CONTRASTIVE REPRESENTATION LEARNING FOR MULTI-SCALE SPATIAL SCENES

Anonymous authors
Paper under double-blind review

ABSTRACT

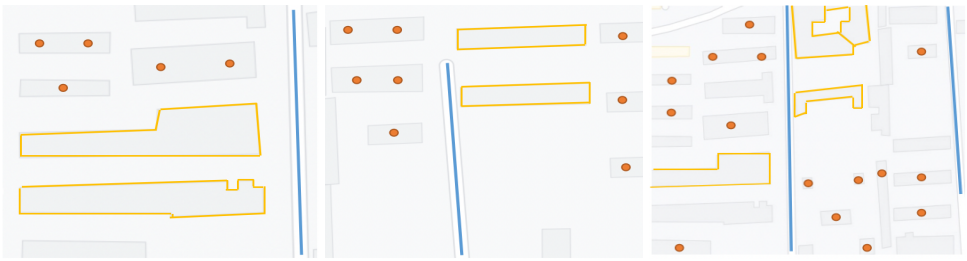
Spatial scenes, which are composed by spatial objects and their spatial relations, are the basis of geographic information retrieval, spatial cognition, and spatial search. Despite the wide usage of spatial scenes, representation learning on spatial scenes that contain complex composition of spatial objects remains a challenge, since the spatial data types of geographic objects (e.g., points, polylines, and polygons) and the geographical scales vary across different spatial scenes. Inspired by recently proposed multi-scale location encoding models such as Space2Vec, we propose a multi-scale spatial scene encoding model called Scene2Vec to solve these representational challenges. In Scene2Vec, a location encoder is used to model the spatial relationships among spatial objects and a feature encoder is used for objects' semantic feature encoding. A scene encoder is developed to integrate the representations of spatial objects into a single scene embedding. Moreover, we propose a spatial scene augmentation method to sample additional points based on the shapes of polyline/polygon-based spatial objects in all scales of spatial scenes. The whole model is trained in a self-supervised manner with a contrastive loss. We conduct experiments on real world datasets for spatial scene retrieval task 1) purely based on points, e.g., points of interest (POIs), and 2) based on multi-structured spatial objects. Results show that Scene2Vec outperforms well-established encoding methods such as Space2Vec and multi-layer perceptrons due to the advantages of the integrated multi-scale representations and the proposed spatial scene augmentation method. Moreover, detailed analysis shows that Scene2Vec has the ability to generate representations of all the three types of spatial objects in a multi-scale manner.

1 INTRODUCTION

Imagine you are wondering around to find a specific place in the real world. Purely based on your memory or expectation, you know the (partially) surroundings/spatial context around it. This is called mental map. Mental map (Graham, 1976; Purchase et al., 2006) is a concept in behavioral geography, which is the basis of human spatial cognition. How to represent real-world environments with mental maps through scientific computational models is a major challege in the spatial cognition ability of intelligent robots, such as autopilot (Badue et al., 2021). One possible solution is through spatial scene encoding (Guo et al., 2022). Spatial scene encoding refers to a neural network architecture that encode a combination of spatial objects in a 2D or 3D Euclidean space into a high dimensional embedding for downstream spatial scene related tasks such as geographic information retrieval, spatial cognition, and spatial search. Spatial scene encoding can be incorporated into state-of-the-art (SOTA) machine learning models for many tasks to make them spatially explicit (Janowicz et al., 2020; Mai et al., 2019). And it has huge application potential in geospatial related domains such as urban planning (Oliveira & Pinho, 2010), land cover classification (Mallet et al., 2011), scene restoration (Ali et al., 2021), etc.

Existing models often focus on encoding specific information of geographic objects while ignoring other aspects of geographic information. For example, Place2Vec (Yan et al., 2017) converts the coordinates of points of interest (POIs) into spatially collocated POI pairs within certain distance bins, and does not preserve information about the (cardinal) direction between points. Mot2vec (Crivellari & Beinat, 2019) provides a meaningful representation of locations, based on the motion behavior of users, defining a direct way of comparing locations' connectivity and providing analogous similar-

ity distributions for places of the same type. Tile2Vec (Jean et al., 2019) utilizes an unsupervised learning objective that makes spatially nearby image tiles similar in the image embedding space. However, both Mot2Vec and Tile2Vec ignore the actual geographical coordinates and convert them into visiting sequences (Mot2Vec) or qualitative distance measures (e.g., nearby or distant image tiles by Tile2Vec). There lacks a general model to directly encode the spatial configuration of spatial scene data which is composed of different types of spatial objects.



(a) (b) (c)

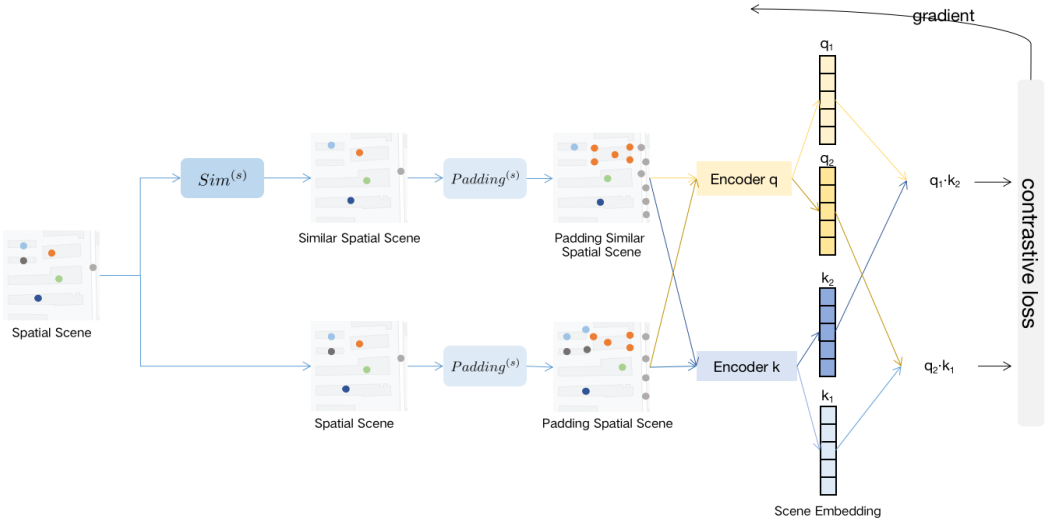
Figure 1: The challenges of modeling spatial scenes with distributions of differeant types of spatial objects. The marks in the (a)(b)(c) indicate the spatial objects. Specifically, the orange points indicate spatial objects represented as points while the blue lines denote the polyline data, the yellow polygons indicate the polygon data. The spatial scale of (a) and (b) are the same while the scale of (c) is smaller. The numbers of spatial objects in these three spatial scenes are different from each other. To achieve multi-scale representation, Scene2Vec concatenates the spatial scene features of K scales and three different spatial data types as the first layer of a location encoding model. The whole model is trained in a unsupervised manner with a contrastuve loss.

