MULTI-SCALE REPRESENTATION LEARNING FOR SPATIAL FEATURE DISTRIBUTIONS USING GRID CELLS

Anonymous authors
Paper under double-blind review

ABSTRACT

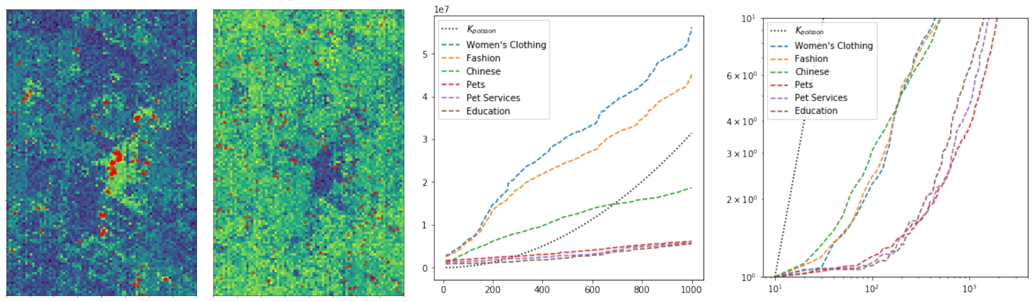
Unsupervised text encoding models have recently fueled substantial progress in Natural Language Processing (NLP). The key idea is to use neural networks to convert words in texts to vector space representations (embeddings) based on word positions in a sentence and their contexts. We see a strikingly similar situation in spatial analysis, which focuses on incorporating both absolute positions and spatial contexts of geographic objects such as Points of Interest (POIs) into models. A general space encoding method is valuable for a multitude of tasks such as POI search, land use classification, point-based spatial interpolation and location aware image classification. However, no such general model exists to date beyond simply applying discretizing or feed forward nets to coordinates, and little effort has been put into jointly modeling distributions with vastly different characteristics, which commonly emerges from GIS data. Meanwhile, Nobel Prize-winning Neuroscience research shows that grid cells in mammals provide a multi-scale periodic representation that functions as a metric for encoding space and are critical for recognizing places and for path-integration. Inspired by this research, we propose a representation learning model called Space2vec to encode the absolute positions and spatial relationships of places. We conduct experiments on real world geographic data and predict types of POIs at given positions based on their 1) locations and 2) nearby POIs. Results show that because of its multi-scale representations Space2vec outperforms well established ML approaches such as RBF kernels, multi-layer feed forward nets, and tile embedding approaches. [1]

1 INTRODUCTION

Unsupervised text encoding models such as Word2Vec (Mikolov et al., 2013), Glove (Pennington et al., 2014), ELMo (Peters et al., 2018), BERT (Devlin et al., 2018) have been effectively utilized in many Natural Language Processing (NLP) tasks. The idea is training deep models which encode words into vector space representations based on their positions in the text and their context. A similar situation can be encountered in the field of Geographic Information Science (GIS). For example, spatial interpolation aims at predicting an attribute value, e.g., elevation, of a given location based on values of nearby observations. Geographic information has become an important component to solve many tasks such as fine-grained image classification (Mac Aodha et al., 2019), point cloud classification and semantic segmentation (Qi et al., 2017), POI type similarity reasoning (Yan et al., 2017), land cover classification (Kussul et al., 2017), and geographic question answering (Mai et al., 2019). Developing general models for vector space representation of any point in space would pave the way to all these applications.

However, existing models often utilize *specific* methods to deal with geographic information and often disregards geographic coordinates. For example, Place2Vec (Yan et al., 2017) converts the coordinates of POIs into spatially collocated POI pairs within certain distance bins, and does not preserve information about distance and direction between POIs. Li et al. (2017) propose DCRNN for traffic forecasting in which the traffic sensor network is converted to a distance weighted graph which necessarily forfeits information about the spatial layout of sensors. There is, however, no general representation model beyond simply applying discretizing (Berg et al., 2014; Tang et al., 2015) or feed forward nets (Chu et al., 2019; Mac Aodha et al., 2019) to coordinates.

[1] link to repository redacted during double-blind review.



(a) Women's Cloth (b) Education (c) Ripley's K (d) Renormalized Ripley's K

Figure 1: The challenge of joint modeling distributions with very different characteristics. (a)(b) The POI locations (red dots) in Las Vegas and Space2vec predicted conditional likelihood of Women's Clothing (with a concentrated distribution) and Education (with an even distribution). The dark area in (b) indicates that the downtown area has more POIs of other types than Women's Clothing. (c) Ripley's K curves for POIs types for which Space2vec has the largest and smallest improvement over $wrap$ (Mac Aodha et al., 2019). Each curve represents the number of POIs of a certain type inside certain radios centered at every POI of that type; (d) Ripley's K curves renormalized by POI densities and shown in log-scale. To efficiently achieve multi-scale representation Space2vec concatenates the grid cell encoding of 64 scales (with wave lengths ranging from 50 meters to $40k$ meters) as the first layer of a deep model, and trains with POI data in an unsupervised fashion.

