Unsupervised Path Representation Learning with Curriculum Negative Sampling

Sean Bin Yang[1], **Chenjuan Guo**[1], **Jilin Hu**[1*], **Jian Tang**[2,3,4], **Bin Yang**[1]
[1]Department of Computer Science, Aalborg University, Denmark
[2]Mila-Quebec AI Institute
[3]HEC Montreal, Canada
[4]CIFAR AI Research Chair
{seany, cguo, hujilin, byang}@cs.aau.dk, jian.tang@hec.ca

Abstract

Path representations are critical in a variety of transportation applications, such as estimating path ranking in path recommendation systems and estimating path travel time in navigation systems. Existing studies often learn task-specific path representations in a supervised manner, which require a large amount of labeled training data and generalize poorly to other tasks. We propose an unsupervised learning framework Path InfoMax (*PIM*) to learn generic path representations that work for different downstream tasks. We first propose a curriculum negative sampling method, for each input path, to generate a small amount of negative paths, by following the principles of curriculum learning. Next, *PIM* employs mutual information maximization to learn path representations from both a global and a local view. In the global view, *PIM* distinguishes the representations of the input paths from those of the negative paths. In the local view, *PIM* distinguishes the input path representations from the representations of the nodes that appear only in the negative paths. This enables the learned path representations encode both global and local information at different scales. Extensive experiments on two downstream tasks, ranking score estimation and travel time estimation, using two road network datasets suggest that *PIM* significantly outperforms other unsupervised methods and is also able to be used as a pre-training method to enhance supervised path representation learning.

1 Introduction

Path representations are crucial for various transportation applications, e.g., travel cost estimation [Hu *et al.*, 2020; Pedersen *et al.*, 2020a], routing [Guo *et al.*, 2020; Pedersen *et al.*, 2020b], path recommendation [Yang and Yang, 2020; Guo *et al.*, 2018], and traffic analysis [Hu *et al.*, 2019; Cirstea *et al.*, 2021]. Path representation learning (*PRL*) aims to obtain distinguishable path representations for different paths in a transportation network and hence facilitating various downstream applications. Existing studies on *PRL* often learn path representations in a supervised manner, which has two limitations. First, they require a large amount of labelled training data. Second, the learned path representations are task-specific, e.g., working well for the task with labels, but generalize poorly to other tasks. The two limitations restrict supervised path representation learning from broader usage, thus calling for unsupervised path representation learning.

Although unsupervised graph representation learning methods exist, they are not designed to capture representations of paths. Node representation learning [Tang *et al.*, 2015; Grover and Leskovec, 2016] learns representations for individual nodes in a graph but does not consider paths, i.e., sequences of nodes. Simply aggregating the node representations of the nodes in a path fails to capture the sequential information in paths. Whole graph representation learning [Sun *et al.*, 2020] learns representations for different graphs, while path representation learning considers different paths from the same graph. In addition, unsupervised graph representation learning often utilize random negative sampling to enable training, which is ineffective for path representation learning.

We propose an unsupervised path representation learning framework Path InfoMax (*PIM*), including a curriculum negative sampling method and a path representation learning method. First, we propose a *curriculum negative sampling strategy* to generate a small number of negative paths for an input path. Instead of randomly select other input paths as negative paths, the strategy follows the principles of curriculum learning [Bengio *et al.*, 2009] to first generate paths that are largely different from the input path and thus are easy to be distinguished from the input path. Then, we gradually generate paths that are increasingly similar to the input path and thus are more difficult to be distinguished from the input path. The proposed curriculum negative sampling facilitates effective learning of distinguishable path representations.

Next, we propose two different discriminators, a *path-path discriminator* and a *path-node discriminator*, to jointly learn path representations. The path-path discriminator captures the representation differences between an input path and its negative paths, which we refer to as a *global* view. The path-node discriminator captures the representation difference between an input path and the representations of the nodes that only appear in its negative paths, which we refer to as a *lo-*

*Corresponding Author: Jilin Hu

cal view. The two discriminators ensure the quality of the learned path representations, because they are distinguishable from not only the representations of negative paths from a global view but also the representations of the nodes in negative paths from a local view. To the best of our knowledge, *PIM* is the first work that studies unsupervised path representation learning. We make the following contributions.

1. We propose a curriculum negative sampling strategy for path representation learning.
2. We propose the path-path and path-node discriminators to learn jointly path representations from a global and a local view.
3. We conduct extensive experiments on two data sets with two downstream tasks to demonstrate the effectiveness of *PIM*.