The key challenge in designing spatial scene encoding models is the problem of modeling complex composition of various number of spatial objects represented in different geometric types in spatial scenes (see an example in Figure 1). More specifically, there are three challenges for spatial scene representation learning. First, the map scales of spatial scenes vary. For instance, the map scale in Figure 1c is smaller than those of Figure 1a and 1b. Second, the number of spatial objects that make up a spatial scene also vary. Last but not least, a spatial scene can be composed of spatial objects that are represented as different geometric types such as points, polylines, and polygons. To overcome these challenges, we aim to develop a general-purpose representation learning model for spatial scene with the following characteristics: 1) A multi-scale encoding method for arbitrary spatial scale of spatial scenes; 2) A permutation invarant architecture which can handle arbitrary number of spatial objects; 3) A encoding method which can seemlessly handle different geometry types including points, polylines, and polygons; 4) An *inductive learning model* – the pretrained spatial scene encoder can be readily utilized on unseen dataset.

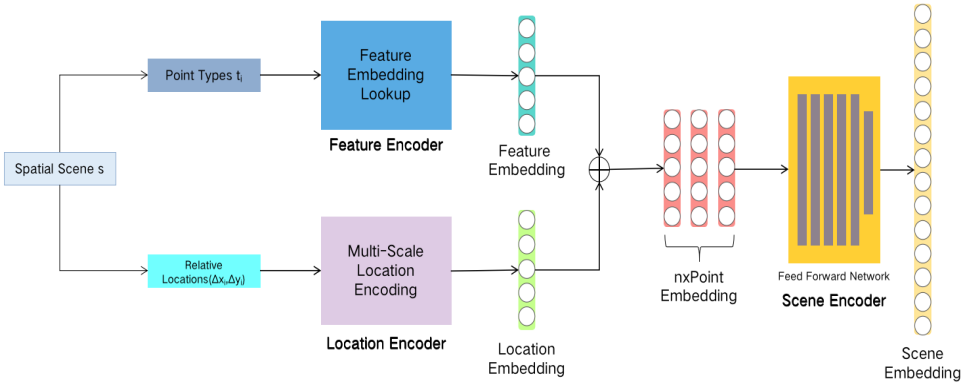
Several recent studies (Gao et al., 2018; Mai et al., 2019; 2020; Dang et al., 2021) present multi-scale location encoding models which show promising results on different geospatial tasks including POI type classification, geo-aware image classification, geographic question answering, and path integration. These models inspire us to develop neural architectures to capture the relative spatial relations among spatial objects. Moreover, approaches related to contrastive loss (Hadsell et al., 2006) play a critical role in unsupervised representation learning, whose core idea is to learn metric similarity between samples. Coincidentally, spatial scene similarity plays a critical role in geoinformatic research (e.g., spatial prediction (Zhu et al., 2018)) – spatial scene similarity can be considered as the criterion to validate the location representation learning model (Guo et al., 2022). Inspired by these researches, we propose a multi-scale spatial scene encoding method to encode spatial objects represented in different geometric types in a spatial scene, their features, as well as their relative spatial relations into an embedding space. We use the momentum contrastive learning-based approach (He et al., 2020) to train our model in an unsupervised learning manner. Correspondingly, we propose a specialized spatial scene augmentation method which can do augmentation on spatial scenes with various scales while preserving their similarity with original scenes. This augmentation method do point sampling based on the shapes of the polyline and polygon type spatial objects in a spatial scene such that it adds randomness (and increase robustness) to the framework. In addition, the combination of the augmentation method and the spatial scene encoder can be regarded as a potential way

for polyline and polygon[1] encoding. Figure 2 shows the methodological framework of spatial scene encoding we proposed. In summary, the main contributions of our work are four-fold:

1. We propose a general-purpose spatial scene encoder framework, Scene2Vec, which uses multi-scale location encoding methods to model relative spatial relations among spatial objects in a spatial scene.
2. We propose a novel spatial scene augmentation method that can generate spatial scenes in arbitrary scales as positive samples by sampling points based on the shape of polyline/polygon type spatial objects.
3. We use a contrastive learning framework to train the proposed model in an unsupervised manner. This encoder combines with the augmentation method can be considered as an encoder for polylines and polygons.
4. We conduct experiments on real world datasets for the spatial scene similarity task. Results show that Scene2Vec outperforms well-established encoding methods such as Space2Vec, multi-layer perceptrons, and can serve as a pretrained model in multiple geospatial tasks.



(a)



(b)

Figure 2: The methodological framework of our proposed spatial scene encoding. (a) Overall workflow of our method including augmenting, encoding and contrastive representation learning; (b) An illuatration of Scen2Vec encoder.

[1]In GIS and spatial analysis, A tree, for instance, be modeled by a point, while street would be represented as a polyline and a shopping mall could be modeled as a polygon.