A key challenge in developing a general-purpose space representation model is how to deal with mixtures of distributions with very different characteristics (see an example in Figure 1), which often emerges from GIS data (McKenzie et al., 2015). For example, there are POI types with concentrated distributions such as women's clothing, while there are also POI types with even distributions such as education. These distributions co-exist in the same space, and yet we want a single space representation to accommodate all of them in a task such as location aware image classification (Mac Aodha et al., 2019). Ripley's K plot (Dixon, 2014) is a spatial analysis method used to describe how point patterns occur over a given area of interest. Figure 1c shows the K plot of several POI types in Las Vegas, and we can see that as the radius grows the numbers of POIs increase at different rates for different POI types. In order to see the relative change of density at different scales, we renormalize the curves by each POI's density and show in log scale in Figure 1d. We can clearly see two distinct POI type groups with different distribution patterns with concentrated and even distributions. If we want to model the distribution of these POIs by discretizing the study area into tiles, we have to use small grid sizes for women's clothing while using larger grid sizes for education because smaller grid sizes lead to over parameterization of the model and overfitting. In order to jointly describe these overlapping distributions with different distribution patterns, we need a space encoding method which utilizes *multi-scale representations*.

Nobel Prize winning Neuroscience research (Abbott & Callaway, 2014) shows that grid cells in mammals provide a multi-scale periodic representation that functions as a metric for encoding space, which are critical for integrating self-motion. Moreover, Blair et al. (2007) show that the multi-scale periodic representation of grid cells can be simulated by summing three cosine grating functions oriented 60° apart, which may be regarded as a simple Fourier model of the hexagonal lattice. This research inspired us to encode locations with multi-scale periodic representations. Our assumption is that decomposed geographic coordinates help machine learning models such as deep neural nets, and multi-scale representations help deal with the inefficiency of intrinsically single-scale methods such as RFB kernels or discretization (tile embeddings). To validate this idea, we propose an encoder-decoder framework to encode the distribution of point-features[2] in space and train such a model in an unsupervised manner. This idea of using sinusoid functions with different frequencies to encode positions in space is similar to position encoding proposed in the Transformer model (Vaswani et al., 2017). However, the position encoding model of Transformer deals with a discrete 1D space – the positions of words in a sentence – while our model works on higher dimensional continuous spaces such as the surface of earth. In summary, the contributions of this work are:

1. We propose an encoder-decoder framework called Space2vec for space encoding using sinusoid functions with different frequencies to model absolute positions and spatial contexts.

[2]In GIS and spatial analysis, 'features' are representations of real-world entities. A tree can, for instance, be modeled by a point-feature, while a street would be represented as a LineString feature.

We also propose a multi-head attention mechanism based on context points. To the best of our knowledge, this is the first attention model that explicitly considers the spatial relationships between the query point and context points.

2. We conduct two types of experiments on POI type prediction based on 1) the location of the query point (location modeling) and 2) its nearby POIs (spatial context modeling) with real world datasets. Results show that Space2vec outperforms commonly used encoding methods such as RBF kernels, multi-layer feed forward nets and tile embedding approaches. This is the first time grid cell-like representations achieve superior results in real-world GIS problems compared to well established representations in machine learning.
3. To understand the advantages of Space2vec we visualize the firing patterns of location models' encoding layer neurons and show how they model spatial structures at different scales by integrating multi-scale representations. Furthermore the firing patterns for the spatial context models' neurons give insight into how the grid cells capture the decreased distance effect with multi-scale representations.

2 Problem Formulation

Distributed representation of point-features in space can be formulated as follows. Given a set of points $\mathcal{P} = \{p_i\}$, i.e., Points of Interests (POIs), in L-D space ($L = 2, 3$) define a function $f_{\mathcal{P},\theta}(\mathbf{x}) : \mathbb{R}^L \rightarrow \mathbb{R}^d$ ($L < d$), which is parameterized by θ and maps any coordinate $\mathbf{x}$ in space to a vector representation of d dimension. Each point (e.g., a restaurant) $p_i = (\mathbf{x}_i, \mathbf{v}_i)$ is associated with a location $\mathbf{x}_i$ and attributes $\mathbf{v}_i$ (i.e., POI features such as type, name, capacity etc.). The function $f_{\mathcal{P},\theta}(\mathbf{x})$ encodes the probability distribution of point features over space and can give a representation of any point in the space. Attributes (e.g. place types such as *Museum*) and coordinate of point can be seen as analogies to words and word positions in commonly used word embedding models.