2 Related Work

Path Representation Learning. Existing proposals on path representation learning are all under the supervised learning setting. Such proposals often require large amount of labeled training data and the learned path representations cannot be easily reused for other tasks. For example, *Deepcas* [Li *et al.*, 2017], *ProxEmbed* [Liu *et al.*, 2017], and *PathRank* [Yang *et al.*, 2020; Yang and Yang, 2020] employ different kinds of RNNs to combine node representations of the nodes in a path to obtain a path representation. Then, the training is performed in an end-to-end fashion by using the labeled training data. Instead, we propose an unsupervised path representation learning framework *PIM* that does not require labeled training data and it generalizes nicely to multiple downstream tasks (cf. Table 1 in Section 5.2). In addition, *PIM* can be used as a pre-training method to enhance existing supervised path representation learning (cf. Figure 3 in Section 5.2). An unsupervised trajectory representation learning method transforms trajectories into images and thus do not apply on paths in graphs [Kieu *et al.*, 2018].

Mutual Information Maximization on Graphs. Motivated by Deep InfoMax [Hjelm *et al.*, 2019], mutual information maximization has been applied for unsupervised graph representation learning. Deep Graph Infomax (*DGI*) [Velickovic *et al.*, 2019] and Graph Mutual Information (*GMI*) [Peng *et al.*, 2020] learn node representations and *InfoGraph* [Sun *et al.*, 2020] learns whole graph representations. Here, negative samples are often randomly drawn from a different graph and the mutual information only considers a local view, e.g., a node representation vs. a graph representation. In *PIM*, we propose a curriculum negative sample strategy to generate negative paths with different overlapping nodes with the input paths from the same graph, which facilitates training. Other advanced negative sampling approaches exist [Wang *et al.*, 2018; Ding *et al.*, 2020], but they are not proposed for graphs and do not follow curriculum learning. In addition, we compute mutual information on both a local view (i.e., the representations of input paths vs. the node representations of negative paths) and a global view (i.e., the representations of input paths vs. negative paths) and use them jointly to train the model, which improves accuracy.

3 Preliminaries

Graph. We consider a directed graph $G = (\mathbb{V}, \mathbb{E})$, where $\mathbb{V}$ is the node set and $\mathbb{E}$ is the edge set and we have $|\mathbb{V}| = N$ and $|\mathbb{E}| = M$. Each node $V_i \in \mathbb{V}$ is associated with a node feature vector $v_i \in \mathbb{R}^D$.

Path. A path $P = \langle V_1, V_2, \ldots, V_Z \rangle$ is a sequence of nodes, where Z is the path length and $P.s = V_1$ and $P.d = V_Z$ are the source and destination of path P, respectively. Each pair of adjacent nodes (V_k, V_{k+1}) is connected by an edge in $\mathbb{E}$, $1 \leq k < Z$. We use $IV(P) \in \mathbb{R}^{Z \times D}$ to represent the concatenation of the node feature vectors of the nodes in path P. We call $IV(P_i)$ the initial view of path P_i.

Problem Definition. Given a set of path $\mathbb{P}$ in graph G, *Path Representation Learning (PRL)* aims at learning a path representation vector $p_i \in \mathbb{R}^{D'}$ for each path $P_i \in \mathbb{P}$. Formally, *PRL* learns a path encoder PE_ψ that takes as input the initial view $IV(P_i)$ of path P_i, i.e., the node features of the nodes in path P_i, and outputs its path representation vector p_i.

$$PE_\psi : \mathbb{R}^{Z \times D} \to \mathbb{R}^{D'}, \quad (1)$$

where ψ indicates the learnable parameters for the path encoder, e.g., weights in a neural network, Z is the length of path P_i, and $D' \ll Z \times D$ is an integer indicating the dimension of the learned path representation vector p_i.

The learned path representation vectors are supposed to support a variety of downstream tasks, e.g., path ranking and path travel time estimation.

4 Path InfoMax

Figure 1 offers an overview of the proposed framework Path InfoMax (*PIM*). *PIM* employs contrastive learning, specifically mutual information maximization, to train the path encoder to produce path representations without requiring task-specific labels.

The path encoder takes as input the initial view of an input path and outputs its path representation (cf. Sec. 4.1). Training the path encoder is supported by a path-path discriminator and a path-node discriminator using negative samples. To this end, we first introduce the curriculum negative sampling strategy to generate negative paths (cf. Sec. 4.2). Then, the path-path discriminator guides the path encoder to produce path representations such that the path representations of input paths can be distinguished from the path representations of negative paths (cf. Sec. 4.3). In addition, the path-node discriminator guides the path encoder to produce path representations such that the path representations of input paths can be distinguished from the node features of the nodes that only appear in the negative paths (cf. Sec. 4.4). Finally, we discuss the final training objectives of *PIM*.

4.1 Path Encoder

Since a path consists of a sequence of nodes, we use a model that is able to encode sequential data, e.g., a recurrent neural network [Hochreiter and Schmidhuber, 1997; Cho *et al.*, 2014] or a Transformer [Vaswani *et al.*, 2017] as the path encoder PE_ψ, where ψ represents the parameters to be learned for the path encoder.