2 RELATED WORK

Theoretical research has been done on spatial scene analysis in the field of geography. Tobler's First Law of Geography (TFL) (Tobler, 1970) gives a general statement of the spatial correlation of geographic phenomena, which provides an theoretical basis for quantitative computation for the expression, analysis and application of geographic information. Frank et al. (1995) further introduced spatial relationships containing topological, direction and distance relations and Bruns & Egenhofer (1996) defined the similarity of spatial scenes based on these relations. Several approaches (Egenhofer, 1997; Haarslev & Wessel, 1997; Winter, 2000; Caduff & Egenhofer, 2005; Liang et al., 2005) have successfully combined mental map (described in Section 1) with real world data by using sketch maps based on the above relationship principles. Zhu et al. (2018) then described the cross-correlation of spatial phenomena based on the spatial similarity and adopted it to implement spatial prediction. All these research focus on the conceptualization of spatial scene similarity and its importance for spatial analysis in the geography and GIS domain. We hope to introduce these geographic knowledge to geospatial artificial intelligence (GeoAI) to better develop the geoinformatics.

Recently, several studies (Yan et al., 2017; Yao et al., 2017; Mai et al., 2019) proposed representation learning models to encode points to high-dimensional vectors and applied them in various downstream tasks such as POI classification and urban land use classification. However, these models purely focus on the single point encoding. Other works (Xu et al., 2018; Rao et al., 2020) used the language model-like architectures to encode a polyline by regarding a series of locations as a sentence for trajectory prediction. The shortcoming of these models is they do not meet the characteristics of the permutation invariant (Qi et al., 2017) of polylines due to the constraints of language model. Moreover, polygon encoder research are still at an early stage because of the difficulties to handle topological relationships. Mai et al. (2020) presented a Space2Vec-based geographic entity encoding model as the first step towards polygon encoding by uniformly sampling a location from within the bounding box of a geographic entity and feeding it to Space2Vec location encoder. However, this model still cannot handle fine-grained polygon geometries. In contrast, we propose a representation model to encode all kinds of spatial objects such as points, polylines, and polygons.

3 PROBLEM FORMULATION

We represent a polyline or a polygon as a series of points in our dataset where $polyline(polygon) = (p_1, ..., p_i, ..., p_m)$, in which $pi = (x_i, v)$ is associated with a location x_i and all the points have the same attributes v. That is, we can use a point set to represent all the three spatial types in a spatial scene. So *representation of spatial scenes* can be formulate as follows. Given a set of spatial scenes $S = s_i$,in L-D (L=2,3.In this paper,L=2) space, each scene $s_i = [c_i, (p_1, ..., p_i, ..., p_n)]$ is associated with a spatial scene description location c_i and a set of spatial objects (i.e., POIs) $p = (p_1, ..., p_i, ..., p_n)$, in which $pi = (x_i, v_i)$ is associated with a location x_i and attributes v_i (i.e. POI type name). We define function $f_{s,\theta}(\boldsymbol{x}) : \mathbb{R}^L \rightarrow \mathbb{R}^d$, which is parameterized by θ and maps any coordinate of a spatial object in a spatial scene to a vector representation of d dimension. The function $f_{s,\theta}(\boldsymbol{x})$ encodes the distribution of spatial objects in a spatial scene and can give a representation of any scale of spatial scene. Attributes (e.g. POI types such as *Market*) and coordinates of spatial objects can be seen as analogies to words and word positions in commonly used language embedding models such as Transformer (Vaswani et al., 2017).

4 METHOD

We propose the *representation learning of spatial scenes* (defined in Section 3) with an encoder architecture combined with a novel augmentation method suitable for spatial scene data:

1. Given a spatial scene $s_i = [c_i, (p_1, ..., p_i, ..., p_n)]$, $Aug^{(s)}$ firstly uses $Sim^{(s)}$ to create a similar spatial scene $s'_i = [c_i, (p'_1, ..., p'_i, ..., p'_m)]$ where m is not necessarily equal to n. Then we use $Padding^{(s)}$ to pad both s_i and s'_i to a fixed length, $s_{i(pad)} = [c_i, (p_1, ..., p_i, ..., p_l)]$ and $s'_{i(pad)} = [c_i, (p'_1, ..., p'_i, ..., p'_l)]$, where l is a hyper-parameter.

2. According to the $s_{i(pad)} = [c_i, (p_1, ..., p_i, ..., p_l)]$, $p_i = (x_i, v_i)$ (we use the same operation to $s'_{i(pad)}$), given in the first step, we firstly use a **point location encoder** $Enc^{(x)}()$ to encode relative location $r_i = x_i - c_i$ into a location embedding $\boldsymbol{e}_i[r_i] \in \mathbb{R}^{m^{(r)}}$ and a **point feature encoder** $Enc^{(v)}()$ to encoder its feature into a feature embedding $\boldsymbol{e}_i[v_i] \in \mathbb{R}^{m^{(v)}}$. $\boldsymbol{e}_i = \boldsymbol{e}_i[r_i] + \boldsymbol{e}_i[v_i] \in \mathbb{R}^m$ is the full representation of point $p_i \in P$, where $m = m^{(r)} = m^{(v)}$. Then we use **scene encoder** to integrate $\boldsymbol{e} = [\boldsymbol{e}_1, ..., \boldsymbol{e}_i, ..., \boldsymbol{e}_l] \in \mathbb{R}^{l \times m}$ to $\boldsymbol{s} \in \mathbb{R}^d$.

4.1 AUGMENTATION

One novelty of this work is from the spatial scene augmentation $Aug^{(s)}$. According to the contrastive learning objective, we need to create a spatial scene augmentation method. We first give the definition of *Spatial Scene Similarity*.

Definition 1. *Given two spatial scenes $s_i = [c_i, (p_1, ..., p_m)]$ and $s_j = [c_j, (p_1, ..., p_n)]$, where $p_i = (x_i, v_i)$. Let $\boldsymbol{d}_i = [x_{i1} - c_i, ..., x_{ik} - c_i]$ and $\boldsymbol{d}_j = [x_{j1} - c_j; ...; x_{jk} - c_j]$ as relative locations of k paried points in two spatial scenes. Let $\boldsymbol{f}_i = [v_{i1}; ...; v_{ik}]$ and $\boldsymbol{f}_j = [v_{j1}; ...; v_{jk}]$ as attributes of those paired points in two spatial scenes. We say s_i is similar to s_j if they satisfy:1)$\boldsymbol{d}_i$ is similar to $\boldsymbol{d}_j$ and 2)$\boldsymbol{f}_i$ is similar to $\boldsymbol{f}_j$. 3) the number of rest points in both two spatial scenes is as less as possible.*

The proof of Definition 1 can be seen in Nedas & Egenhofer (2008). Here we use *similar* to indicate the algorithms of similarity calculation (e.g.,cosine similarity). Moreover, this description about the spatial scene similarity in Definition 1 is purely based on calculations of geospatial relationships (Nedas & Egenhofer, 2008).

