSTAN [3] *randomly* sample negative POIs among the POIs that were (i) *geographically close* to a user but (ii) were not checked-in.

These models are trained in such a way that negative POIs are predicted to have a low preference score. Note that, as training proceeds (*i.e.*, the number of epochs increases), more non-checked-in POIs are sampled as negative POIs. That is, the number of POIs predicted to have a *low preference score* by the model (*i.e.*, the POIs regarded as *EN POIs*) gradually increases as training proceeds. Thus, we claim that RN sampling employed in existing studies performs similarly as *EN sampling* does as training proceeds.

To verify this claim, we empirically investigate how the respective ratio of HN/EN POIs among the negative POIs of a user varies as the model is trained, by using two real-world datasets (*i.e.*, NYC and TKY) [36]. First, for each user, we select M POIs that she has not checked-in at each time point in the sequence.[1] Then, we predict her preference scores for these M POIs from 0 to 1 by the recommender model (which is being trained). We repeat this process in training the model for a total of 70 epochs, and we split all pairs of user-POI into five groups based on their preference scores (*i.e.*, 0.0–0.2, 0.2–0.4, ... and 0.8–1.0). As models, we employ GeoSAN [23] and CatDM [38], which are state-of-the-art methods for the next-POI recommendation.

Figure 1 shows the results for GeoSAN, where the x-axis denotes the range of preference scores predicted by the model and the y-axis denotes the percentage of user-POI pairs corresponding to each range.[2] The POIs belonging to the range of 0.0–0.2 and the range of 0.8–1.0 can be regarded as EN POIs and HN POIs of the user, respectively. From the middle phase of the training (*i.e.*, Epoch=30), we observe that most pairs (more than 95% for both datasets) belong to the range of 0.0–0.2; if a negative POI is randomly sampled for a user, it is *highly likely* to be an *EN POI* of that user. In this sense, the model trained with RN sampling is equivalent (in practice) to the model trained with *EN sampling* as the training progresses. In the following subsection, we will exhibit the limitations of the training scheme with this EN sampling.

[1]Following [23], we sample M negative POIs that were *geographically close* to a user at each time point (M=2,000).
[2]For the results for CatDM, please refer to Appendix C.1.

2.2 Limitations of EN sampling

While employing EN sampling, the model is trained to predict a ranking for positive POIs higher than that for EN POIs. Here, note that the rankings of EN POIs are at the *bottom* among the total negative POIs for that user at that time point. Therefore, the ranking of the positive POI, which is predicted to be *only higher than that of EN POIs*, is likely to be located around the *middle*. That is, there may still remain many negative POIs with a *higher ranking than that of the positive POI*, which indicates that the model is being trained in the *incorrect way*.

To validate this claim empirically, we present the difference in the predicted rankings for positive POIs for the two models trained by HN sampling and EN sampling, respectively, by using two real-world datasets (*i.e.*, NYC and TKY). We employ GeoSAN [23] as the model and refer to GeoSAN-HN and GeoSAN-EN as the models trained by HN sampling and EN sampling, respectively. GeoSAN-HN and GeoSAN-EN are trained by sampling the top-10 and bottom-10 POIs as negative POIs, respectively, based on the predicted preference scores among the total non-checked-in POIs at each time point. Then, through each trained model, we predict the preference score for the POI that a user checked-in *last* in the sequence (*i.e.*, *ground truth*, GT) and the preference scores for all negative POIs at that time point. After sorting all these POIs in descending order by the predicted scores, we divide GT POIs into five ranges according to their rankings among all POIs, *i.e.*, top 0–20%, ..., and 80–100%.

Figure 2 shows the results, where the x-axis represents the ranking range and the y-axis represents the percentage of GT POIs corresponding to each range. For the NYC dataset, we observe that more than 70% of GT POIs belong to the range of top 0–20% based on the rankings predicted by GeoSAN-HN; the rankings for most GT POIs are *correctly* predicted to be at the top among all POIs. On the other hand, by GeoSAN-EN, only about 30% of GT POIs belong to the range of top 0–20%; the rankings for the remaining (*i.e.*, around 70% of) GT POIs are *incorrectly* predicted to be around or below the middle, which can lead to *degraded* recommendation accuracy. To address this limitation, we propose a novel training scheme for next-POI recommendation models based on HN sampling.

3 PROPOSED APPROACH

In this section, we first define the *Degree of Positiveness* (DoP) that indicates the degree to which a user is likely to check-in a POI at a time point. With the concept of DoP, we present our model-training scheme based on HN sampling for the next-POI recommendation.[3]

3.1 Degree of Positiveness (DoP)

For a user u at a given time point t, the following two factors decide DoP for a POI v.

- $c(u, v, t)$: the degree to which POI v has the *characteristics* preferred by user u, given the previous check-in sequence of u up to the time point $(t-1)$.
- $d(u, v, t)$: the geographical distance between user u and POI v at the time point t.

A positive POI that u has checked-in at t can be regarded as the POI with the highest DoP for u at t. In this sense, HN POIs of u at t

[3]For the notations used in this paper, please refer to Appendix B.

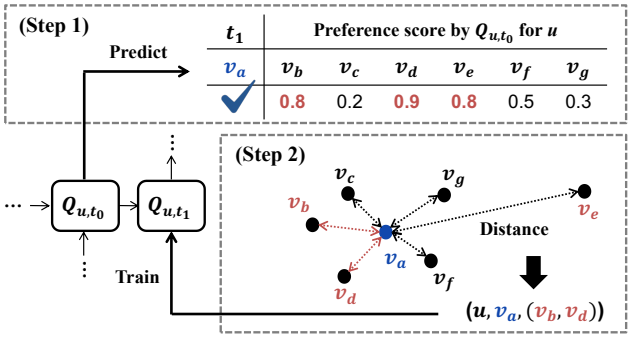


Figure 3: Overview of our proposed model-training scheme.

are regarded as the POIs *not-checked-in* by u at t but with a fairly *high DoP* comparable to that of the positive POI.[4]

There can be three strategies to give *priority between the two factors for DoP* in finding the HN POIs of each user: (i) prioritizing $c(u, v, t)$ over $d(u, v, t)$; (ii) prioritizing $d(u, v, t)$ over $c(u, v, t)$; and (iii) giving equal priority to both of them. Among these strategies, we adopt strategy (i): (Step 1) we first decide the candidates for HN POIs based only on $c(u, v, t)$, and then (Step 2) we sample HN POIs by using a *fine-grained preference score* adjusted by both $c(u, v, t)$ and $d(u, v, t)$. We empirically demonstrate that the HN POIs obtained from this strategy significantly help a model *converge to higher accuracy* in early- or mid-stages of training compared to the other two strategies in Section 4.

The overall procedure of our model-training scheme based on the HN sampling is illustrated in Figure 3. We explain the details of Steps 1 and 2 in Figure 3 in subsections 3.2 and 3.3, respectively.

3.2 Filtering POIs (Step 1)

As shown in Figure 1, HN POIs typically account for only a *very small portion* ($\leq 3\%$) during model training. In order to select such a small number of HN POIs precisely, we aim to filter out the negative POIs unlikely to become the HN POIs of u at t among the total negative POIs in Step 1.

The HN POIs are *not easily distinguished* from the positive POI of u at t by the model during the training process. In other words, the HN POIs can be regarded as the POIs *positioned very close* to the positive POI in the *latent feature space*. The degree of having characteristics preferred by a user on a POI is *highly associated* with the distance of the positive POI and that POI in the latent feature space; the POIs with a high/low degree of having the characteristics preferred by u are positioned close/far from her positive POI, respectively, in the latent feature space. For this reason, the POIs with a *low degree of having the characteristics* preferred by u are *unlikely* to become the HN POIs of u at t (since the model will predict preference scores for them *much lower* than that of the positive POI of u at t). To filter our such POIs, we consider $c(u, \bar{v}, t)$ and leverage the model trained by the u's check-in sequence up to $(t-1)$ (*i.e.*, the model that is being trained for the next-POI recommendation) to determine $c(u, \bar{v}, t)$.