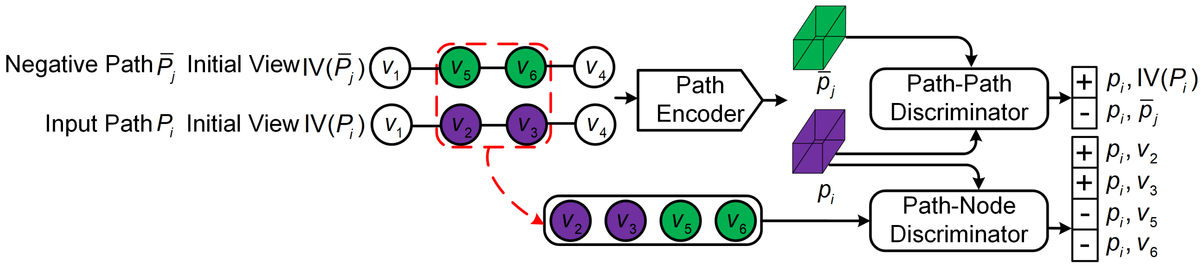


Figure 1. *PIM* Overview. The Path Encoder takes as input the initial view $IV(P_i)$ of input path P_i and the initial view $IV(\bar{P}_j)$ of negative path $\bar{P}_j$, and returns their representations p_i and $\bar{p}_j$, respectively. The Path-Path Discriminator takes as input a pair of path representations and decides whether they are from the same path. A positive pair, e.g., $(p_i, IV(P_i))$, refers to two different representation views of the same input path P_i. A negative pair, e.g., $(p_i, \bar{p}_j)$, refers to the path representations of an input path vs. its negative path. The Path-Node Discriminator takes as input a (input path representation, node feature vector) pair and decides whether the node is from the input path. A positive pair, e.g., (p_i, v_2), represents the path representation of P_i and a node feature vector of node v_2 that only appears in P_i. A negative pair, e.g., (p_i, v_5), represents the path representation of the input path and a node feature vector of node v_5 that only appears in the negative path.

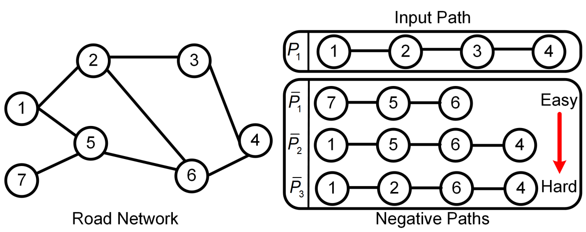


Figure 2. Curriculum Negative Sampling.

Given a path $P_i = \langle V_1, V_2, \ldots, V_Z \rangle$, we use its initial view $IV(P_i) \in \mathbb{R}^{Z \times D}$ as the input to the path encoder, which returns its path representation vector $p_i \in \mathbb{R}^{D'}$.

4.2 Curriculum Negative Sampling

Motivated by curriculum learning [Bengio *et al.*, 2009], we propose a curriculum negative sampling method to generate negative samples. The idea behind curriculum learning is that we start to train a model with easier samples first, and then gradually increase the difficulty levels. In our setting, we first generate negative paths that are different from the input path, e.g., paths without any overlapping nodes with the input path. In this case, it can be easy to train a path encoder that returns distinguishable representations of the input path and the negative paths. Then, we gradually generate negative paths that are increasingly similar to the input path, e.g., sharing the same source and destination with the input path and with increasingly overlapping nodes. This makes more difficult for the path encoder to generate distinguishable path representations. Figure 2 shows three negative paths $\bar{P}_1$, $\bar{P}_2$, and $\bar{P}_3$ with increasingly difficulties for input path P_1, along with the underlying road network graph.

Specifically, for each input path P_1, we first randomly select a path from the path set $\mathbb{P}$ as the first negative path. Next, we use the source and the destination of P_1 as the input to call the top-k diversified shortest path algorithm [Liu *et al.*, 2018] to generate paths that share the same source and destination of P_1. This algorithm allows us to set different diversity thresholds, enabling us to generate negative paths with different overlapping nodes with the input path.

4.3 Global Mutual Information Maximization

We proceed to the learning of the path encoder using the negative paths. We first consider a global view of the path representations. We expect that the learned path representations are distinguishable from the path representations of the negative paths.

To this end, we first construct negative and positive pairs for training a *path-path discriminator* $D^{PP}_{\omega_1}$. In a negative pair $\langle (p_i, \bar{p}_j), - \rangle$, p_i and $\bar{p}_j$ represent the path representations of input path P_i and a negative path $\bar{P}_j$, respectively, which are both returned by the path encoder PE_ψ. In a positive pair $\langle (p_i, IV(P_i)), + \rangle$, p_i is still the path representations of input path P_i returned by the path encoder and $IV(P_i)$ is the initial view of path P_i (cf. Section 4.1). Here, p_i and $IV(P_i)$ represent two different views, i.e., a view from the path encoder vs. a view from the node features, of the same input path P_i. Figure 1 shows examples of a negative and a positive pair.