Inspired by Definition 1, we set up our similar spatial scene generation method $Sim^{(s)}$ to use addition, deletion and translation strategies to create similar spatial scene. Given a spatial scene $s_i = [c_i, (p_1, ..., p_n)]$ in a scale d, the method $Sim^{(s)}$ firstly add or delete several (e.g.,0-2) points with probability p_1 based on the number of points in the spatial scene and the scale d. Then $Sim^{(s)}$ randomly translates several(e.g.,0-2) points in a slightly small scale with probability p_2. $s'_i = [c_i, (p'_1, ..., p'_i, ..., p'_m)]$ is generated after these steps, where m is not necessarily equal to n according to Denifition 1.

However, the input with variable lengths still remains as a challenge for neural network mini-batch training. Inspired by Mai et al. (2022), we can pad the spatial scene s_i (same operation implemented to the similar one s'_i) to the same length l with the randomly sample points of polylines (e.g., road) or the edge of polygons (e.g., shopping mall), which can preserve the original information of the spatial scene. This is similar to the padding operation in CNN (Krizhevsky et al., 2017), so we call it spatial scene padding $Padding^{(s)}$. We can get $s_{i(pad)} = [c_i, (p_1, ..., p_i, ..., p_l)]$ and $s'_{i(pad)} = [c_i, (p'_1, ..., p'_i, ..., p'_l)]$ when using $Padding^{(s)}$, $s'_{i(pad)}$ is still similar to $s_{i(pad)}$. Note that with $Padding^{(s)}$, we can augment spatial scenes in different spatial scales while preserving its original features. Moreover, this process gives an idea of representation of polylines and polygons. Specifically, given a polyline or polygon with its basic information, our augmentation can enrich the number of points with the same feature to represent its shape, which contains its topological relationship. The left part of Figure 2a shows the workflow of our augmentation method. Initially, all spatial objects in spatial scenes are represented by a single point, the different colors of points indicate different features of these points. We first use $Sim^{(s)}$ to create a similar spatial scene, then we use $Padding^{(s)}$ to augment polylines or polygons data type with their geospatial shape.

Point Feature Encoder Each point $p_j = (x_j, v_j)$ in a spatial scene s_i has its point features v_i such as POI type name (e.g.,school). The point feature encoder $Enc^{(v)}()$ encodes such feature v_j into a feature embedding $\boldsymbol{e}_j[v_j] \in \mathbb{R}^{m^{(v)}}$. In this study, we consider POI types as point features. Let T is the total number of POI types. $Enc^{(v)}()$ is a embedding look-up function.

Point Location Encoder Inspired by Space2Vec (Mai et al., 2019), we use a relative point location encoder $Enc^{(x)}$ which uses sinusoid functions with different frequencies to encode relative locations in spatial scene s_i. Given x_j in the studied 2D space of $p_j = (x_j, v_j)$ in s_i, we first calculate the relative location $\boldsymbol{r}_j = x_j - c_j$, the location encoder $Enc^{(x)}(\boldsymbol{r}_j) = FFN(LE_j(\boldsymbol{r}_j))$

where $LE_j(\boldsymbol{r}_j) = [LE_{j,0}; ...; LE_{j,n}(\boldsymbol{r}_i); ...; LE_{j,N-1}(\boldsymbol{r}_j)]$ is a concatenation of multi-scale representations. Here N is the total number of spatial scene scales. $FFN()$ refers to a fully connected feed-forward network. In this study, $FFN()$ contains of three linear transformations with a ReLU activation and layer normalization (Ba et al., 2016) between each. Similar to Space2Vec, we use three unit vector $\boldsymbol{a}_k (k = 1, 2, 3)$ and the angle between any of them is $2\pi/3$ and $\lambda = \frac{scale_{max}}{scale_{min}}$ to represents the ratio of max scale and min scale of spatial scene. At each scale n, $LE_{j,n}(\boldsymbol{r}_j) = [LE_{j,n,1}(\boldsymbol{r}_i); , LE_{j,n,2}(\boldsymbol{r}_i); LE_{j,n,3}(\boldsymbol{r}_i)]$

$$LE_{j,n,k}(\boldsymbol{r}_i) = [\cos \frac{< \boldsymbol{r}_j, \boldsymbol{a}_k >}{scale_{min} \cdot \lambda^{n/N-1}}; \sin \frac{< \boldsymbol{r}_j, \boldsymbol{a}_k >}{scale_{min} \cdot \lambda^{n/N-1}}] \forall k = 1, 2, 3; \tag{1}$$

Here our point location encoder encodes the relative locations to the same representation dimension as point feature encoder, that is $m^{(r)} = m^{(v)}$.

Scene Encoder After point feature encoder and point location Encoder, we get feature embeddings $[\boldsymbol{e}_1[v_1], ..., \boldsymbol{e}_l[v_l]]$ and relative location embeddings $[\boldsymbol{e}_1[\boldsymbol{r}_1], ..., \boldsymbol{e}_l[\boldsymbol{r}_l]]$ of spatial objects in spatial scene s_i. We simply use $\boldsymbol{e}_j = \boldsymbol{e}_j[v_j] + \boldsymbol{e}_j[r_j] (\forall i = 1, ..., l)$ to integrate two kinds of embeddings inspired by the combination of positional encoding and input embedding in Transformer (Vaswani et al., 2017). Then we use $FFN()$ to model $[\boldsymbol{e}_1, ..., \boldsymbol{e}_l] \in \mathbb{R}^{l \times m}$ to $\boldsymbol{s} \in \mathbb{R}^d$. Figure 2b shows the architecture of our model.