3 Related Work

There has been theoretical research on neural network based path integration/spatial localization models and their relationships with grid cells. Both Cueva & Wei (2018) and Banino et al. (2018) showed that grid-like spatial response patterns emerge in trained networks for navigation tasks which demonstrate that grid cells are critical for vector-based navigation. Moreover, Gao et al. (2019) propose a representational model for grid cells in navigation tasks which has good quality such as magnified local isometry. All these research is focusing on understanding the relationship between the grid-like spatial response patterns and navigation tasks from a theoretical perspective. In contrast, our goal focuses on utilizing these theoretical results on real world data in geoinformatics.

Radial Basis Function (RBF) kernel is a well established approach to generating learning friendly representation from points in space for machine learning algorithms such as SVM classification (Baudat & Anouar, 2001) and regression (Bierens, 1994). However, the representation is example based – i.e., the resultant model uses the positions of training examples as the centers of Gaussian kernel functions (Maz'ya & Schmidt, 1996). In comparison, the grid cell based space encoding relies on sine and cosine functions, and the resultant model does not need to store training examples.

Recently the computer vision community shows increasing interests in incorporating geographic information (e.g. coordinate encoding) into neural network architectures for multiple tasks such as image classification (Tang et al., 2015) and fine grained recognition (Berg et al., 2014; Chu et al., 2019; Mac Aodha et al., 2019). Both Berg et al. (2014) and Tang et al. (2015) proposed to discretize the study area into regular grids. To model the geographical prior distribution of the image categories, the grid id is used for GPS encoding instead of the raw coordinates. However, choosing the correct discretization is challenging (Openshaw, 1984; Fotheringham & Wong, 1991), and incorrect choices can significantly affect the final performance (Moat et al., 2018; Lechner et al., 2012). In addition, discretization does not scale well in terms of memory use. To overcome these difficulties, both Chu et al. (2019) and Mac Aodha et al. (2019) advocated the idea of inductive location encoders which directly encode coordinates into a location embedding. However, both of them directly feed the coordinates into a feed forward neural network (Chu et al., 2019) or residual blocks (Mac Aodha et al., 2019) without any feature decomposition strategy. Our experiments show that this direct encoding approach is insufficient to capture the spatial feature distribution and Space2vec significantly outperforms them by integrating spatial representations of different scales.

4 METHOD

We solve this unsupervised inductive learning problem with an encoder-decoder architecture:

1. Given a point $p_i = (\mathbf{x}_i, \mathbf{v}_i)$ a **point space encoder** $Enc^{(x)}()$ encodes location $\mathbf{x}_i$ into a space embedding $\mathbf{e}[\mathbf{x}_i] \in \mathbb{R}^{d^{(x)}}$ and a **point feature encoder** $Enc^{(v)}()$ encodes its feature into a feature embedding $\mathbf{e}[\mathbf{v}_i] \in \mathbb{R}^{d^{(v)}}$. $\mathbf{e} = [\mathbf{e}[\mathbf{x}_i]; \mathbf{e}[\mathbf{v}_i]] \in \mathbb{R}^d$ is the full representation of point $p_i \in \mathcal{P}$, where $d = d^{(x)} + d^{(v)}$. $[;]$ represents vector concatenation. While for a point $p_j \notin \mathcal{P}$ in the studied space, it only has space embedding $\mathbf{e}[\mathbf{x}_j]$ since its $\mathbf{v}_i$ is unknown.
2. We developed two types of decoders which can be used independently or jointly. A **location decoder** $Dec_s()$ reconstructs point feature embedding $\mathbf{e}[\mathbf{v}_i]$ given space embedding $\mathbf{e}[\mathbf{x}_i]$, and a **spatial context decoder** $Dec_c()$ reconstructs the feature embedding $\mathbf{e}[\mathbf{v}_i]$ of point p_i based on the space and feature embeddings $\{\mathbf{e}_{i1}, ..., \mathbf{e}_{ij}, ..., \mathbf{e}_{in}\}$ of nearest neighboring points $\{p_{i1}, ..., p_{ij}, ..., p_{in}\}$, where n is a hyper-parameter.

4.1 ENCODER

Point Feature Encoder Each point $p_i = (\mathbf{x}_i, \mathbf{v}_i)$ in a point set $\mathcal{P}$ is often associated with features such as the air pollution station data associate with some air quality measures, a set of POIs with POI types and check-ins, a set of points from survey and mapping with elevation values, a set of points from geological survey with mineral content measure, and so on. The point feature encoder $Enc^{(v)}()$ encodes such features $\mathbf{v}_i$ into a feature embedding $\mathbf{e}[\mathbf{v}_i] \in \mathbb{R}^{d^{(v)}}$. The implementation of $Enc^{(v)}()$ depends on the nature of these features. For example, if each point represents a POI with multiple POI types, the feature embedding $\mathbf{e}[\mathbf{v}_i]$ can simply be the mean of each POI type embedding $\mathbf{e}[\mathbf{v}_i] = \dfrac{1}{H}\sum_{h=1}^{H} \mathbf{t}_h^{(\gamma)}$, where $\mathbf{t}_h^{(\gamma)}$ indicates the hth POI type embedding of a POI p_i with H POI types. To help with training L_2 normalization can be applied to the POI type embedding matrix.