[4]Following the existing studies [5, 38], we exclude the POIs that have been checked-in by u before t from the negative POIs at t.

Here, any existing models that can infer the user's check-in pattern from the check-in sequence (*e.g.*, GeoSAN [23], STAN [25]) can be employed. We formulate $c(u, \bar{v}, t)$ with the preference score of u for $\bar{v}$ predicted by the model as follows:

$$c(u, \bar{v}, t) = Q_{u,(t-1)}(\bar{v}), \tag{1}$$

where $Q_{u,(t-1)}$ denotes the model trained by the check-in sequence of u up to $(t-1)$; $Q_{u,(t-1)}(\bar{v})$ denotes the preference score of u for $\bar{v}$ predicted by model $Q_{u,(t-1)}$ $(0 \leq Q_{u,(t-1)}(\bar{v}) \leq 1)$.

Then, we sort the negative POIs of u at t in descending order of $c(u, \bar{v}, t)$ and filter out bottom-ranked POIs. The remaining negative POIs are defined as the set $C_{u,t}(m)$ of candidates for the HN POIs:

$$C_{u,t}(m) = \{\bar{v} | rank(c(u, \bar{v}, t)) \leq m\}, \bar{v} \in \bar{V}_{u,t}, \tag{2}$$

where $\bar{V}_{u,t}$ represents the set of all POIs that u has not checked-in at t; $rank(c(u, \bar{v}, t))$ represents the ranking of $\bar{v}$ among the POIs in $\bar{V}_{u,t}$ after sorting; m represents the size of $C_{u,t}(m)$. In other words, by Step 1, we select the top-m POIs with the highest $c(u, \bar{v}, t)$ predicted by model $Q_{u,(t-1)}$ among the POIs in $\bar{V}_{u,t}$, and use them as candidates for the HN POIs of u at t.[5]

For example, in Step 1 of Figure 3, the POI v_a checked-in by a user u is regarded as the positive POI at the time point t_1. Given m=3, among the POIs not checked-in by u (*i.e.*, $v_b \sim v_g$), the top-3 POIs with the highest preference scores predicted by model Q_{u,t_0} are determined as the candidates for the HN POIs at t_1 (*i.e.*, $C_{u,t_1}(3) = \{v_b, v_d, v_e\}$).

3.3 Sampling HN POIs via DoP (Step 2)

Unlike other recommendation domains, such as OTT and e-commerce, where considering only the distance in the latent feature space is *sufficient* to find the HN items [26, 28, 39], we claim that the *geographical distance* should be also considered to find the HN POIs in the next-POI domain; if a negative POI $\bar{v}$ in $C_{u,t}(m)$ with a *high degree* of having the characteristics preferred u at t is geographically located *far away* from u, then u is unlikely to check-in $\bar{v}$ at t.

Using real-world datasets, we conducted a statistical analysis to verify the relationship between a user's check-in probability for a POI and the distance between the POI and the user. Specifically, we calculated the geographical distance between two (positive) POIs that u has *successively* checked-in on her check-in sequence. Then, we obtained the distribution of a distance between every pair of successive POIs. In Figure 4 showing the results for the NYC dataset,[6] we can observe that about 70% of all pairs belong to the range of a distance less than 5km.

Thus, in Step 2, we first obtain the distance between each POI $\bar{v} \in C_{u,t}(m)$ and u at t. Specifically, we formulate the distance $d(u, \bar{v}, t)$ as follows, while regarding the location of the positive POI that u has checked-in at t as the location of u at t:

$$d(u, \bar{v}, t) = Haversine(v_{u,t}, \bar{v}), \bar{v} \in C_{u,t}(m), \tag{3}$$

where $Haversine(\cdot)$ denotes the *Haversine formula* that determines the shortest distance between two locations by using their latitudes and longitudes [30], which has been widely employed in existing

[5]To prevent $C_{u,t}(m)$ from always consisting of the same (top-m) POIs, we construct $\bar{V}_{u,t}$ of M POIs randomly sampled from all negative POIs of u at t (*e.g.*, M=2,000), following the previous studies for HN sampling in other domains [28, 39].
[6]For the results of using another dataset (*i.e.*, TKY), please refer to Appendix C.2.

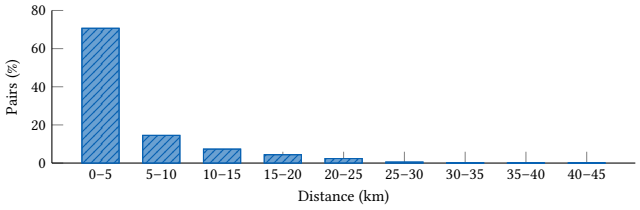


Figure 4: Distribution of pairs of successive POIs over the distance between them (NYC).

studies for the next-POI recommendation [11, 32]; $v_{u,t}$ denotes the positive POI of u at t. Then, we perform the *min-max scaling* on the values of $d(u, \bar{v}, t)$ obtained for every POI $\bar{v} \in C_{u,t}(m)$: $d(u, \bar{v}, t)$ of $\bar{v}$ located farthest (resp. closest) from u becomes 1 (resp. 0) (*i.e.*, $0 \leq d(u, \bar{v}, t) \leq 1$).

Then, we aim to determine the HN POIs among the POIs $\in C_{u,t}(m)$ by considering not only the degree of having the characteristics preferred by u but also the geographical closeness to u at t. Consequently, we formulate DoP of u for $\bar{v}$ at t by two factors (*i.e.*, $c(u, \bar{v}, t)$ and $d(u, \bar{v}, t)$) as follows:

$$DoP(u, \bar{v}, t) = (1 - d(u, \bar{v}, t)) \cdot c(u, \bar{v}, t), \tag{4}$$

where $DoP(u, \bar{v}, t)$ can be regarded as a *fine-grained preference score adjusted* by both $c(u, \bar{v}, t)$ and $d(u, \bar{v}, t)$. In other words, $DoP(u, \bar{v}, t)$ increases as $c(u, \bar{v}, t)$ increases or $d(u, \bar{v}, t)$ decreases. Note that *multiplication* adopted in Eq. (4) has been widely used in other studies for recommender systems [1, 2, 7]. [7]

Finally, we sample the top-n HN POIs with the highest $DoP(u, \bar{v}, t)$ among the m candidate POIs (*i.e.*, $C_{u,t}(m)$). We define the set $H_{u,t}(n)$ of these top-n HN POIs of u at t as follows:

$$H_{u,t}(n) = \{\bar{v} | rank(DoP(u, \bar{v}, t)) \leq n\}, \tag{5}$$

where n represents the number of HN POIs to sample. In other words, our proposed concept of DoP aids in effectively sampling u's HN POIs which are located close to u's positive POI *both in the latent feature space and in the geographical space.*

In Step 2 of Figure 3, while the preference scores predicted by Q_{u,t_0} for v_b, v_d, and v_e are similar to each other, v_e is located farther from v_a (*i.e.*, location of u) than the other two POIs. That is, $DoP(u, v_e, t_1)$ is smaller than $DoP(u, v_b, t_1)$ and $DoP(u, v_d, t_1)$. For this reason, given n=2, we can finally sample v_b and v_d as the HN POIs of u at t_1 (*i.e.*, $H_{u,t_1}(2) = \{v_b, v_d\}$).

3.4 Training a next-POI recommender model

Given the set of $H_{u,t}(n)$ of n HN POIs of u at t obtained by Step 2 and the corresponding positive POI $v_{u,t}$, we train model $Q_{u,(t-1)}$ with the following *cross-entropy* loss [14]:

$$\mathcal{L}_{u,t} = -(log(Q_{u,(t-1)}(v_{u,t})) + \sum_{\bar{v} \in H_{u,t}(n)} log(1 - Q_{u,(t-1)}(\bar{v}))). \tag{6}$$

To minimize $\mathcal{L}_{u,t}$, $Q_{u,(t-1)}$ is trained toward predicting the preference score for $v_{u,t}$ close to 1 and the preference scores for $\bar{v} \in H_{u,t}(n)$ close to 0. We refer to $Q_{u,t}$ as the model trained by these positive POI and HN POIs of u at t in this way (*i.e.*, the model trained by the check-in sequence up to t). At the "next" time point (t+1), we repeat Steps 1 and 2 through the model $Q_{u,t}$. For example, as shown in Figure 3, we sample new HN POIs of u at t_2 by using Q_{u,t_1} which has been trained by u's positive POI v_a and two HN POIs v_b and v_d at t_1.