4.2 Contrastive Learning

According to He et al. (2020), *"unsupervised learning trains encoders to perform dictionary look-up: an encoded "query" should be similar to its matching key and dissimilar to others. Learning is formulated as minimizing a contrastive loss (Hadsell et al., 2006)"*. Spatial scene similarity is also the basis of spatial prediction (Zhu et al., 2018). Therefore, we use contrastive learning method to train Scene2Vec. Specifically, we use the InfoNCE (Oord et al., 2018) as our contrastive loss to measure the similarity.

$$\mathcal{L}_q = -log \frac{\exp(q \cdot k_+ / \boldsymbol{\tau})}{\sum_{i=0}^{\mathcal{K}} \exp(q \cdot k_i / \boldsymbol{\tau})} \tag{2}$$

where $\boldsymbol{\tau}$ is a temperature hyper-parameter, we set $\boldsymbol{\tau}$ to 0.7 in this study. Correspondingly, we use a key encoder f_k with parameters θ_k and a query encoder f_q with parameters θ_q (both encoders use Scene2Vec) to get the key embedding k and query embedding q. Then we use momentum contrastive method (He et al., 2020) and inspired by Chen et al. (2021) that when the batch size is big enough (e.g.,4096), there is no need to build a big key embedding dictionary. Initially, we copy the parameters θ_k from θ_q and prohibit the gradient to update θ_k. We use a momentum coefficient (He et al., 2020) $m \in [0, 1)$ to update θ_k instead, m is a momentum hyper-parameter.

$$\theta_k \leftarrow m\theta_k + (1 - m)\theta_q \tag{3}$$

Moreover, we adopt symmetrized loss $\mathcal{L}_q(q_1, k_2) + \mathcal{L}_q(q_2, k_1)$ in Chen et al. (2021) to enhance the performance. We can get q_1, q_2, k_1, k_2 from the same spatial scene data s_i augmented by $Aug^{(s)}$ twice then encoded by f_q and f_k separately. The right part of Figure 2a shows the specific procedure of contrastive learning.

5 Experiment

We conduct experiments on real world data for the spatial scene similarity task 1) purely based on points and 2) based on multi-structure spatial objects to evaluate the performance of Scene2Vec compared with other widely used location encoding methods. For those baselines, we simply change the last layer of these encoders to get the same representation dimension as our model.

Baselines We compare Scene2Vec with three baselines including 1) *ffn* directly using feed-forward nets (Chu et al., 2019); 2) *rbf* Radial Basis Function (RBF) kernels (Baudat & Anouar, 2001); and 3) *Space2Vec* using multi-scale representation encoder (Mai et al., 2019).

Dataset We implement our experiments to evaluate the proposed model by using OpenStreetMap (OSM) (Haklay & Weber, 2008) data and large-scale commercial map data from Esri for the same area in Beijing, China. There are 58276 spatial objects with 22 different types in this dataset (Guo et al., 2022). We use the Pseudo-Mercator projection coordinate system to project the spatial objects geographic coordinates into projection coordinates. Initially, we separate them into 6480 spatial scenes with a scale of $400m \times 400m$ as the basis spatial scene data.

5.1 Spatial Scenes Augmentation

Based on the augmentation method in Section 4.1, we create similar spatial scenes and enrich our training dataset. Especially, this augment process is more likely to fulfill what has been occurred in our mental map in Section 1,which may be deviate or incomplete. Specifically, we use a set of operations (e.g. add, deletion, and spatial transformation) to randomly augment points data in spatial scenes. As for polylines and polygons, we generate points of polylines or polygons based on its spatial footprints and the given scale. We enlarge the training dataset by 20 times with the data augmentation techniques. The detailed parameters setting for data augmentation of different types of spatial objects are shown in Table 1.

Table 1: The parameter settings of spatial scene augmentation.

Geographic objects	Selection probability	Simulation probabilities and transformations		
		Drop	Data Type Aug	Transformation
Schools and education institutions	0.3	0.1	0.3	$0.3/\pm 100m$
Hotel accommodation	0.2	0.2	0.2	$0.2/\pm 100m$
Governmental agencies and institutes	0.2	0.2	0.1	$0.2/\pm 100m$
Roads	0.4	0.3	0.4	$0.2/\pm 50m$
stations	0.4	0.3	0	$0.2/\pm 50m$
Greenbelt and plants	0.3	0.3	0.4	$0.2/\pm 100m$
Restaurants	0.5	0.4	0.3	$0.2/\pm 100m$
Residential areas	0.3	0.4	0.4	$0.2/\pm 100m$
Rivers and lakes	0.3	0.2	0.2	$0.2/\pm 100m$
Shopping malls and markets	0.4	0.2	0.2	$0.3/\pm 100m$
Office buildings and commercial districts	0.4	0.4	0.3	$0.2/\pm 100m$
Hospitals and health care providers	0.2	0.2	0.2	$0.2/\pm 100m$
Life service business	0.2	0.3	0.4	$0.2/\pm 100m$
Scenic spots and resorts	0.2	0.2	0.1	$0.2/\pm 100m$

We further use cosine similarity to compute the similarity between paired spatial scenes we generated (see Definition 1). The results are shown in Table 2. The *Random augmentation* indicates generating random points based on the spatial objects features in the original spatial scene, which will cause great deviation in the relative geographical location while ours will not.

Table 2: Similarity evaluation of augmentation methods

Method	Similarity			
	Point	Polyline	Polygon	all
Random augmentation	0.323	0.237	0.102	0.211
Spatial scene augmentation	0.922	0.832	0.811	0.824

5.2 Spatial Scene Retrieval Tasks

As mentioned above, we evaluate *Scene2Vec* mainly based on spatial scene retrieval tasks of different data types of spatial objects 1) purely on points and 2) on all three data types – points, polyglines, and polygons. In these experiments, we first build a spatial scene embedding database based on the real-world Beijing spatial scenes using *Scene2Vec*, then we randomly select 350 simulated spatial scenes with various categories using our augmentation method together with 150 raw spatial scene

data for the comparison experiments to other baseline methods mentioned above. We using *K-Nearest Neighbor* (KNN) method to evaluate the test results. We evaluate all models using *Mean Reciprocal Rank* (MRR) and *HIT@k* (the chance of the true spatial scene being ranked to top k), where $k = [3, 5, 10]$.