Point Space Encoder A part of the novelty of this paper is from the point space encoder $Enc^{(x)}()$. We first introduce Theorem 1 which provide an analytical solution $\phi(\mathbf{x})$ to encode any location $\mathbf{x} \in \mathbb{R}^2$ in 2D space to a distributed representation:

Theorem 1. *Let $\mathbf{\Psi}(\mathbf{x}) = (e^{i\langle \mathbf{a}_j, \mathbf{x}\rangle}, j = 1, 2, 3)^T \in \mathbb{C}^3$ where $e^{i\theta} = \cos\theta + i\sin\theta$ is the Euler notation of complex values; $\langle \mathbf{a}_j, \mathbf{x}\rangle$ is the inner product of $\mathbf{a}_j$ and $\mathbf{x}$. $\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3 \in \mathbb{R}^2$ are 2D vectors such that the angle between $\mathbf{a}_k$ and $\mathbf{a}_l$ is $2\pi/3$, $\forall j, \|\mathbf{a}_j\| = 2\sqrt{\alpha}$. Let $\mathbf{C} \in \mathbb{C}^{3\times 3}$ be a random complex matrix such as $\mathbf{C}^*\mathbf{C} = \mathbf{I}$. Then $\phi(\mathbf{x}) = \mathbf{C}\mathbf{\Psi}(\mathbf{x})$, $M(\Delta\mathbf{x}) = \mathbf{C}diag(\mathbf{\Psi}(\Delta\mathbf{x}))\mathbf{C}^*$ satisfies*

$$\phi(\mathbf{x} + \Delta\mathbf{x}) = M(\Delta\mathbf{x})\phi(\mathbf{x}) \tag{1}$$

and

$$\langle \phi(\mathbf{x} + \Delta\mathbf{x}), \phi(\mathbf{x})\rangle = d(1 - \alpha\|\Delta\mathbf{x}\|^2) \tag{2}$$

where $d = 3$ is the dimension of $\phi(\mathbf{x})$ and $\Delta\mathbf{x}$ is a small displacement from $\mathbf{x}$.

The proof of Theorem 1 can be seen in Gao et al. (2019). $\phi(\mathbf{x}) = \mathbf{C}\mathbf{\Psi}(\mathbf{x}) \in \mathbb{C}^3$ amounts to a 6-dimension real value vector and each dimension shows a *hexagon* firing pattern which models the grid cell behavior. Because of the periodicity of $sin()$ and $cos()$, $\phi(\mathbf{x})$ does not form a global codebook of 2D positions, i.e. there can be $\mathbf{x} \neq \mathbf{y}$, but $\phi(\mathbf{x}) = \phi(\mathbf{y})$. That means $\phi(\mathbf{x})$ does not encode $\mathbf{x}$ uniquely. Inspired by Theorem 1 and the multi-scale periodic representation of grid cells for coding space, we set up our point space encoder $\mathbf{e}[\mathbf{x}] = Enc^{(x)}_{theory}(\mathbf{x})$ to use sine and cosine functions of different frequencies to encode positions in space.

Given any point $\mathbf{x}$ in the studied 2D space, the space encoding $Enc^{(x)}_{theory}(\mathbf{x}) = \mathbf{NN}(PE^{(t)}(\mathbf{x}))$ where $PE^{(t)}(\mathbf{x}) = [PE^{(t)}_0(\mathbf{x}); ...; PE^{(t)}_s(\mathbf{x}); ...; PE^{(t)}_{S-1}(\mathbf{x})]$ is a concatenation of multi-scale representations of $d^{(x)} = 6S$ dimensions. Here S is the total number of grid scales and $s = 0, 1, 2, ..., S-1$. $\mathbf{NN}()$ represents fully connected ReLU layers. Let $\mathbf{a}_1 = [1, 0]^T, \mathbf{a}_2 = [-1/2, \sqrt{3}/2]^T, \mathbf{a}_3 = [-1/2, -\sqrt{3}/2]^T \in \mathbb{R}^2$ be three unit vectors and the angle between any of them is $2\pi/3$. $\lambda_{min}, \lambda_{max}$ are the minimum and maximum grid scale and $g = \frac{\lambda_{max}}{\lambda_{min}}$. At each scale

s, $PE_s^{(t)}(\mathbf{x}) = [PE_{s,1}^{(t)}(\mathbf{x}); PE_{s,2}^{(t)}(\mathbf{x}); PE_{s,3}^{(t)}(\mathbf{x})]$ is a concatenation of three components, where

$$PE_{s,j}^{(t)}(\mathbf{x}) = [\cos(\frac{\langle \mathbf{x}, \mathbf{a}_j \rangle}{\lambda_{min} \cdot g^{s/(S-1)}}); \sin(\frac{\langle \mathbf{x}, \mathbf{a}_j \rangle}{\lambda_{min} \cdot g^{s/(S-1)}})] \forall j = 1, 2, 3; \tag{3}$$