We repeat this process until the "last" time point in u's check-in sequence. We also train the model Q with respect to other users by repeating Steps 1 and 2 at every time point until the last point of their check-in sequence. Finally, through the model Q trained by *full* check-in sequences of all users in this way, we find and recommend next-POIs for each user u at the current time point.

Algorithm 1 Our proposed model-training scheme

Require: T, M, m, and n
1: Initialize parameters of $Q_{u,0}$
2: **for** $t \leftarrow 1$ to T **do**
3: Construct $\bar{V}_{u,t}$ with M negative POIs
4: **for** each POI $\bar{v} \in \bar{V}_{u,t}$ **do**
5: $c(u, \bar{v}, t) \leftarrow Q_{u,(t-1)}(\bar{v})$
6: **end for**
7: Sort all POIs in $\bar{V}_{u,t}$ based on $c(u, \bar{v}, t)$
8: $C_{u,t}(m) \leftarrow \emptyset$
9: **for** $i \leftarrow 1$ to m **do**
10: Insert $\bar{V}_{u,t}[i]$ into $C_{u,t}(m)$
11: **end for**
12: **for** each POI $\bar{v} \in C_{u,t}(m)$ **do**
13: Compute $d(u, \bar{v}, t)$ by Eq. (3) and $DoP(u, \bar{v}, t)$ by Eq. (4)
14: **end for**
15: Sort all POIs in $C_{u,t}(m)$ based on $DoP(u, \bar{v}, t)$
16: $H_{u,t}(n) \leftarrow \emptyset$
17: **for** $i \leftarrow 1$ to n **do**
18: Insert $C_{u,t}(m)[i]$ into $H_{u,t}(n)$
19: **end for**
20: Train $Q_{u,(t-1)}$ with Eq. (6) and update to $Q_{u,t}$
21: **end for**
22: Recommend POIs that u would check-in at (T+1) via $Q_{u,T}$

Algorithm 1 sketches the model training process through our scheme using one user and one epoch for ease of explanation. Firstly, T denotes the length of the user u's check-in sequence (*i.e.*, the number of check-in points), and we perform the model training through HN sampling a total of T times (lines 2–21). Specifically, in lines 3–10, the set $C_{u,t}(m)$ of candidate HN POIs for user u at time point t is determined through filtering based on preference scores (*i.e.*, $c(u, \bar{v}, t)$) (Step 1). Then, in lines 12–19, among the POIs belonging to $C_{u,t}(m)$, the top-n POIs with high DoPs are considered as u's HN POIs, composing the set $H_{u,t}(n)$ (Step 2). In line 20, the model Q is trained by using HN POIs. Finally, as in line 22, recommendations for next-POIs for a time point (T+1) are provided to u by the model $Q_{u,T}$ trained by a "full" check-in sequence of u.

3.5 Time complexity analysis

Assuming the time required for predicting preference scores for one sample is denoted as s, in Step 1, we calculate preference scores for M samples and then sort the samples based on these scores to find m HN POI candidates (*i.e.*, $O(M \cdot s + M \cdot \log(M))$). In Step

[7] For the experimental results of adopting the *sigmoid*, please refer to Appendix C.3.

Table 1: Comparison of accuracy between the original RN sampling (*i.e.*, Orig) and our proposed HN sampling scheme (*i.e.*, Ours)

Dataset	Metric	PLSPL			CatDM			STAN			STKGRec			GeoSAN		
		Orig	Ours	Gain (%)	Orig	Ours	Gain (%)	Orig	Ours	Gain (%)	Orig	Ours	Gain (%)	Orig	Ours	Gain (%)
NYC	H@5	0.272	**0.400**	47.1	0.220	**0.249**	13.2	0.307	**0.399**	30.0	0.402	**0.436**	8.5	0.356	**0.441**	23.9
	H@10	0.402	**0.519**	29.1	0.261	**0.291**	11.5	0.393	**0.461**	17.3	0.484	**0.523**	8.1	0.463	**0.566**	22.2
	G@5	0.172	**0.281**	63.4	0.189	**0.216**	14.3	0.215	**0.290**	34.9	0.299	**0.324**	8.4	0.223	**0.283**	26.9
	G@10	0.214	**0.320**	49.5	0.203	**0.230**	13.3	0.243	**0.310**	27.6	0.326	**0.353**	8.3	0.259	**0.327**	26.3
TKY	H@5	0.172	**0.314**	82.5	0.188	**0.227**	20.9	0.209	**0.310**	48.3	0.398	**0.427**	7.3	0.534	**0.641**	19.9
	H@10	0.257	**0.433**	68.5	0.241	**0.282**	16.8	0.290	**0.387**	33.4	0.469	**0.503**	7.2	0.630	**0.743**	18.0
	G@5	0.115	**0.210**	82.8	0.164	**0.200**	22.2	0.142	**0.220**	54.9	0.306	**0.332**	8.5	0.385	**0.460**	19.7
	G@10	0.142	**0.248**	74.6	0.182	**0.218**	19.9	0.168	**0.245**	45.8	0.329	**0.356**	8.2	0.419	**0.500**	19.1
Bright -kite	H@5	0.655	**0.737**	12.5	0.633	**0.640**	1.2	0.569	**0.699**	22.8	0.68	**0.696**	2.4	0.650	**0.668**	2.7
	H@10	0.707	**0.774**	9.5	0.644	**0.652**	1.3	0.644	**0.753**	16.9	0.736	**0.754**	2.4	0.724	**0.789**	9.1
	G@5	0.470	**0.609**	29.5	0.626	**0.634**	1.3	0.433	**0.570**	31.6	0.568	**0.585**	3.0	0.390	**0.474**	21.7
	G@10	0.488	**0.621**	27.3	0.630	**0.638**	1.3	0.469	**0.588**	25.4	0.587	**0.604**	3.0	0.427	**0.513**	20.1

2, we calculate the DoP for m candidate POIs, sort the samples based on this, and determine n HN POIs (*i.e.*, $O(m \cdot \log(m))$). Then, we compute the cross-entropy loss from the predicted preference scores for the n HN POIs and the corresponding positive POI. Since we have already calculated preference scores for all M POIs, only $O(n)$ time is required for loss calculation. In summary, the time complexity of our scheme is $O(M \cdot s + M \cdot \log(M) + m \cdot \log(m) + n)$. Here, m and n are much smaller than M (*e.g.*, M=500, m=50, and n=10), so it can be approximated to $O(M \cdot s + M \cdot \log(M))$. For RN sampling, the time complexity for finding negative samples is $O(1)$. Then, to calculate the loss, we need to compute preference scores for n negative samples and derive the cross-entropy loss from them. Hence, the time complexity is $O(n \cdot s + n)$, which can be approximated to $O(n \cdot s)$.

Note that using *GPU parallelization* can reduce the time complexity for predicting preference scores to a maximum of s and the time complexity for sorting to a maximum of $\log(M)$ (depending on the number of GPU processing units) [21, 29]. Thus, with the parallelization, the time complexity of our scheme can become $O(s + \log(M))$ while that of RN sampling can become $O(s)$. Moreover, the number of epochs required for *convergence* is typically *larger* for RN sampling than for HN sampling [26, 35, 39]. Therefore, the overall time complexity of our scheme and RN sampling is $O((s + \log(M)) \cdot e)$ and $O(s \cdot E)$, respectively, where e and E ($e < E$) denote the number of epochs. In RQ 4 in Section 4, we demonstrate that the elapsed time of our scheme becomes *smaller* than that of RN sampling, due to the significant difference between E and e.