5.2.1 SPATIAL SCENE RETRIEVAL TASK BASED ON POINTS

Since points are the most common spatial data type, we design the first experiment using a spatial scene dataset with purely points spatial objects. This means that in the spatial scene augmentation process, we abnegate the $Padding^{(s)}$ procedure and only use $Sim^{(s)}$ to obtain the similar spatial scenes with the same number of spatial objects. Table 3 shows the average metrics of different models with their best hyper-parameter setting on the contrastive learning. We can see that *knn* and *ffn* represent the relevance of the data itself to some extent, are less competitive. Single scale representation method *rbf* has a better performance. There is, however, the best results come from multi-scale representation models.

Table 3: The evaluation results of different models on the test dataset with point spatial objects

	Testing			
	MRR	HIT@3	HIT@5	HIT@10
knn	0.443	0.452	0.501	0.522
ffn	0.374	0.411	0.446	0.506
rbf(σ=500)	0.488	0.492	0.591	0.622
$Space2Vec_{grid}$(λ_{min}=100)	0.548	0.608	0.676	0.752
$Space2Vec_{theory}$(λ_{min}=100)	0.555	0.616	0.672	0.761
Scene2Vec($scale_{min}$=1k)	0.492	0.518	0.581	0.696
Scene2Vec($scale_{min}$=500)	0.563	0.616	0.684	0.768
Scene2Vec($scale_{min}$=100)	**0.598**	**0.652**	**0.734**	**0.812**

Compared with *rbf*(σ=500), both *Space2Vec* and *Scene2Vec* have a better performance because of their multi-scale structure. Specifically, we use two different types of *Space2Vec* and *Scene2Vec* in different minimum scales. We can see the result of *Scene2Vec* with 1k as it minimum scale is relatively poor, which indicates that the model will ignore some information of small scales. The improvements over $Space2Vec_{theory}$(λ_{min}=100) are -11.3%, +1.4%, +7.7% MRR for $scale_{min}$=1k, 500, 100 respectively.

5.2.2 SPATIAL SCENE RETRIEVAL TASK BASED ON THREE GEOSPATIAL DATA TYPES

In order to evaluate the capability of *Scene2Vec* of encoding polylines and polygons, we design this experiment to evaluate spatial scene retrieval using spatial objects with all data types (point, polyline and polygon). Here we first augment the polyline and polygon data types using *data type aug* operation instead, we use a series of points to represent a polyline or a polygon, which conforms to the geographic common representation. Table 4 shows the results of these models.

Table 4: The evaluation results of different models on the test dataset with all kinds of spatial objects

	Testing			
	MRR	HIT@3	HIT@5	HIT@10
knn	0.136	0.15	0.168	0.196
ffn	0.137	0.144	0.181	0.0.248
rbf(σ=500)	0.374	0.412	0.446	0.506
$Space2Vec_{grid}$(λ_{min}=100)	0.478	0.514	0.576	0.684
$Space2Vec_{theory}$(λ_{min}=100)	0.485	0.508	0.576	0.681
Scene2Vec($scale_{min}$=1k)	0.474	0.502	0.568	0.674
Scene2Vec($scale_{min}$=500)	0.547	0.591	0.656	0.718
Scene2Vec($scale_{min}$=100)	**0.564**	**0.612**	**0.686**	**0.773**

Because of the *data type aug* operation, there will be a considerable number of points with the same point feature. Therefore, *knn* and *ffn* have a terrible performance limited by their poor representational ability. Moreover, we can see that *Space2Vec* demonstrated its ability to encode polylines and polygons as the first step, and our model gets the best performance, which means our model is more robust compared with *Space2Vec*.

5.3 Spatial Analysis of Spatial Scene Encoding

To better illustrate the ability of our model to encode all these three data types of spatial objects. We generate a series of points which refers to the shape of a polyline or a polygon with its POI type as its label. Then we use the models mentioned above as the pre-trained model to implement a classification task which is similar to *point cloud classification* (Qi et al., 2017). We use Accuracy to evaluate these models.

Table 5: Comparing encoding performance based on the spatial data type classification

	Polylines	Polygons
ffn	0.322	0.331
rbf(σ=500)	0.482	0.476
$Space2Vec_{grid}$(λ_{min}=100)	0.633	0.652
$Space2Vec_{theory}$(λ_{min}=100)	0.663	0.687
Scene2Vec($scale_{min}$=100)	**0.676**	**0.712**
Main Types	Roads, Greenbelts	Residential areas Shopping malls and markets

Table 5 shows that *ffn* has poor ability to encode complex spatial data type and single-scale representation model *rbf* also performs terrible compared with multi-scale representation models. Moreover, *Scene2Vec* gets the best performance.

We further evaluate these methods by *land use clustering*. Specifically, we use clustering method to group the spatial scene data embeddings encoded by these models and calculate the accuracy of the clustering results. Unsurprisingly, Table 6 shows *Scene2Vec* has an average 3.5% promotion compared with $Space2Vec_{theory}$ in the same minimum scale 100.

Table 6: Comparing encoding performance based on the land use clustering

	Commercial districts	Residential areas	Public areas	Scenic spots
ffn	0.231	0.325	0.466	0.311
rbf(σ=500)	0.521	0.533	0.654	0.525
$Space2Vec_{grid}$(λ_{min}=100)	0.732	0.753	0.798	0.823
$Space2Vec_{theory}$(λ_{min}=100)	0.744	0.784	0.802	0.816
Scene2Vec($scale_{min}$=100)	**0.792**	**0.814**	**0.833**	**0.832**

6 Conclusion

In this work, we introduce an spatial scene representation model called *Scene2Vec* as a general-purpose representation model for spatial scenes that contain different spatial data types including points, polylines and polygons. The model can be trained in an unsupervised manner related to contrastive loss. We conduct two experiments on spatial scenes retrieval tasks with spatial objects based 1) purely on points and 2) on three spatial data types. Our analysis shows that *Scene2Vec* has the ability to encode various spatial data types and performs better than other baselines. In the future, we will integrate more spatial data types to build a more general representation model for spatial scenes and bring it to more downstream geospatial problem solving tasks.