$\mathbf{NN}()$ and $PE^{(t)}(\mathbf{x})$ are analogies of $\mathbf{C}$ and $\mathbf{\Psi}(\mathbf{x})$ in Theorem 1.

Similarly we can define another space encoder $Enc_{grid}^{(x)}(\mathbf{x}) = \mathbf{NN}(PE^{(g)}(\mathbf{x}))$ inspired by the position encoding model of Transformer (Vaswani et al., 2017), where $PE^{(g)}(\mathbf{x}) = [PE_0^{(g)}(\mathbf{x}); ...; PE_s^{(g)}(\mathbf{x}); ...; PE_{S-1}^{(g)}(\mathbf{x})]$ is still a concatenation of its multi-scale representations, while $PE_s^{(g)}(\mathbf{x}) = [PE_{s,1}^{(g)}(\mathbf{x}); PE_{s,2}^{(g)}(\mathbf{x})]$ handles each component l of $\mathbf{x}$ separately:

$$PE_{s,l}^{(g)}(\mathbf{x}) = [\cos(\frac{\mathbf{x}^{[l]}}{\lambda_{min} \cdot g^{s/(S-1)}}); \sin(\frac{\mathbf{x}^{[l]}}{\lambda_{min} \cdot g^{s/(S-1)}})] \forall l = 1, 2 \tag{4}$$

4.2 DECODER

Two types of decoders are designed for two major types of GIS problems: location modeling and spatial context modeling (See Section 5.1).

Location Decoder $Dec_s()$ directly reconstructs point feature embedding $\mathbf{e}[\mathbf{v}_i]$ given its space embedding $\mathbf{e}[\mathbf{x}_i]$. We use one layer feed forward neural network $\mathbf{NN}_{\text{dec}}()$

$$\mathbf{e}[\mathbf{v}_i]' = Dec_s(\mathbf{v}_i|\mathbf{x}_i; \theta_{\text{dec}_s}) = \mathbf{NN}_{\text{dec}}(\mathbf{e}[\mathbf{x}_i]) \tag{5}$$

For training we use inner product to compare the reconstructed feature embedding $\mathbf{e}[\mathbf{v}_i]'$ against the real feature embeddings of $\mathbf{e}[\mathbf{v}_i]$ and other negative points (see detail in Sec 4.3).

Spatial Context Decoder $Dec_c()$ reconstructs the feature embedding $\mathbf{e}[\mathbf{v}_i]$ of the center point p_i based on the space and feature embeddings $\{\mathbf{e}_{i1}, ..., \mathbf{e}_{ij}, ..., \mathbf{e}_{in}\}$ of n nearby points $\{p_{i1}, ..., p_{ij}, ..., p_{in}\}$. Note that the feed-in order of context points should not affect the prediction results, which can be achieved by permutation invariant neural network architectures (Zaheer et al., 2017) like PointNet (Qi et al., 2017).

$$\mathbf{e}[\mathbf{v}_i]' = Dec_c(\mathbf{v}_i|\{\mathbf{e}_{i1}, ..., \mathbf{e}_{ij}, ..., \mathbf{e}_{in}\}; \theta_{\text{dec}_c}) = g(\frac{1}{K}\sum_{k=1}^{K}\sum_{j=1}^{n}\alpha_{ijk}\mathbf{e}[\mathbf{v}_{ij}]) \tag{6}$$

Here g is an activation function such as sigmoid. $\alpha_{ijk} = \frac{exp(\sigma_{ijk})}{\sum_{o=1}^{n} exp(\sigma_{iok})}$ is the attention of p_i with its jth neighbor through the kth attention head, and

$$\sigma_{ijk} = LeakyReLU(\mathbf{a}_k^T[\mathbf{e}[\mathbf{v}_i]_{init}; \mathbf{e}[\mathbf{v}_{ij}]; \mathbf{e}[\mathbf{x}_i - \mathbf{x}_{ij}]]) \tag{7}$$

where $\mathbf{a}_k \in \mathbb{R}^{2d^{(v)}+d^{(x)}}$ is the attention parameter in the kth attention head. The multi-head attention mechanism is inspired by Graph Attention Network (Veličković et al., 2018).