4 EVALUATION

4.1 Experimental Setup

We performed experiments on three real-world datasets, which have been widely adopted in the next-POI recommendation: NYC and TKY from Foursquare [36] and Brightkite [6] as shown in Table A in Appendix A.2. We employed two popular metrics used in existing studies [23, 25, 38] to compute the accuracy: *hit rate* (namely, H) and *NDCG* (namely, G). As a (base) next-POI recommender model, we used five state-of-the-art models: PLSPL [34], CatDM [38], STAN [25], STKGRec [5], and GeoSAN [23]. We set the number m of candidates for the HN POIs and the number n of HN POIs to sample at each time point to 50 and 10, respectively. For the specifics of the evaluation protocol, refer to Appendix A.3.

4.2 Experimental Results

Our experiments are designed to answer the following four key research questions (RQs).

- **RQ 1.** How effective is our proposed training scheme for state-of-the-art base models?
- **RQ 2.** How effective is filtering out POIs in Step 1?
- **RQ 3.** How effective is considering the distance in Step 2?
- **RQ 4.** How much time does our proposed scheme require?
- **RQ 5.** How does the accuracy vary depending on n?[8]

4.2.1 RQ 1. For each base model, we compare the accuracy between the model trained by our proposed scheme based on HN sampling (*i.e.*, Steps 1 and 2) and the model trained by its original training scheme. For example, the original STAN [25] is trained by randomly sampling n negative POIs among all negative POIs of a user, while the original GeoSAN [23] is trained by randomly sampling n negative POIs among the negative POIs geographically close to a user.

In Table 1, for *all* the datasets, we observe that *all* models trained by our proposed scheme (*i.e.*, Ours) consistently and universally outperform the original model (*i.e.*, Orig) in terms of *all* the metrics. For Brightkite, the gains are slightly smaller than those for the other two datasets. We attribute this to the fact that the average number of POIs checked-in by a user in Brightkite (*i.e.*, 18) is much smaller than in NYC (*i.e.*, 44) and TKY (*i.e.*, 60), which might make it relatively difficult for a model to give accurate preference scores to negative POIs. Nonetheless, the consistent results for all the datasets successfully validate that our proposed scheme using the

[8] For the results with respect to m, please refer to Appendix C.6.

Table 2: Comparison of accuracy among our scheme and its two variants with respect to filtering out POIs in Step 1

Dataset	Metric	PLSPL			CatDM		
		Filt-d	*w/o* Filt	Ours	Filt-d	*w/o* Filt	Ours
NYC	H@5	0.328	0.381	**0.400**	0.226	0.234	**0.249**
	H@10	0.453	0.508	**0.519**	0.267	0.282	**0.291**
	G@5	0.220	0.261	**0.281**	0.195	0.202	**0.216**
	G@10	0.260	0.302	**0.320**	0.209	0.218	**0.230**
TKY	H@5	0.207	0.268	**0.314**	0.187	0.192	**0.227**
	H@10	0.347	0.396	**0.433**	0.241	0.247	**0.282**
	G@5	0.127	0.171	**0.210**	0.159	0.166	**0.200**
	G@10	0.172	0.212	**0.248**	0.177	0.184	**0.218**

Dataset	Metric	STAN			STKGRec		
		Filt-d	*w/o* Filt	Ours	Filt-d	*w/o* Filt	Ours
NYC	H@5	0.267	0.344	**0.399**	0.431	0.435	**0.436**
	H@10	0.343	0.406	**0.461**	0.514	0.518	**0.523**
	G@5	0.189	0.248	**0.290**	0.320	0.322	**0.324**
	G@10	0.214	0.268	**0.310**	0.347	0.349	**0.353**
TKY	H@5	0.200	0.282	**0.310**	0.420	0.426	**0.427**
	H@10	0.272	0.362	**0.387**	0.494	**0.504**	0.503
	G@5	0.137	0.197	**0.220**	0.323	0.329	**0.332**
	G@10	0.160	0.223	**0.245**	0.347	0.354	**0.356**

concept of DoP is *more beneficial* to improving accuracy than the RN sampling scheme employed in the existing models.

4.2.2 RQ 2. We designed two variants of our scheme with respect to filtering out POIs in Step 1: (i) filtering out negative POIs with the high value of $d(u, \bar{v}, t)$ (*i.e.*, the POIs geographically far from u at t) and then sampling HN POIs based on DoP among the remaining POIs (namely, Filt-d); and (ii) sampling HN POIs based on DoP *without filtering* out any negative POIs (namely, *w/o* Filt). Between two factors of DoP, Filt-d prioritizes $d(u, \bar{v}, t)$ over $c(u, \bar{v}, t)$, while *w/o* Filt gives equal priority to two factors.

Table 2 shows the results on the NYC and TKY datasets. The order of the three schemes from the most accurate to the least accurate is (Ours > *w/o* Filt > Filt-d) regardless of the models, which indicates the following observation: *prioritizing the preference score* (*i.e.*, $c(u, \bar{v}, t)$) *rather than the distance* (*i.e.*, $d(u, \bar{v}, t)$) is more effective in accurately finding the HN POIs.

Moreover, to verify whether these negative POIs help the model correctly position the user's positive POI in a high ranking during training, we observed the difference in accuracy among three schemes at every epoch during training.

Figure 5 displays the results, where we set the maximum of epochs to 50 and obtained NDCG (G@10) through the validation set for NYC.[9] For PLSPL [34] and CatDM [38], Ours considerably contributes to improving the accuracy from the beginning of the training and converges to higher accuracy than the other two

[9]For the results of using another dataset (*i.e.*, TKY), please refer to Appendix C.4.

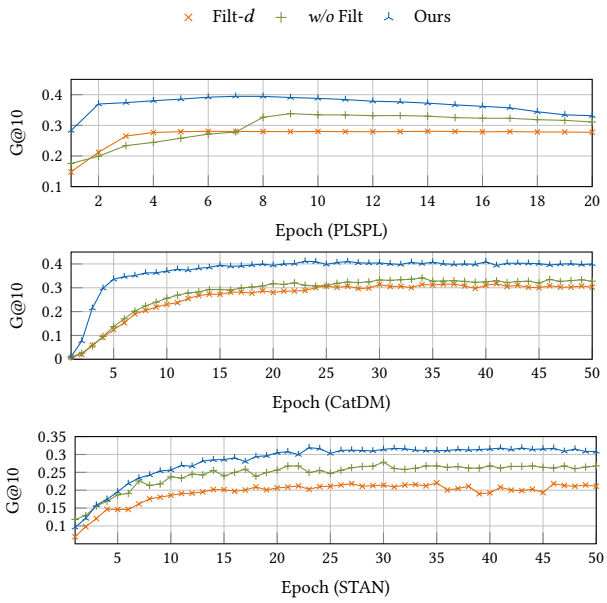


Figure 5: Accuracy obtained as training proceeds when using PLSPL, CatDM, and STAN as a base model (NYC).

schemes. For STAN [25], in the early phase of the training (*i.e.*, less than 5 epochs), the three schemes do not show a significant difference in accuracy. As the training progresses, however, the model trained by Ours shows the greatest improvement in accuracy and finally converges to the highest accuracy. These results indicate that our strategy of filtering out POIs, which prioritizes $c(u, \bar{v}, t)$ over $d(u, \bar{v}, t)$, is effective for next-POI recommendations.

4.2.3 RQ 3. We designed two variants of our scheme with respect to considering a geographical distance in Step 2: (i) sampling the POIs with the *long distance* from the user (*i.e.*, contrary to our scheme) as HN POIs among the POIs obtained by Step 1 (namely, Dist-l)[10]; and (ii) sampling top-n POIs with the highest value of $c(u, \bar{v}, t)$ *without considering the distance* (namely, *w/o* Dist).[11]

Table 3 exhibits the following observations: (i) exploiting the distance based on the strategy (*i.e.*, Dist-l) *opposite* to Ours shows worse accuracy than using only the preference scores to find the HN POIs (*i.e.*, *w/o* Dist); (ii) pursuing a high preference score and a short distance in HN sampling enhances accuracy.