Next, we use mutual information maximization to train the path-path discriminator $D^{PP}_{\omega_1}$ such that it is able to make a binary classification on the negative vs. positive pairs. Specifically, we aim at maximizing the estimated mutual information (MI) over the positive and negative pairs.

$$\arg\max_{\psi,\omega_1} \sum_{P_i \in \mathbb{P}} I_{\psi,\omega_1}(p_i, \mathbb{NP}_i),$$

where $I_{\psi,\omega_1}(\cdot,\cdot)$ is the MI estimator modeled by the path-path discriminator $D^{PP}_{\omega_1}$ that is parameterized by parameters ω_1 and the path encoder PE_ψ that is parameterized by parameters ψ. Path P_i is an input path from $\mathbb{P}$ and p_i is its path representation returned by the path encoder. $\mathbb{NP}_i$ includes the negative paths of P_i. Following [Velickovic *et al.*, 2019; Hjelm *et al.*, 2019], we use a noise-contrastive type objective with a standard binary cross-entropy loss on the positive pairs and the negative pairs, as shown in Equation 2.

$$\mathcal{I}_{\psi,\omega_1}(p_i, \mathbb{NP}_i) := \frac{1}{1+|\mathbb{NP}_i|}\Big(\mathbb{E}_{\mathbb{P}}\left[\log D^{PP}_{\omega_1}(p_i, IV(P_i))\right] + \sum_{\bar{P}_j \in \mathbb{NP}_i} \mathbb{E}_{\mathbb{NP}_i}\left[\log\left(1 - D^{PP}_{\omega_1}(p_i, \bar{p}_j)\right)\right]\Big) \tag{2}$$

Here, we use $\mathbb{E}_{\mathbb{P}}$ and $\mathbb{E}_{\mathbb{NP}_i}$ to denote expectations w.r.t. the empirical probability distribution of the input paths and the negative paths. Note that p_i and $\bar{p}_j$ are the path representations returned by the path encoder PE_ψ. Thus, maximizing the MI estimator enables the training of both the path encoder (i.e., parameters ψ) and the path-path discriminator (i.e., parameters ω_1).

4.4 Local Mutual Information Maximization

We proceed to consider a local view of the path representations. We expect that the learned path representations are distinguishable from the node feature vectors of the nodes from input vs. negative paths. This is particularly important when distinguishing two paths with significant overlapping nodes. We introduce a positive node set $\mathbb{X}_i$ that includes nodes appearing only in the input path P_i but not the negative paths and a negative node set $\mathbb{Y}_i$ that includes nodes appearing only in the negative paths but not the input path P_i. We then construct negative and positive pairs for training a *path-node discriminator* $D^{PN}_{\omega_2}$. In a negative pair $\langle (p_i, v_j), - \rangle$, p_i represents the path representations of input path P_i, returned by the path encoder PE_ψ; v_j represents the node feature vector of a negative node $V_j \in \mathbb{Y}_i$. Similarly, in a positive pair $\langle (p_i, v_k), + \rangle$, v_k represents the node feature vector of a positive node $V_k \in \mathbb{X}_i$. Figure 1 shows examples of two negative and two positive such pairs for the path-node discriminator.

Similar to the path-path discriminator training, we also employ mutual information maximization to train the path-node discriminator $D^{PN}_{\omega_2}$. In particular, we have

$$\arg\max_{\psi,\omega_2} \sum_{P_i \in \mathbb{P}} I_{\psi,\omega_2}(p_i, \mathbb{X}_i \cup \mathbb{Y}_i),$$

where I_{ψ,ω_2} is the MI estimator modeled by the path-node discriminator $D^{PN}_{\omega_2}$ that is parameterized by parameters ω_2 and the path encoder PE_ψ that is parameterized by parameters ψ. We use a noise-contrastive with a BCE loss, similar to Equation 2, to compute $I_{\psi,\omega_2}(p_i, \mathbb{X} \cup \mathbb{Y})$ as follows.

$$\begin{aligned} I_{\psi,\omega_2}(p_i, \mathbb{X}_i \cup \mathbb{Y}_i) := &\frac{1}{|\mathbb{X}_i \cup \mathbb{Y}_i|} \Big(\sum_{v_k \in \mathbb{X}_i} \mathbb{E}_{\mathbb{X}_i} \left[\log \mathcal{D}^{PN}_{\omega_2}(p_i, v_k) \right] + \\ &\sum_{v_j \in \mathbb{Y}_i} \mathbb{E}_{\mathbb{Y}_i} \left[\log \left(1 - D^{PN}_{\omega_2}(p_i, v_j) \right) \right] \Big) \end{aligned} \tag{3}$$

4.5 Maximization of *PIM*

We combine both the global and local mutual information maximization when training the final *PIM* model, see below.

$$\arg\max_{\psi,\omega_1,\omega_2} \sum_{P_i \in \mathbb{P}} \left(I_{\psi,\omega_1}(p_i, \mathbb{NP}_i) + I_{\psi,\omega_2}(p_i, \mathbb{X}_i \cup \mathbb{Y}_i) \right).$$

5 Experiments

We conduct experiments to investigate the effectiveness of the proposed unsupervised path representation learning framework *PIM* on two downstream tasks using two data sets. In addition, we also demonstrate that *PIM* is able to use as a pre-training method to enhance supervised path representation learning.