To represent the spatial relationship (distance and direction) between each context point $p_{ij} = (\mathbf{x}_{ij}, \mathbf{v}_{ij})$ and the center point $p_i = (\mathbf{x}_i, \mathbf{v}_i)$, we use the space encoder $Enc^{(x)}()$ to encode the displacement between them $\Delta\mathbf{x}_{ij} = \mathbf{x}_i - \mathbf{x}_{ij}$. Note that we are modeling the spatial interactions between the center point and n context points simultaneously.

In Eq. 7, $\mathbf{e}[\mathbf{v}_i]_{init}$ indicates the initial guess of the feature embedding $\mathbf{e}[\mathbf{v}_i]$ of point p_i which is computed by using another multi-head attention layer as Eq. 6 where the weight $\alpha'_{ijk} = \frac{exp(\sigma'_{ijk})}{\sum_{o=1}^{n} exp(\sigma'_{iok})}$. Here, σ'_{ijk} is computed as Eq. 8 where the query embedding $\mathbf{e}[\mathbf{v}_i]$ is excluded.

$$\sigma'_{ijk} = LeakyReLU(\mathbf{a}_k'^T[\mathbf{e}[\mathbf{v}_{ij}]; \mathbf{e}[\mathbf{x}_i - \mathbf{x}_{ij}]]) \tag{8}$$

4.3 TRAINING

The unsupervised learning task can simply be maximizing the log likelihood of observing the true point p_i at position $\mathbf{x}_i$ among all the points in $\mathcal{P}$

$$\mathcal{L}_{\mathcal{P}}(\theta) = -\sum_{p_i \in \mathcal{P}} \log P(p_i|p_{i1}, ..., p_{ij}, ..., p_{in}) = -\sum_{p_i \in \mathcal{P}} \log \frac{\exp(\mathbf{e}[\mathbf{v}_i]^T\mathbf{e}[\mathbf{v}_i]')}{\sum_{p_o \in \mathcal{P}} \exp(\mathbf{e}[\mathbf{v}_o]^T\mathbf{e}[\mathbf{v}_i]')} \tag{9}$$

Here only the feature embedding of p_i is used (without space embedding) to prevent revealing the identities of the point candidates, and $\theta = [\theta_{\text{enc}}; \theta_{\text{dec}}]$

Negative sampling by Mikolov et al. (2013) can be used to improve the efficiency of training

$$\mathcal{L}'_{\mathcal{P}}(\theta) = -\sum_{p_i \in \mathcal{P}} \Big(\log \sigma(\mathbf{e}[\mathbf{v}_i]^T \mathbf{e}[\mathbf{v}_i]') + \frac{1}{|\mathcal{N}_i|} \sum_{p_o \in \mathcal{N}_i} \log \sigma(-\mathbf{e}[\mathbf{v}_o]^T \mathbf{e}[\mathbf{v}_i]') \Big) \tag{10}$$

Here $\mathcal{N}_i \subseteq \mathcal{P}$ is a set of sampled negative points for p_i ($p_i \notin \mathcal{N}_i$) and $\sigma(x) = 1/(1+e^{-x})$.

5 Experiment

In this section we compare Space2vec with commonly used position encoding methods, and analyze them both quantitatively and qualitatively. We compare Space2vec to poplar coordinate encoding approaches including 1) $direct$ directly applying feed forward nets (Chu et al., 2019); 2) $tile$ discretization (Berg et al., 2014; Adams et al., 2015; Tang et al., 2015); 3) $wrap$ feed forward nets with coordinate wrapping (Mac Aodha et al., 2019); and 4) rbf Radial Basis Function (RBF) kernels (Baudat & Anouar, 2001; Bierens, 1994). See Appendix A.1 for details of the baselines.

5.1 Dataset and Tasks

To test the proposed model, we conduct experiments on geographic datasets with POI position and type information. We utilize the open-source dataset published by Yelp Data Challenge and select all POIs within the Las Vegas downtown area [3]. There are 21,830 POIs with 1,191 different POI types in this dataset. Note that each POI may be associated with one or more types, and we do not use any other meta-data such as business names, reviews for this study. We project geographic coordinates into projection coordinates using the NAD83/Conus Albers projection coordinate system[4]. The POIs are split into training, validation, and test dataset with ratios 80%:10%:10%. We create two tasks setups which represent different types of modeling need in Geographic Information Science:

- **Location Modeling** predicts the feature information associated with a POI based on its location $\mathbf{x}_i$ represented by the *location decoder* $Dec_s()$. This represents a large number of location prediction problems such as image fine grained recognition with geographic prior (Chu et al., 2019), and species potential distribution prediction (Zuo et al., 2008).
- **Spatial Context Modeling** predicts the feature information associated with a POI based on its context $\{\mathbf{e}_{i1}, ..., \mathbf{e}_{ij}, ..., \mathbf{e}_{in}\}$ represented by the *spatial context decoder* $Dec_c()$. This represents a collections of spatial context prediction problem such as spatial context based facade image classification (Yan et al., 2018), and all spatial interpolation problems.