4.2.4 RQ 4. In Table 4, we present the execution time of our proposed scheme and RN sampling, where NYC and PLSPL [34] are employed as the dataset and the base model, respectively. As our proposed ideas for HN sampling (*i.e.*, Steps 1 and 2) are incorporated, the elapsed time of Ours per epoch increases, compared to the original PLSPL employing RN sampling. However, it is worth noting that the total number of epochs required for the model to converge is much *smaller* in Ours (*i.e.*, 12 epochs) than in the original PLSPL (*i.e.*, 47 epochs). This is generally consistent with the

[10]In other words, Dist-l finds top-n HN POIs with the highest $DoP_{long}(u, \bar{v}, t)$ by formulating it as $(c(u, \bar{v}, t) \cdot d(u, \bar{v}, t))$.

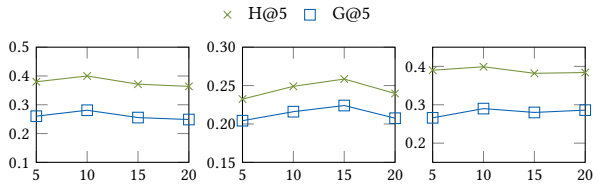
[11]For the results of using only the distance to determine HN POIs (*i.e.*, *w/o* Score), please refer to Appendix C.5.

Table 3: Comparison of accuracy among our scheme and its two variants with respect to using a distance in Step 2

Dataset	Metric	PLSPL			CatDM		
		Dist-l	*w/o* Dist	Ours	Dist-l	*w/o* Dist	Ours
NYC	H@5	0.369	0.386	**0.400**	0.205	0.243	**0.249**
	H@10	0.499	0.508	**0.519**	0.247	0.281	**0.291**
	G@5	0.253	0.265	**0.281**	0.168	0.211	**0.216**
	G@10	0.295	0.305	**0.320**	0.184	0.223	**0.230**
TKY	H@5	0.267	0.286	**0.314**	0.185	0.222	**0.227**
	H@10	0.401	0.406	**0.433**	0.238	0.272	**0.282**
	G@5	0.171	0.185	**0.210**	0.155	0.198	**0.200**
	G@10	0.213	0.224	**0.248**	0.173	0.213	**0.218**
Dataset	Metric	STAN			STKGRec		
		Dist-l	*w/o* Dist	Ours	Dist-l	*w/o* Dist	Ours
NYC	H@5	0.367	0.381	**0.399**	0.418	0.425	**0.436**
	H@10	0.440	0.444	**0.461**	0.492	0.514	**0.523**
	G@5	0.263	0.276	**0.290**	0.323	0.317	**0.324**
	G@10	0.287	0.297	**0.310**	0.347	0.347	**0.353**
TKY	H@5	0.286	0.305	**0.310**	0.423	0.423	**0.427**
	H@10	0.373	0.386	**0.387**	0.499	0.495	**0.503**
	G@5	0.203	0.212	**0.220**	0.326	0.329	**0.332**
	G@10	0.231	0.239	**0.245**	0.349	0.353	**0.356**

Table 4: Comparison of elapsed times (PLSPL)

Dataset	Method	Elapsed time (sec)	
		1 epoch	Convergence
NYC	Orig	140.4	**6598.8** (47 epochs)
	Ours (Step 1)	164.1	1969.2 (12 epochs)
	Ours (Steps 1/2)	384.5	**4614.0** (12 epochs)



(a) PLSPL **(b) CatDM** **(c) STAN**

Figure 6: Accuracies obtained by varying n (NYC).

previous research findings indicating that the number of epochs required for accuracy convergence in HN sampling is smaller than RN sampling [26, 35, 39]. Consequently, even when applying all our ideas to train the model, the total time required for convergence is *less* than that of the original PLSPL by about 30%.

4.2.5 RQ 5. In Figure 6, we show the changes in accuracy by varying the number n of HN POIs to sample from 5 to 20 in increments of 5. Here, we employed PLSPL [34], CatDM [38], and STAN [25] as base models. Overall, we can observe that our proposed model-training scheme is *insensitive* to the value of n, and we adopted ten as n, which shows marginally better results than the rest.

5 RELATED WORK

In general recommendation domains such as OTT, e-commerce, and multimedia [4, 15–17, 19, 20], various studies for HN sampling have been conducted. First, DNS [39] and AOBPR [26] regard a negative item with a *high preference score* predicted by the models as *informative* for their model training. The loss over a positive item and the negative item results in *large gradients*, which can help the models be effectively trained in the *correct way*. In particular, AOBPR [26] introduces a way to *efficiently* find HN samples through a *mixture model of the sampling distribution*. In [28], researchers demonstrate that HN sampling is more advantageous than *non-sampling* in optimizing the *One-way Partial AUC* (OPAUC) metric, thereby validating that HN sampling is more effective than non-sampling in improving the top-N recommendation accuracy. Next, [40] claims that sampling items that are hard negative as well as true negative can prevent the model from *overfitting*. In this sense, GDNS [40] presents a way of finding these negative samples by observing changes in predicted preference scores for negative samples over *successive epochs*.[12] Finally, RecNS [37] proposes a HN sampling method for *graph-based recommendation* models (*e.g.*, GNN [8], LightGCN [9]). Our work differs from these studies in that we propose the model-training scheme based on HN sampling in the next-POI recommendation for the first time.

6 CONCLUSIONS

In this paper, we conducted an *exhaustive and comprehensive* study of the negative sampling issue for the next-POI recommendation, while most existing approaches have simply employed the RN sampling for each user during model training. Based on our key observation that RN sampling performs as *EN sampling* as model training proceeds, we pointed out the limitation of existing studies by validating that EN sampling is more *disadvantageous* than HN sampling in terms of improving accuracy. To address this limitation, we introduced the novel concept of *DoP* that determines HN POIs, proposing the model-training scheme based on HN sampling with the following two steps: (Step 1) filtering out the POIs by the degree of having the characteristics preferred by a user; (Step 2) sampling HN POIs, located close to her positive POI *both in the latent/geographical space*, via DoP. Experimental results demonstrated that all the state-of-the-art models incorporated with our training scheme achieved significant benefits in accuracy.

ACKNOWLEDGMENT

This work was supported by Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No.RS-2022-00155586 (A High-Performance Big-Hypergraph Mining Platform for Real-World Downstream Tasks), No.2020-0-01373 (Artificial Intelligence Graduate School Program (Hanyang University)), and No.2022-0-00352).

[12]For the results of comparison with employing GDNS, please refer to Appendix C.7.

REFERENCES

[1] Hong-Kyun Bae, Jeewon Ahn, Dongwon Lee, and Sang-Wook Kim. 2023. LANCER: A Lifetime-Aware News Recommender System. In *AAAI Conference on Artificial Intelligence (AAAI)*. 4141–4148.

[2] Hong-Kyun Bae, Yeon-Chang Lee, and Kyungsik Han Sang-Wook Kim. 2023. A Competition-Aware Approach to Accurate TV Show Recommendation. In *IEEE International Conference on Data Engineering (ICDE)*. 2822–2834.

[3] Gang Cao, Shengmin Cui, and Inwhee Joe. 2023. Improving the Spatial–temporal Aware Attention Network with Dynamic Trajectory Graph Learning for Next Point-Of-Interest Recommendation. *Information Processing & Management* 60, 3 (2023), 1–19.

[4] Dong-Kyu Chae, Jihoo Kim, Sang-Wook Kim, and Duen Horng Chau. 2020. AR-CF: Augmenting Virtual Users and Items in Collaborative Filtering for Addressing Cold-Start Problems. In *ACM SIGIR Conference on Research and Development in Information Retrieval*. 1251–1260.