REFERENCES

Shahnewaz Ali, Yaqub Jonmohamadi, Ross Crawford, Davide Fontanarosa, and Ajay K. Pandey. Surgery Scene Restoration for Robot Assisted Minimally Invasive Surgery. *arXiv e-prints*, 2021.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

C. Badue, R. Guidolini, R. V. Carneiro, P. Azevedo, V. B. Cardoso, A. Forechi, Frj Luan, R. F. Berriel, T. M. Paixo, and F. W. Mutz. Self-driving cars: A survey. *Pergamon*, 2021.

G. Baudat and F. Anouar. Kernel-based methods and function approximation. In *IJCNN'01. International Joint Conference on Neural Networks*, volume 2, pp. 1244–1249, 2001.

Tom Bruns and Max Egenhofer. Similarity of spatial scenes. In *Seventh international symposium on spatial data handling*, pp. 31–42. Delft, The Netherlands, 1996.

David Caduff and Max J Egenhofer. Geo-mobile queries: Sketch-based queries in mobile gis-environments. In *International Workshop on Web and Wireless Geographical Information Systems*, pp. 143–154. Springer, 2005.

Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9640–9649, 2021.

Grace Chu, Brian Potetz, Weijun Wang, Andrew Howard, Yang Song, Fernando Brucher, Thomas Leung, and Hartwig Adam. Geo-aware networks for fine-grained recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, Oct 2019.

Alessandro Crivellari and Euro Beinat. From motion activity to geo-embeddings: Generating and exploring vector representations of locations, traces and visitors through large-scale mobility data. *ISPRS International Journal of Geo-Information*, 8(3):134, 2019.

Suogui Dang, Yining Wu, Rui Yan, and Huajin Tang. Why grid cells function as a metric for space. *Neural Networks*, 142:128–137, 2021.

Max J Egenhofer. Query processing in spatial-query-by-sketch. *Journal of Visual Languages & Computing*, 8(4):403–424, 1997.

Andrew U Frank et al. *Spatial Information Theory: A Thoretical Basis for GIS. International Conference on Spatial Information Theory, COSIT'95*. Number 988. Springer Science & Business Media, 1995.

Ruiqi Gao, Jianwen Xie, Song-Chun Zhu, and Ying Nian Wu. Learning grid cells as vector representation of self-position coupled with matrix representation of self-motion. In *International Conference on Learning Representations*, 2018.

Elspeth Graham. What is a mental map? *Area*, pp. 259–262, 1976.

Danhuai Guo, Shiyin Ge, Shu Zhang, Song Gao, Ran Tao, and Yangang Wang. Deepssn: A deep convolutional neural network to assess spatial scene similarity. *Transactions in GIS*, 26(4):1914–1938, 2022.

Volker Haarslev and Michael Wessel. Querying gis with animated spatial sketches. In *Proceedings. 1997 IEEE Symposium on Visual Languages (Cat. No. 97TB100180)*, pp. 197–204. IEEE, 1997.

Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pp. 1735–1742, 2006.

Mordechai Haklay and Patrick Weber. Openstreetmap: User-generated street maps. *IEEE Pervasive computing*, 7(4):12–18, 2008.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9729–9738, 2020.

Krzysztof Janowicz, Song Gao, Grant McKenzie, Yingjie Hu, and Budhendra Bhaduri. Geoai: spatially explicit artificial intelligence techniques for geographic knowledge discovery and beyond. *International Journal of Geographical Information Science*, 34(4):625–636, 2020.

Neal Jean, Sherrie Wang, Anshul Samar, George Azzari, David Lobell, and Stefano Ermon. Tile2vec: Unsupervised representation learning for spatially distributed data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 3967–3974, 2019.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.

Shuang Liang, Zhengxing Sun, and Bin Li. Sketch retrieval based on spatial relations. In *International Conference on Computer Graphics, Imaging and Visualization (CGIV'05)*, pp. 24–29. IEEE, 2005.

Gengchen Mai, Krzysztof Janowicz, Bo Yan, Rui Zhu, Ling Cai, and Ni Lao. Multi-scale representation learning for spatial feature distributions using grid cells. In *International Conference on Learning Representations*, 2019.

Gengchen Mai, Krzysztof Janowicz, Ling Cai, Rui Zhu, Blake Regalia, Bo Yan, Meilin Shi, and Ni Lao. Se-kge: A location-aware knowledge graph embedding model for geographic question answering and spatial semantic lifting. *Transactions in GIS*, 24(3):623–655, 2020.

Gengchen Mai, Krzysztof Janowicz, Yingjie Hu, Song Gao, Bo Yan, Rui Zhu, Ling Cai, and Ni Lao. A Review of Location Encoding for GeoAI: Methods and Applications. *International Journal of Geographical Information Science*, 36:639–673, April 2022.

Clément Mallet, Frédéric Bretar, Michel Roux, Uwe Soergel, and Christian Heipke. Relevance assessment of full-waveform lidar data for urban area classification. *ISPRS journal of photogrammetry and remote sensing*, 66(6):S71–S84, 2011.

Konstantinos A Nedas and Max J Egenhofer. Spatial-scene similarity queries. *Transactions in GIS*, 12(6):661–681, 2008.

Vitor Oliveira and Paulo Pinho. Evaluation in urban planning: Advances and prospects. *Journal of Planning Literature*, 24(4):343–361, 2010.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Helen C Purchase, Eve Hoggan, and Carsten Görg. How important is the "mental map"?–an empirical investigation of a dynamic graph layout algorithm. In *International Symposium on Graph Drawing*, pp. 184–195. Springer, 2006.

Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 652–660, 2017.

Jinmeng Rao, Song Gao, Yuhao Kang, and Qunying Huang. Lstm-trajgan: A deep learning approach to trajectory privacy protection. *arXiv preprint arXiv:2006.10521*, 2020.

Waldo R Tobler. A computer movie simulating urban growth in the detroit region. *Economic geography*, 46(sup1):234–240, 1970.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

Stephan Winter. Location similarity of regions. *ISPRS Journal of Photogrammetry and Remote Sensing*, 55(3):189–200, 2000.

Yanyu Xu, Zhixin Piao, and Shenghua Gao. Encoding crowd interaction with deep neural network for pedestrian trajectory prediction. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5275–5284, 2018.