We use POI prediction metrics to evaluate these models. Given the real point feature embedding $\mathbf{e}[\mathbf{v}_i]$ and N negative feature embeddings $\mathcal{N}_i = \{\mathbf{e}[\mathbf{v}_i]^-\}$, we compare the predicted $\mathbf{e}[\mathbf{v}_i]'$ with them by cosine distance. The cosine scores are used to rank $\mathbf{e}[\mathbf{v}_i]$ and N negative samples. The negative feature embeddings are the feature embeddings of points p_j randomly sampled from $\mathcal{P}$ and $p_i \neq p_j$. We evaluate each model using Negative Log-Likelihood (NLL), Mean Reciprocal Rank (MRR) and HIT@5 (the chance of the true POI being ranked to top 5. We train and test each model 10 times to estimate standard deviations. See Appendix A.2 for hyper-parameter selection details.

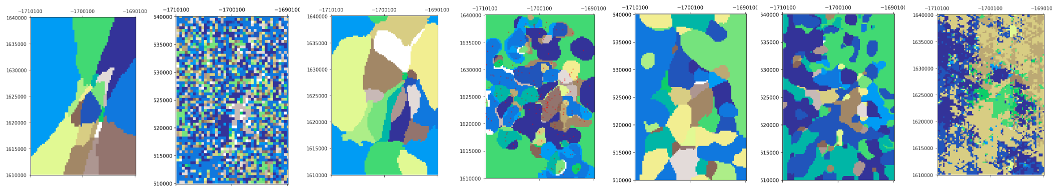
5.2 Location Modeling Evaluation

We first study location modeling with the **location decoder** $Dec_s()$ in Section 4.2. We use a negative sample size of $N = 100$. Table 1 shows the average metrics of different models with their best hyper-parameter setting on the validation set. We can see that $direct$ and $theorydiag$ are less competitive, only beating the $random$ selection baseline. Other methods with single scale representations – including $tile$, $wrap$, and rbf – perform better. The best results come from various version of the grid cell models, which are capable of dealing with multi-scale representations.

In order to understand the reason for the superiority of grid cell models we provide qualitative analysis of their representations. We apply hierarchical clustering to the space embeddings produced by studied models using cosine distance as the distance metric (See Fig. 2). we can see that when

[3]The geographic range is (35.989438, 36.270897) for latitude and (-115.047977, -115.3290609) for longitude.

[4]`https://epsg.io/5070-1252`



(a) $direct$ (b) $tile$ (c) $wrap$ (d) rbf (σ=1k) (e) λ_{min}=1k (f) λ_{min}=500 (g) λ_{min}=50

Figure 2: Embedding clustering of (a) $direct$; (b) $tile$ with the best cell size $c = 500$; (c) $wrap$ ($h = 3, o = 512$); (d) rbf with the best σ (1k) and 200 anchor points (red) and (e)(f)(h) $theory$ models with different λ_{min}, but fixed $\lambda_{max} = 40k$ and $S = 64$. All models use 1 hidden ReLU layers of 512 neurons except $wrap$.

Table 1: The evaluation results of different location models on the validation and test dataset.

	Train	Validation			Testing	
	NLL	NLL	MRR	HIT@5	MRR	HIT@5
$random$		-	0.052 (0.002)	4.8 (0.5)	0.051 (0.002)	5.0 (0.5)
$direct$	1.285	1.332	0.089 (0.001)	10.6 (0.2)	0.090 (0.001)	11.3 (0.2)
$tile$ (c=500)	1.118	1.261	0.123 (0.001)	16.8 (0.2)	0.120 (0.001)	17.1 (0.3)
$wrap$(h=3,o=512)	1.222	1.288	0.112 (0.001)	14.6 (0.1)	0.119 (0.001)	15.8 (0.2)
rbf (σ=1k)	1.209	1.279	0.115 (0.001)	15.2 (0.2)	0.123 (0.001)	16.8 (0.3)
$grid$ (λ_{min}=50)	1.156	1.258	0.128 (0.001)	18.1 (0.3)	0.139 (0.001)	**20.0** (0.2)
$hexa$ (λ_{min}=50)	1.230	1.297	0.107 (0.001)	14.0 (0.2)	0.105 (0.001)	14.5 (0.2)
$theorydiag$ (λ_{min}=50)	1.277	1.324	0.094 (0.001)	12.3 (0.3)	0.094 (0.002)	11.2 (0.3)
$theory$ (λ_{min}=1k)	1.207	1.281	0.123 (0.002)	16.3 (0.5)	0.121 (0.001)	16.2 (0.1)
$theory$ (λ_{min}=500)	1.188	1.269	0.132 (0.001)	17.6 (0.3)	0.129 (0.001)	17.7 (0.2)
$theory$ (λ_{min}=50)	1.098	1.249	**0.137** (0.002)	**19.4** (0.1)	**0.144** (0.001)	**20.0** (0.2)