[5] Wei Chen, Huaiyu Wan, Shengnan Guo, Haoyu Huang, Shaojie Zheng, Jiamu Li, Shuohao Lin, and Youfang Lin. 2022. Building and Exploiting Spatial–temporal Knowledge Graph for Next POI Recommendation. *Knowledge-Based Systems* 258 (2022), 1–12.

[6] Eunjoon Cho, Seth A Myers, and Jure Leskovec. 2011. Friendship and Mobility: User Movement in Location-based Social networks. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 1082–1090.

[7] Kyung-Jae Cho, Yeon-Chang Lee, Kyungsik Han, Jaeho Choi, and Sang-Wook Kim. 2019. No, That's Not My Feedback: TV Show Recommendation Using Watchable Interval. In *IEEE International Conference on Data Engineering (ICDE)*. 316–327.

[8] Wenqi Fan, Yao Ma, Qing Li, Yuan He, Eric Zhao, Jiliang Tang, and Dawei Yin. 2019. Graph Neural Networks for Social Recommendation. In *International Conference on World Wide Web (WWW)*. 417–426.

[9] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, YongDong Zhang, and Meng Wang. 2020. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. In *ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. 639–648.

[10] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-term Memory. *Neural Computation* 9, 8 (1997), 1735–1780.

[11] Zheng Huang, Jing Ma, Yushun Dong, Natasha Zhang Foutz, and Jundong Li. 2022. Empowering Next POI Recommendation with Multi-relational Modeling. In *ACM SIGIR Conference on Research and Development in Information Retrieval*. 2034–2038.

[12] Won-Seok Hwang, Juan Parc, Sang-Wook Kim, Jongwuk Lee, and Dongwon Lee. 2016. "Told you I didn't like it": Exploiting Uninteresting Items for Effective Collaborative Filtering. In *IEEE International Conference on Data Engineering (ICDE)*. 349–360.

[13] Md Ashraful Islam, Mir Mahathir Mohammad, Sarkar Snigdha Sarathi Das, and Mohammed Eunus Ali. 2022. A Survey on Deep Learning based Point-of-Interest (POI) Recommendations. *Neurocomputing* 472 (2022), 306–325.

[14] Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive Sequential Recommendation. In *IEEE International Conference on Data Mining (ICDM)*. 197–206.

[15] Taeri Kim, Yeon-Chang Lee, Kijung Shin, and Sang-Wook Kim. 2022. MARIO: Modality-Aware Attention and Modality-Preserving Decoders for Multimedia Recommendation. In *ACM International Conference on Information and Knowledge Management (CIKM)*. 993–1002.

[16] Taeho Kim, Juwon Yu, Won-Yong Shin, Hyunyoung Lee, Ji hui Im, and Sang-Wook Kim. 2023. LATTE: A Framework for Learning Item-features to Make a Domain-Expert for Effective Conversational Recommendation. In *ACM SIGKDD International Conference on Knowledge Discovery & Data mining*. 1144–1153.

[17] Yunyong Ko, Jae-Seo Yu, Hong-Kyun Bae, Yongjun Park, Dongwon Lee, and Sang-Wook Kim. 2021. MASCOT: A Quantization Framework for Efficient Matrix Factorization in Recommender Systems. In *IEEE International Conference on Data Mining (ICDM)*. 290–299.

[18] Jongwuk Lee, Won-Seok Hwang, Juan Parc, Youngnam Lee, Sang-Wook Kim, and Dongwon Lee. 2019. l-Injection: Toward Effective Collaborative Filtering Using Uninteresting Items. *IEEE Transactions on Knowledge and Data Engineering* 31, 1 (2019), 3–16.

[19] Yeon-Chang Lee, Sang-Wook Kim, and Dongwon Lee. 2018. gOCCF: Graph-theoretic One-class Collaborative Filtering based on Uninteresting Items. In *AAAI Conference on Artificial Intelligence (AAAI)*. 3448–3456.

[20] Youngnam Lee, Sang-Wook Kim, Sunju Park, and Xing Xie. 2018. How to Impute Missing Ratings?: Claims, Solution, and its Application to Collaborative Filtering. In *International Conference on World Wide Web (WWW)*. 783–792.

[21] Hao Li, Kenli Li, Jiyao An, and Keqin Li. 2018. MSGD: A Novel Matrix Factorization Approach for Large-Scale Collaborative Filtering Recommender Systems on GPUs. *IEEE Transactions On Parallel and Distributed Systems* 29, 7 (2018), 1530–1544.

[22] Ranzhen Li, Yanyan Shen, and Yanmin Zhu. 2018. Next point-of-interest recommendation with temporal and multi-level context attention. In *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 1110–1115.

[23] Defu Lian, Yongji Wu, Yong Ge, Xing Xie, and Enhong Chen. 2020. Geography-Aware Sequential Location Recommendation. In *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2009–2019.

[24] Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. 2016. Predicting the Next Location: A Recurrent Model with Spatial and Temporal Contexts. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

[25] Yingtao Luo, Qiang Liu, and Zhaocheng Liu. 2021. STAN: Spatio-temporal Attention Network for Next Location Recommendation. In *International Conference on World Wide Web (WWW)*. 2177–2185.

[26] Steffen Rendle and Christoph Freudenthaler. 2014. Improving Pairwise Learning for Item Recommendation from Implicit Feedback. In *ACM International Conference on Web Search and Data Mining (WSDM)*. 273–282.

[27] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. 1986. Learning Representations by Back-propagating Errors. *Nature* 323, 6088 (1986), 533–536.

[28] Wentao Shi, Jiawei Chen, Fuli Feng, Jizhi Zhang, Junkang Wu, Chongming Gao, and Xiangnan He. 2023. On the Theories Behind Hard Negative Sampling for Recommendation. In *International Conference on World Wide Web (WWW)*. 812–822.

[29] Wei Tan, Liangliang Cao, and Liana Fong. 2016. Faster and Cheaper: Parallelizing Large-Scale Matrix Factorization on GPUs. In *ACM International Symposium on High-Performance Parallel and Distributed Computing (HPDC)*. 219–230.

[30] Glen Van Brummelen and Elizabeth A Hamm. 2014. Heavenly Mathematics: The Forgotten Art of Spherical Trigonometry. *Aestimatio: Sources and Studies in the History of Science* 11 (2014), 127–130.

[31] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is All You Need. *Advances in Neural Information Processing Systems (NeurIPS)* 30 (2017).

[32] Zhaobo Wang, Yanmin Zhu, Haobing Liu, and Chunyang Wang. 2022. Learning Graph-based Disentangled Representations for Next POI Recommendation. In *ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. 1154–1163.

[33] Heitor Werneck, Nícollas Silva, Matheus Carvalho Viana, Fernando Mourão, Adriano CM Pereira, and Leonardo Rocha. 2020. A Survey on Point-of-Interest Recommendation in Location-based Social Networks. In *Brazilian Symposium on Multimedia and the Web (WebMedia)*. 185–192.

[34] Yuxia Wu, Ke Li, Guoshuai Zhao, and Xueming Qian. 2020. Personalized Long-and Short-term Preference Learning for Next POI Recommendation. *IEEE Transactions on Knowledge and Data Engineering* 34, 4 (2020), 1944–1957.

[35] Lanling Xu, Jianxun Lian, Wayne Xin Zhao, Ming Gong, Linjun Shou, Daxin Jiang, Xing Xie, and Ji-Rong Wen. 2022. Negative Sampling for Contrastive Representation Learning: A Review. *arXiv preprint arXiv:2206.00212* (2022).

[36] Dingqi Yang, Daqing Zhang, Vincent. W. Zheng, and Zhiyong Yu. 2015. Modeling User Activity Preference by Leveraging User Spatial Temporal Characteristics in LBSNs. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 45, 1 (2015), 129–142.

[37] Zhen Yang, Ming Ding, Xu Zou, Jie Tang, Bin Xu, Chang Zhou, and Hongxia Yang. 2023. Region or Global? A Principle for Negative Sampling in Graph-Based Recommendation. *IEEE Transactions on Knowledge and Data Engineering (IEEE TKDE)* 35, 6 (2023), 6264–6277.