5.1 Experimental Setup

Road Network and Paths

We obtain two road network graphs from OpenStreetMap. The first is from Aalborg, Denmark, consisting of 8,893 nodes and 10,045 edges. The second is from Harbin, China, consisting of 5,796 nodes and 8,498 edges. We also obtain two substantial GPS trajectory data sets on the two road networks. We consider 52,494 paths in the Aalborg network and 37,079 paths in the Harbin network.

Downstream Tasks

Path Travel Time Estimation. Each path is associated with its travel time (seconds) obtained from trajectories. We aim at building a regression model to estimate the travel time of paths. We evaluate the accuracy of the estimations by Mean Absolute Error (**MAE**), Mean Absolute Relative Error (**MARE**) and Mean Absolute Percentage Error (**MAPE**).

Path Ranking. Given a set of paths, which often share the same source and destination, each path is associated with a ranking score in range $[0, 1]$. The ranking scores are obtained with the help of trajectories by following an existing study [Yang *et al.*, 2020]. In path ranking, we aim at building a regression model to estimate the ranking scores of the paths. To evaluate the accuracy of the estimated ranking scores, we not only report the **MAE** of the estimated ranking scores but also use Kendall rank correlation coefficient (denoted by τ) and Spearman's rank correlation coefficient (denoted by ρ) to measure the consistency between the ranking derived by the estimated ranking scores vs. the ranking derived by the ground truth ranking scores. Smaller **MAE** and higher τ and ρ values indicate higher accuracy.

Baselines

We compare *PIM* with seven baseline methods.

- **Node2vec** [Grover and Leskovec, 2016], Deep Graph InfoMax (**DGI**) [Velickovic *et al.*, 2019], Graphical Mutual Information Maximization (**GMI**) [Peng *et al.*, 2020] are three unsupervised node representation learning models, which output the node representation for each node in a graph. We use the average of the node representations of the nodes in a path to get the path representation of the path. We also consider using concatenation instead of average, but resulting worse accuracy.
- **Memory Bank (MB)** [Wu *et al.*, 2018] is an unsupervised learning approach to obtain representations from high-dimensional data. It uses a memory bank to achieve the negative samples from current batch to train an encoder, then gets the representation based on contrastive loss. We re-implement **MB** with an LSTM encoder to better capture the sequential information to get the path representations.
- **InfoGraph** [Sun *et al.*, 2020] is an unsupervised whole graph representation learning model. Here, we treat a path as a graph to learn the path representation.
- **BERT** [Devlin *et al.*, 2019] is an unsupervised language representation learning model. To enable training, we (1) treat a path as a sentence and mask some nodes in the path; and (2) split a path P into two sub-paths

Method	Aalborg						Harbin					
	Travel Time Estimation			Path Ranking			Travel Time Estimation			Path Ranking		
	MAE	MARE	MAPE	MAE	τ	ρ	MAE	MARE	MAPE	MAE	τ	ρ
Node2vec	121.43	0.27	31.04	0.18	0.66	0.70	258.91	0.22	23.17	0.15	0.70	0.72
DGI	192.63	0.42	82.44	0.54	0.49	0.52	528.71	0.39	86.53	0.21	0.59	0.60
GMI	136.58	0.30	50.81	0.23	0.58	0.61	979.68	0.73	192.45	0.24	0.55	0.56
MB	243.97	0.53	84.17	0.35	0.34	0.38	533.41	0.40	86.01	0.27	0.31	0.34
BERT	254.17	0.54	61.61	0.36	0.38	0.39	514.95	0.57	49.80	0.28	0.45	0.46
InfoGraph	132.28	0.29	39.47	0.17	0.69	0.73	391.45	0.44	44.60	0.29	0.68	0.72
PIM	**76.10**	**0.16**	**17.28**	**0.12**	**0.72**	**0.76**	**125.76**	**0.14**	**13.73**	**0.11**	**0.75**	**0.79**

Table 1. Overall Accuracy on Travel Time Estimation and Ranking Score Estimation.

P_1 and P_2, and consider (P_1, P_2) as a valid Q&A pair and (P_2, P_1) as an invalid Q&A pair because the former keeps a meaningful order while the latter does not.

- **PathRank** [Yang *et al.*, 2020] is a *supervised* path representation learning model based on GRU. **PathRank** takes into account the labels from a specific downstream task to obtain path representations.