Bo Yan, Krzysztof Janowicz, Gengchen Mai, and Song Gao. From itdl to place2vec: Reasoning about place type similarity and relatedness by learning embeddings from augmented spatial contexts. In *Proceedings of the 25th ACM SIGSPATIAL*, pp. 1–10, 2017.

Yao Yao, Xia Li, Xiaoping Liu, Penghua Liu, Zhaotang Liang, Jinbao Zhang, and Ke Mai. POI2Vec. *International Journal of Geographical Information Science*, 31(4):825–848, 2017.

A-Xing Zhu, Guonian Lu, Jing Liu, Cheng-Zhi Qin, and Chenghu Zhou. Spatial prediction based on third law of geography. *Annals of GIS*, 24(4):225–240, 2018.

A APPENDIX

A.1 AUGMENTATION

To help understanding our specially designed spatial scene augmentation method, we first give the Algorithm 1 as the pseudo code of our method, which consists of two parts, similar spatial scene generation operation and padding operation. Note that in the *Padding* operation, the points we choosing in s_i and s'_i are partially the same to confirm their similarity. Moreover, We guarantee that after the padding operation both the number of points in s_i and s'_i are m. Here α_1 and α_2 are two hyperparameters we define beforehand as two probability thresholds. In practice, we use $\alpha_1 = 0.4$ and $\alpha_2 = 0.3$.

Algorithm 1: Workflow of spatial scene augmentation method.

Similar Spatial Scene Generation Operation*:*
Input: *Raw spatial scene data* s_i
Output: *Similar spatial scene data* s'_i

Initialize $\mathcal{P}_1 \sim U(0,1)$,$\mathcal{P}_2 \sim U(0,1)$ *randomly;*
if $\mathcal{P}_1 > \alpha_1$ **then**
  implement add or delete *operation:*
    n *refers to the number of points* p_i *in* s_i
    if $n < 4$ **then**
      adding 1-3 points in s_i
    else
      deleting 0-2 points in s_i *randomly*
if $\mathcal{P}_2 > \alpha_2$ **then**
  implement translation *operation:*
    randomly choose 1-3 points in s_i *to translate based on the* Transformation *column in Table 1*

Padding Operation*:*
Input: *Raw spatial scene data* s_i*, similar spatial scene data* s'_i *and padding number* m
Output: *spatial scene pair* $[s_{i(pad)}, s'_{i(pad)}]$ *with the same number of spatial objects*

n,k *refer to the number of points in* s_i, s'_i, *separately*
while $n < m$ **do**
  randomly choose a polyline/polygon point p_j *in* s_i
    get the geometry feature of p_j
    randomly add points based on the geometry feature
implement the same operation to s'_i

Moreover, we design a simple ablation experiment about our augmentation method. We compare the spatial scene retrieve results of *Scene2Vec* combined with *Similar* operation, *Padding* operation and both two operations. We can see that from Table 7 both operations can perform well individually, while the best results come from a combination of both. It is mainly because the combination of the two operations enhance the similarity and add some latent information.

Table 7: The ablation experiment results of augmentation method

	Testing			
	MRR	HIT@3	HIT@5	HIT@10
Scene2Vec$_{Similar}$	0.526	0.596	0.656	0.742
Scene2Vec$_{Padding}$	0.557	0.616	0.672	0.756
Scene2Vec	0.564	0.612	0.686	0.773

To evaluate the stability of *Padding* operation, we conduct the experiment of different padding points numbers in spatial scene retrieve task, the result in Table 8 shows that the evaluation indicators have slight change. Combined with other experiments in our article, we can see that our augmentation

method can generate reliable spatial scene data. The overall result shows a trend of first rising and then falling, we consider the main reason is as the number of points increases, the complexity of the entire data increases geometrically, which will affect the experimental results to some extent.

Table 8: The evaluation results of different padding points numbers in spatial scene retrieve task

	Testing			
	MRR	HIT@3	HIT@5	HIT@10
Scene2Vec($padding_{num}$=0)	0.526	0.596	0.656	0.742
Scene2Vec($padding_{num}$=2)	0.543	0.594	0.666	0.762
Scene2Vec($padding_{num}$=4)	0.561	0.622	0.688	0.774
Scene2Vec($padding_{num}$=6)	0.568	0.63	0.718	0.786
Scene2Vec($padding_{num}$=8)	0.565	0.622	0.696	0.778
Scene2Vec($padding_{num}$=10)	0.564	0.612	0.686	0.773
Scene2Vec($padding_{num}$=15)	0.552	0.606	0.674	0.752
Scene2Vec($padding_{num}$=20)	0.525	0.584	0.644	0.734
Scene2Vec($padding_{num}$=30)	0.481	0.508	0.58	0.682

A.2 Hyper-Parameter Selection

Augmentation The main hyper-parameters in augmentation are the number of points (spatial objects) in a spatial scene. We set the number $m = (6, 8, 10, 15, 20, 30)$, moreover, the result in Table 8 shows that all these parameters perform well in our retrieve experiment. After comprehensive consideration of geographical significance and time cost, we choose $m = 10$ as our benchmark.

Spatial Scene Modeling Grid search method is used for hyperparameter tuning based on their performance on the test datasets. The hyper-parameters of *Point Feature Encoder* and *Point Location Encoder* on gird search with $m^{(v)} = m^{(r)} = (32, 64, 128, 256, 512)$, and $scale_{min} = (5, 10, 50, 100, 300, 500)$ while $scale_{max} = 50k$ is decided based on the total size of our study area. As for *Scene Encoder*, the hyper-parameter are selected from $d = (64, 128, 256, 512, 1024)$. Combined accuracy and efficiency, we find out the best performance of different grid cell based models are obtained then $m^{(v)} = m^{(r)} = 64$, $scale_{min} = 100$ and $d = 128$. Moreover, we set the $batch_size = 4096$ based on the contrastive learning mechanisms. As for the *FFN()* in *Scene Encoder* and other models, we select the hyper-parameter from $f = (1, 2, 3)$ and $h = (2048, 4096, 8192)$. We find out $f = 2$ and $h = [8192, 4096]$ give the best performance.