[38] Fuqiang Yu, Lizhen Cui, Wei Guo, Xudong Lu, Qingzhong Li, and Hua Lu. 2020. A Category-aware Deep Model for Successive POI Recommendation on Sparse Check-in Data. In *International Conference on World Wide Web (WWW)*. 1264–1274.

[39] Weinan Zhang, Tianqi Chen, Jun Wang, and Yong Yu. 2013. Optimizing Top-N Collaborative Filtering via Dynamic Negative Item Sampling. In *ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. 785–788.

[40] Qiannan Zhu, Haobo Zhang, Qing He, and Zhicheng Dou. 2022. A Gain-Tuning Dynamic Negative Sampler for Recommendation. In *International Conference on World Wide Web (WWW)*. 277–285.

Table A: Statistics of three real-world datasets

Dataset	# of users	# of POIs	# of check-ins	Sparsity (%)
NYC	1,083	8,434	171,493	99.47
TKY	2,293	12,740	482,118	99.52
Brightkite	1,866	11,698	704,673	99.84

A IMPLEMENTATION DETAIL

A.1 Environments

Base models for the next-POI recommendation were implemented with PyTorch 1.13.1, Pandas 1.1.5, Tensorflow 1.15.0, scikit-learn 1.0.2, Torchtext 0.6.0, and Python 3.7. All experiments were conducted on desktops with 64 GB memory, Intel i9-10900K CPU (3.7 GHz, 20M cache), and Nvidia GeForce RTX 3070.

A.2 Dataset Preprocessing

We conducted experiments on three popular real-world datasets, NYC and TKY from Foursquare, and Brightkite. The statistics of these datasets are shown in Table A. The sparsity of the dataset equals the ratio of missing cells out of the total cells in the user-POI check-in matrix. For NYC and TKY, which are check-in datasets collected from New York and Tokyo from April 12, 2012, to February 16, 2013, respectively, we kept only users who had visited more than 5 POIs and kept only POIs visited by more than 5 users. For GeoSAN [23], however, we maintained only users whose check-in sequence length was at least 100 due to the memory space issue. For Brightkite, which contains a vast number of check-ins of users from April 2008 to October 2010, we left users whose check-in sequence length was at least 100, and we left POIs visited by more than 10 users. We note that PLSPL [34] and CatDM [38] exploit categories of POIs while training their models, but Brightkite does not include category information. Thus, we obtained category information of POIs from the Foursquare API and then *merged* them into the Brightkite dataset.

A.3 Evaluation Protocol

To evaluate the accuracy of base models precisely, we divided each dataset into training/validation/test sets and tuned the hyperparameters by following the respective study that proposed each model.

- PLSPL [34]: The training set, validation set, and test set consist of the initial 80%, 10%, and 10% of the check-ins of each user, respectively. We set the batch size, hidden dimension, number of epochs, and learning rate as 32, 128, 20, and 0.001, respectively. Moreover, we note that PLSPL [34] is trained by using *all* negative POIs of a user *without sampling*. To reduce the computation overhead, we trained the original PLSPL with a value of n large enough (n=500) to show results similar to its original results.
- CatDM [38]: The training set, validation set, and test set consist of the initial 80%, 10%, and 10% of the check-ins of each user, respectively. Within the test set, we selected the last 24-hour check-ins for every user, designating the first POI as the current location, and regarding the subsequent check-ins as the ground truth. We set the batch size, hidden dimension, and learning rate as 1, 64, and 0.001, respectively. We employed the pair-wise loss function to train CatDM.
- STAN [25]: The training set, validation set, and test set consist of the initial 80%, 10%, and 10% of the check-ins of each user, respectively. We set the batch size, number of epochs, and learning rate as 1, 50, and 0.003, respectively.
- GeoSAN [23]: For each user's check-in sequence, we regard the last check-in as the test set and the remaining (previous) check-ins as the training set for the user. We divided each user's check-in sequence into subgroups of 100 check-ins. Then, we regarded the last sub-group as the test set and the remaining subgroups as the training set for the user. We set the batch size, hidden dimension, number of epochs, and learning rate as 32, 100, 100, and 0.001, respectively.
- STKGRec [5]: We divided each user's check-in sequence into 24-hour *sessions*, where at least 3 check-ins are contained in each session. The training set, validation set, and test set consist of the initial 80%, 10%, and 10% of the sessions of each user, respectively. We set the batch size, hidden dimension, number of epochs, and learning rate as 64, 100, 100, and 0.0001, respectively. We employed the pair-wise loss function to train STKGRec.

B NOTATIONS

Table B summarizes the notations used in this paper.

C ADDITIONAL EXPERIMENTAL RESULTS

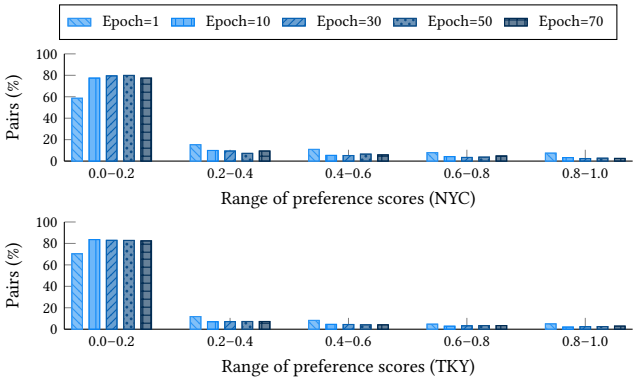


Figure A: Percentage of user-POI pairs according to the range of preference scores predicted by CatDM for each epoch.

C.1 Percentage of user-POI pairs according to the range of preference scores

Figure A shows the results for CatDM, where the x-axis denotes the range of preference scores predicted by the model and the y-axis denotes the percentage of user-POI pairs corresponding to each range. The POIs belonging to the range of 0.0–0.2 and the range of 0.8–1.0 can be regarded as EN POIs and HN POIs of the user, respectively. From the middle phase of the training (*i.e.*, Epoch=30), we observe that most pairs (more than 80% for both datasets) belong to the range of 0.0–0.2; if a negative POI is randomly sampled for a user, it is *highly likely* to be an *EN POI* of that user. In this sense,

Table B: Notations

Notation	Description
u, v, and t	A user u, a POI v, and a time point t
$\bar{V}_{u,t}$	A set of all (negative) POIs that u has not checked-in at t ($\bar{v} \in \bar{V}_{u,t}$)
$c(u, v, t)$	The degree to which v has the characteristics preferred by u, given the previous check-in sequence of u up to the time point t
$d(u, v, t)$	The geographical distance between u and v at t
$Q_{u,t}$	The next-POI recommendation model trained by the check-in sequence of u up to t
$Q_{u,t}(\bar{v})$	The preference score of u for $\bar{v}$ predicted by model $Q_{u,t}$
$C_{u,t}(m)$	A set of m candidates for HN POIs of u at t
$DoP(u, \bar{v}, t)$	DoP of user u for POI $\bar{v}$ at t, formulated by $c(u, \bar{v}, t)$ and $d(u, \bar{v}, t)$
$H_{u,t}(n)$	A set of n HN POIs of u at t
$rank(c(u, \bar{v}, t))$	The ranking of $\bar{v}$ among the POIs in $\bar{V}_{u,t}$ after they are sorted in the descending order of $c(u, \bar{v}, t)$
$rank(DoP(u, \bar{v}, t))$	The ranking of $\bar{v}$ among the POIs in $C_{u,t}(m)$ after they are sorted in the descending order of $DoP(u, \bar{v}, t)$

Table C: Comparison of accuracy among the variants for computing DoP (NYC)

	Metric			
Method	H@5	H@10	G@5	G@10
Original	0.307	0.393	0.215	0.243
Sigmoid	0.391	0.452	0.283	0.303
Multiplication (Ours)	**0.399**	**0.461**	**0.290**	**0.310**
Gain (%)	2.0	2.0	2.5	2.3

the model trained with RN sampling is equivalent (in practice) to the model trained with *EN sampling* as the training progresses.