Among these baselines, *Node2vec*, *DGI*, *GMI*, *MB*, *InfoGraph*, and *BERT* are *unsupervised learning* approaches, which do not employ labels from specific downstream tasks to produce path representations. In contrast, *PathRank* is a *supervised learning* approach, where it employ labels from specific downstream tasks to produce path representations, meaning that the obtained path representations are different when using labels from different downstream tasks.

Regression Model

For all unsupervised learning approaches, we first obtain a task-independent path representation and then apply a regression model to solve different downstream tasks using task-specific labels. In the experiments, we choose *Gaussian Process Regressor* (*GPR*) to make travel time and ranking score estimation. We randomly choose 85%, 10%, and 5% of the paths as the training, validation, and test sets.

Implementation Details

We use an LSTM as the path encoder. We use *node2vec* [Grover and Leskovec, 2016], an unsupervised node representation learning method, to obtain a 128 dimensional node feature vector for each node, i.e., $D = 128$. We set the path representation size $D' = 128$. In the curriculum negative sampling, for each input path, we generate four negative paths—the first two paths are randomly selected from $\mathbb{P}$ and the third and the fourth paths are two paths returned by the top-k diversified shortest paths with different overlapping nodes with the input path. We use Adam [Kingma and Ba, 2015] for optimization with learning rate of 0.001. All algorithms are implemented in Pytorch 1.7.1. We conduct experiments on Ubuntu 18.04.5 LTS, with 40 Intel(R) Xeon(R) Gold 5215 CPUs @ 2.50GHz and four Quadro RTX 8000 GPU cards. Code is available at https://github.com/Sean-Bin-Yang/Path-InfoMax.git.

5.2 Experimental Results

Overall accuracy on both downstream tasks

Table 1 shows the results on travel time and ranking score estimation. *PIM* consistently outperforms all baselines on both tasks and on both data sets. *Node2vec*, *DGI*, and *GMI* fail to capture the dependencies among node feature vectors in paths. In contrast, *PIM* considers such dependencies by using the LSTM based path encoder. In addition, the two discriminators further improve the accuracy.

InfoGraph implicitly considers node feature vector sequences. However, the discriminator in *InfoGraph* only considers the local view. In addition, *InfoGraph* considers other paths in the same batch as negative samples, whereas *PIM* employs curriculum negative sampling to generate negative samples. *PIM* outperforms *InfoGraph* suggests that the proposed curriculum negative sampling and jointly consider both local and global views are effective.

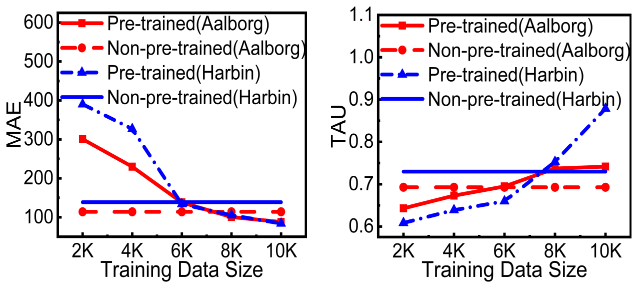
Although *MB* and *BERT* also capture dependencies among the node feature vectors in paths, such methods only achieve relatively poor accuracy. This is because *MB* often requires large amount of negative samples (e.g., more than 256), which is not feasible in our setting. Although the unsupervised training strategy in *BERT* works well for NLP, it does not fit our problem setting on learning path representations.

Using *PIM* as a Pre-training Method

In this experiment, we consider *PIM* as a pre-training method for the supervised method *PathRank*. *PathRank* employs an GRU that takes as input node feature vectors in a path and predicts travel time or ranking scores. To use *PIM* as a pre-training method for *PathRank*, we use a GRU based path encoder. Then, we first train *PIM* in an unsupervsied manner, and then use the learned parameters in the GRU path encoder to initialize the GRU in *PathRank*. Finally, we use the labelled training paths to fine tune *PathRank*.

Figure 3 shows the travel time estimation performance of *PathRank* with vs. without pre-training on both data sets. When not using pre-training, we train *PathRank* using 10K labelled training paths. We observe that: (1) when using pre-training, we are able to achieve the same accuracy of the non-pre-training *PathRank* using less labelled training paths, e.g., ca. 7K for Aalborg and 6K for Harbin. (2) when using 10K labelled training paths, the pre-training *PathRank* achieves higher accuracy than the non-pre-training *PathRank*. We observe similar results on the other task of path ranking, suggesting that *PIM* can be used as a pre-training method to enhance supervised methods.

5.3 Ablation Studies



(a) Travel Time Estimation (b) Path Ranking

Figure 3. Effects of Pre-training.

	Travel Time Estimation			Path Ranking		
	MAE	MARE	MAPE	MAE	τ	ρ
Global	237.92	0.51	85.88	0.34	0.22	0.25
Local	118.03	0.25	26.20	0.14	0.70	0.74
Joint	**76.10**	**0.16**	**17.28**	**0.12**	**0.72**	**0.76**

Table 2. Effects of Local and Global MI Maximization, Aalborg.

Impact of Local and Global MI Maximization

We investigate the impact of jointly using both path-path and path-node discriminators to consider both the local and global MI maximization. We consider two variants of *PIM* where (1) we only use the path-path discriminator to maximize the global MI and (2) we only use the path-node discriminator to maximize the local MI. Table 4 shows that jointly maximizing both the local and global MI achieves the best accuracy, which justifies our design choices of using both the path-path and path-node discriminators.

Impact of Curriculum Negative Sample Strategy

To investigate the effectiveness of the proposed curriculum negative sample strategy, we compare it with the following two strategies.

1. Random only: it randomly selects paths from $\mathbb{P}$.
2. Top-k only: it employs the top-k diversified shortest path algorithms to generate negative paths sharing the same origin and destination with the input path with different overlapping nodes.

To make a fair comparison, we use each strategy to generate the same number of negative paths, i.e., 4. Table 3 shows that the top-k only strategy is better than random only, suggesting that it is important to distinguish the representations of input paths vs. paths sharing the same origin and destination. The proposed curriculum negative sampling strategy achieves the best accuracy, suggesting that training *PIM* from easy to hard negative paths help further improves accuracy.

Impact of Negative Path Numbers

We investigate the impact of using different numbers of negative paths. We vary the number of negative paths K from 1, 2, 3, to 4. Recall that when using curriculum negative sampling, the first two paths are from $\mathbb{P}$ and the last two paths are from the top-k diversified shortest path finding algorithm. Table 4 suggests that when using more negative paths, the accuracy improves. The accuracy improvements from 2 to 3 is the largest, suggesting that the top-k algorithm is very effective on generating high quality negative paths.

	Travel Time Estimation			Path Ranking		
	MAE	MARE	MAPE	MAE	τ	ρ
Rand.	101.16	0.22	23.51	0.14	0.65	0.69
Top-k	100.87	0.22	22.31	0.13	0.72	0.75
Curr.	**76.10**	**0.16**	**17.28**	**0.12**	**0.72**	**0.76**

Table 3. Effects of Curriculum Negative Sample Strategy, Aalborg.

	Travel Time Estimation			Path Ranking		
	MAE	MARE	MAPE	MAE	τ	ρ
K=1	119.77	0.29	32.91	0.19	0.58	0.63
K=2	107.46	0.26	29.22	0.18	0.59	0.63
K=3	87.58	0.19	20.00	0.12	0.71	0.74
K=4	**76.10**	**0.16**	**17.28**	**0.12**	**0.72**	**0.76**

Table 4. Effects of Negative Path Numbers, Aalborg.

Posi. Nods.	Travel Time Estimation			Path Ranking		
	MAE	MARE	MAPE	MAE	τ	ρ
20%	114.31	0.25	24.92	0.20	0.65	0.70
40%	111.33	0.24	24.08	0.16	0.66	0.70
60%	104.57	0.23	22.94	0.14	0.68	0.71
80%	101.31	0.23	22.56	0.13	0.68	0.72
100%	**76.10**	**0.16**	**17.28**	**0.12**	**0.72**	**0.76**
Neg. Nods.	**Travel Time Estimation**			**Path Ranking**		
	MAE	MARE	MAPE	MAE	τ	ρ
20%	130.90	0.29	28.21	0.19	0.60	0.65
40%	110.86	0.24	25.30	0.15	0.67	0.70
60%	105.70	0.23	24.01	0.13	0.67	0.71
80%	102.80	0.22	23.35	0.13	0.68	0.72
100%	**76.10**	**0.16**	**17.28**	**0.12**	**0.72**	**0.76**

Table 5. Effects of Positive / Negative Nodes, Aalborg.

Impact of Positive/Negative Nodes in local MI

To study the impact of positive and negative nodes, we consider cases where we only use 20%, 40%, 60%, 80% of positive or negative nodes. Table 5 shows that the accuracy increases when using more less positive and negative nodes.

6 Conclusions

We study unsupervised path representation learning without using task-specific labels. We propose a novel contrastive learning framework Path InfoMax (*PIM*), including a curriculum negative sampling strategy to generate a small number of negative paths and a training mechanism that jointly learns distinguishable path representations from both a global and a local view. Finally, we conduct experiments on two datasets with two downstream tasks. Experimental results show that *PIM* outperforms other unsupervised methods and, as a pre-training method, *PIM* is able to enhance supervised path representation learning.

Acknowledgments

This work was supported by Independent Research Fund Denmark under agreements 8022-00246B and 8048-00038B, the VILLUM FONDEN under agreement 34328, and the Innovation Fund Denmark centre, DIREC.

References

[Bengio *et al.*, 2009] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *ICML*, pages 41–48, 2009.

[Cho *et al.*, 2014] Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. In *EMNLP*, pages 103–111, 2014.