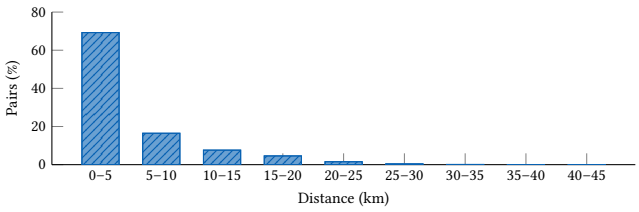


Figure B: Distribution of pairs of successive POIs over the distance between them (TKY).

C.2 Distribution of pairs of successive POIs over the distance between them

Using real-world datasets, we conducted a statistical analysis to verify the relationship between a user's check-in probability for a POI and the distance between the POI and the user. Figure B illustrates the results for the TKY. The x-axis indicates the range of a distance and the y-axis indicates the ratio of pairs of successive POIs corresponding to the range of a distance. It can be observed that about 70% of all pairs for both datasets belong to the range of a distance less than 5km; users generally have a strong tendency to check-in the POI *located close to their current locations* as their next-POIs.

C.3 Accuracy of variants for computing DoP

We designed an additional method for calculating DoP as follows: applying the *sigmoid* function to a linear combination of the predicted preference score $c(u, \bar{v}, t)$ and the distance score $d(u, \bar{v}, t)$ (*i.e.*, $DoP(u, \bar{v}, t) = \sigma(c(u, \bar{v}, t) \cdot (1 - d(u, \bar{v}, t)))$). The experimental results using the STAN [25] model are presented in Table C. In Table C, we observe that, regardless of the metrics, employing the multiplication (*i.e.*, Ours) exhibits higher accuracy than employing the sigmoid function by about 2.5%.

C.4 Accuracy obtained as training proceeds

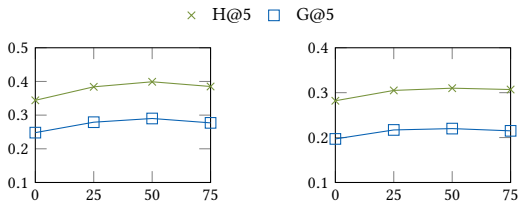
Figure C displays the results, where we set the maximum of epochs to 50 and obtained NDCG (G@10) through the validation set for TKY. For PLSPL [34] and CatDM [38], Ours considerably improves the accuracy from the beginning of the training and converges to higher accuracy than the other two schemes. For STAN [25], in the early phase of the training (*i.e.*, less than 5 epochs), the three schemes do not show a significant difference in accuracy. As the training progresses, however, the model trained by Ours shows the greatest improvement in accuracy and finally converges to the highest accuracy. These results indicate that our strategy of filtering out POIs, which prioritizes $c(u, \bar{v}, t)$ over $d(u, \bar{v}, t)$, is effective for next-POI recommendations.

C.5 Ablation study with respect to determining HN POIs

To verify the effectiveness of our design choice to determine HN POIs, we further evaluated the accuracy with a distance-only scheme (*i.e.*, *w/o* Score). Tables D and E present the experimental results for the NYC and TKY datasets, respectively. For most base models, the scheme of using only preference score (*i.e.*, *w/o* Dist) shows higher accuracy than that of using only distance (*i.e.*, *w/o* Score). This coincides with the results in Table 3 of our paper, which demonstrated that the preference scores have higher importance in determining DoP than the distance.

Table E: Comparison of accuracy with the variations of our training scheme (TKY)

Model	Method	H@5	H@10	G@5	G@10
PLSPL	*w/o* Score	0.143	0.272	0.082	0.122
	w/o Dist	0.286	0.406	0.185	0.224
	Ours	**0.314**	**0.433**	**0.210**	**0.248**
CatDM	*w/o* Score	0.207	0.260	0.183	0.201
	w/o Dist	0.222	0.272	0.198	0.213
	Ours	**0.227**	**0.282**	**0.200**	**0.218**
STK-GRec	*w/o* Score	0.416	0.487	0.321	0.344
	w/o Dist	0.423	0.495	0.329	0.353
	Ours	**0.427**	**0.503**	**0.332**	**0.356**



(a) NYC **(b) TKY**

Figure D: Accuracies obtained by varying m (STAN).

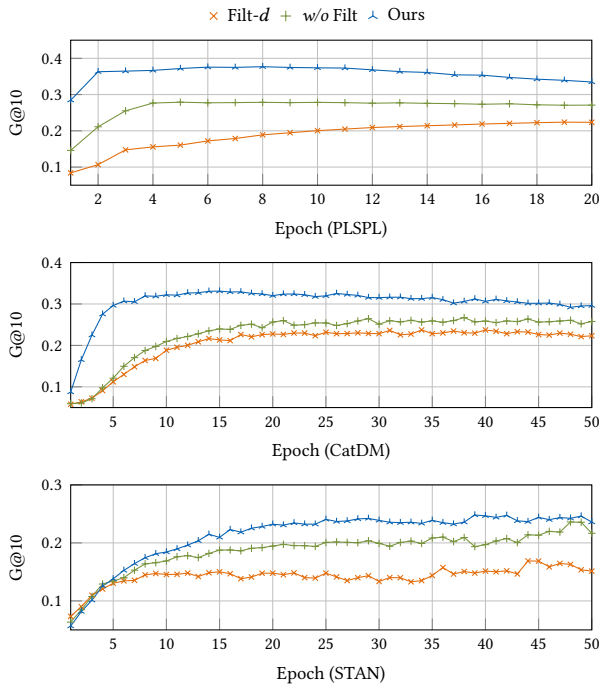


Figure C: Accuracy obtained as training proceeds (TKY).

Table D: Comparison of accuracy with the variations of our training scheme (NYC)

Model	Method	H@5	H@10	G@5	G@10
PLSPL	*w/o* Score	0.204	0.358	0.141	0.188
	w/o Dist	0.386	0.508	0.265	0.305
	Ours	**0.400**	**0.519**	**0.281**	**0.320**
CatDM	*w/o* Score	0.227	0.267	0.198	0.212
	w/o Dist	0.243	0.281	0.211	0.223
	Ours	**0.249**	**0.291**	**0.216**	**0.230**
STK-GRec	*w/o* Score	0.429	0.514	0.320	0.347
	w/o Dist	0.425	0.514	0.317	0.347
	Ours	**0.436**	**0.523**	**0.324**	**0.353**

C.6 Accuracy obtained by varying m

Figure D displays the changes in accuracy depending on the number m of candidates for HN POIs from 0 to 75 in increments of 25, where m=0 indicates the result of sampling HN POIs without filtering out any negative POIs. Regardless of the value of m, we can demonstrate that obtaining candidate HN POIs through filtering is more effective than the case of m=0. Additionally, we can observe that the difference in accuracy is *not substantial* based on the value of m, and we used 50 as the value for m.

C.7 Accuracy of employing state-of-the-art method for HN sampling

We compared the accuracy with applying one of the state-of-the-art HN sampling techniques, GDNS [40], to the next-POI recommendation model. GDNS is based on the assumption that the items with consistently high preference scores over multiple epochs are likely to be false negatives. Therefore, it regards only the items whose preference scores were initially high but decreased over epochs as the user's HN items. Table F shows the results of comparing the accuracy between GDNS and Ours w/o Dist using STAN as the base model are presented. (Note that we evaluated the recommendation accuracy only for a randomly sampled group of 100 users for each dataset.)

Table F: Comparison of accuracy with the variations of our training scheme (TKY)

Dataset	Method	H@5	H@10	G@5	G@10
NYC	GDNS	0.211	0.253	0.163	0.176
	Ours *w/o* Dist	**0.506**	**0.598**	**0.359**	**0.390**
TKY	GDNS	0.201	0.267	0.134	0.155
	Ours *w/o* Dist	**0.333**	**0.424**	**0.237**	**0.266**