[Cirstea *et al.*, 2021] Razvan-Gabriel Cirstea, Tung Kieu, Chenjuan Guo, Bin Yang, and Sinno Jialin Pan. Enhancenet: Plugin neural networks for enhancing correlated time series forecasting. In *ICDE*, pages 1739–1750, 2021.

[Devlin *et al.*, 2019] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, pages 4171–4186, 2019.

[Ding *et al.*, 2020] Jingtao Ding, Yuhan Quan, Quanming Yao, Yong Li, and Depeng Jin. Simplify and robustify negative sampling for implicit collaborative filtering. In *NeurIPS*, 2020.

[Grover and Leskovec, 2016] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *KDD*, pages 855–864, 2016.

[Guo *et al.*, 2018] Chenjuan Guo, Bin Yang, Jilin Hu, and Christian S. Jensen. Learning to route with sparse trajectory sets. In *ICDE*, pages 1073–1084, 2018.

[Guo *et al.*, 2020] Chenjuan Guo, Bin Yang, Jilin Hu, Christian S. Jensen, and Lu Chen. Context-aware, preference-based vehicle routing. *VLDB J.*, 29(5):1149–1170, 2020.

[Hjelm *et al.*, 2019] R. Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Philip Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. In *ICLR*, 2019.

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

[Hu *et al.*, 2019] Jilin Hu, Chenjuan Guo, Bin Yang, and Christian S. Jensen. Stochastic weight completion for road networks using graph convolutional networks. In *ICDE*, pages 1274–1285, 2019.

[Hu *et al.*, 2020] Jilin Hu, Bin Yang, Chenjuan Guo, Christian S. Jensen, and Hui Xiong. Stochastic origin-destination matrix forecasting using dual-stage graph convolutional, recurrent neural networks. In *ICDE*, pages 1417–1428, 2020.

[Kieu *et al.*, 2018] Tung Kieu, Bin Yang, Chenjuan Guo, and Christian S. Jensen. Distinguishing trajectories from different drivers using incompletely labeled trajectories. In *CIKM*, pages 863–872, 2018.

[Kingma and Ba, 2015] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[Li *et al.*, 2017] Cheng Li, Jiaqi Ma, Xiaoxiao Guo, and Qiaozhu Mei. Deepcas: An end-to-end predictor of information cascades. In *WWW*, pages 577–586, 2017.

[Liu *et al.*, 2017] Zemin Liu, Vincent W. Zheng, Zhou Zhao, Fanwei Zhu, Kevin Chen-Chuan Chang, Minghui Wu, and Jing Ying. Semantic proximity search on heterogeneous graph by proximity embedding. In *AAAI*, pages 154–160, 2017.

[Liu *et al.*, 2018] Huiping Liu, Cheqing Jin, Bin Yang, and Aoying Zhou. Finding top-k shortest paths with diversity. *IEEE Trans. Knowl. Data Eng.*, 30(3):488–502, 2018.

[Pedersen *et al.*, 2020a] Simon Aagaard Pedersen, Bin Yang, and Christian S. Jensen. Anytime stochastic routing with hybrid learning. *PVLDB*, 13(9):1555–1567, 2020.

[Pedersen *et al.*, 2020b] Simon Aagaard Pedersen, Bin Yang, and Christian S. Jensen. Fast stochastic routing under time-varying uncertainty. *VLDB J.*, 29(4):819–839, 2020.

[Peng *et al.*, 2020] Zhen Peng, Wenbing Huang, Minnan Luo, Qinghua Zheng, Yu Rong, Tingyang Xu, and Junzhou Huang. Graph representation learning via graphical mutual information maximization. In *WWW*, pages 259–270, 2020.

[Sun *et al.*, 2020] Fan-Yun Sun, Jordan Hoffmann, Vikas Verma, and Jian Tang. Infograph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization. In *ICLR*, 2020.

[Tang *et al.*, 2015] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. LINE: large-scale information network embedding. In *WWW*, pages 1067–1077, 2015.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, pages 5998–6008, 2017.

[Velickovic *et al.*, 2019] Petar Velickovic, William Fedus, William L. Hamilton, Pietro Liò, Yoshua Bengio, and R. Devon Hjelm. Deep graph infomax. In *ICLR*, 2019.

[Wang *et al.*, 2018] Peifeng Wang, Shuangyin Li, and Rong Pan. Incorporating GAN for negative sampling in knowledge representation learning. In *AAAI*, pages 2005–2012, 2018.

[Wu *et al.*, 2018] Zhirong Wu, Yuanjun Xiong, Stella X. Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *CVPR*, pages 3733–3742, 2018.

[Yang and Yang, 2020] Sean Bin Yang and Bin Yang. Learning to rank paths in spatial networks. In *ICDE*, pages 2006–2009, 2020.

[Yang *et al.*, 2020] Sean Bin Yang, Chenjuan Guo, and Bin Yang. Context-aware path ranking in road networks. *IEEE TKDE*, 2020.