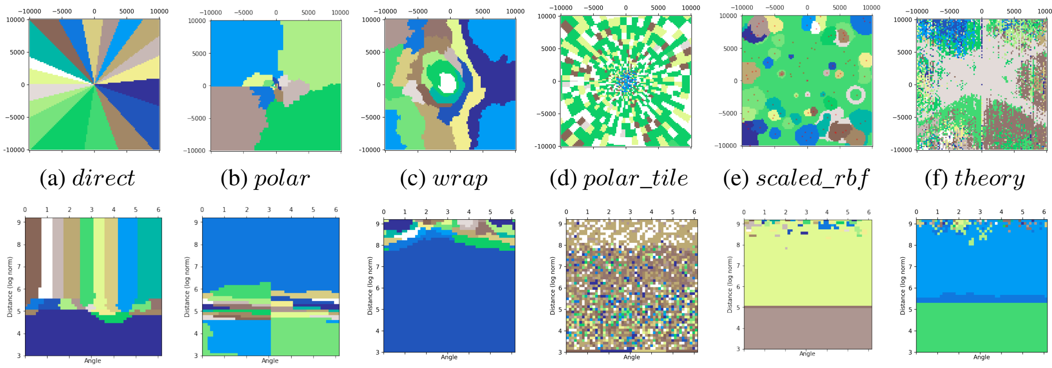
restricted to large grid sizes ($\lambda_{min} = 1k$), $theory$ has similar representation (Fig. 2d, 2e, and Fig. 4d, 4e) and performance compared to rbf ($\sigma = 1k$). However it is able to significantly outperform rbf ($\sigma = 1k$) (and $tile$ and $wrap$) when small grid sizes ($\lambda_{min} = 500, 50$) are available. The relative improvements over rbf ($\sigma = 1k$) are -0.2%, +0.6%, +2.1% MRR for λ_{min}=1k, 500, 50 respectively.

We compare the performance of two $theory$ models with $\lambda_{min} = 50$ and 500 on different types of POIs. We found that the biggest improvements come from POI types with concentrated distributions such as women's clothing, restaurants, nightclubs, bars while there is no or negative improvements on POI types with even distributions (e.g., education, hospital, schools, pet and financial services). This result is well aligned with the previous result about different spatial distribution patterns of different POI types (McKenzie et al., 2015). Smaller grid cells can discriminate POI types which tend to cluster together. In contrast, when rbf uses a smaller σ (and more anchors) or $tile$ uses smaller tile sizes their number of parameters grows quickly, thus leading to overfitting.

5.3 Spatial Context Modeling Evaluation

Next, we evaluate the **spatial context decoder** $Dec_c()$ in Sec. 4.2. We use the same evaluation set up as location modeling. The context points are obtained by querying the n-th nearest points using PostGIS ($n = 10$). As for validation and test datasets, we make sure the center points are all unknown during the training phase. Table 2 shows the evaluation results of different models for spatial context modeling. The baseline approaches ($direct$, $tile$, $wrap$, rbf) generally perform poorly in context modeling. We designed specialized version of these approaches ($polar$, $polar\_tile$, $scaled\_rbf$) with polar coordinates, which lead to significantly improvements. Note that these are models proposed by us specialized for context modeling and therefore are less general than the grid cell approaches. Nevertheless the grid cell approaches are able to perform better than the specialized approaches.

Figure 6 shows the space embedding clustering results in both Cartesian and polar coordinate systems. We can see that $direct$ (Fig. 3a, 3g) only captures the distance information when the context POI is very close ($log(\| \Delta\mathbf{x}_{ij} \| + 1) \leqslant 5$) while in the farther spatial context it purely models the direction information. $polar$ (Fig. 3b, 3h) has the similar behaviors but captures the distance information in a more fine-grained manner. $wrap$ (Fig. 3c, 3i) mainly focuses on differentiating relative positions in



(a) *direct* (b) *polar* (c) *wrap* (d) *polar_tile* (e) *scaled_rbf* (f) *theory*

(g) *direct* (h) *polar* (i) *wrap* (j) *polar_tile* (k) *scaled_rbf* (l) *theory*

Figure 3: Embedding clustering in the original space of (a) $direct$; (b) $polar$; (c) $wrap$, h=2,o=512; (d) $polar\_tile$, F = 64, (e) $scaled\_rbf$, σ = 40, β=0.1; and (f) $theory$, $\lambda_{min} = 10$, $\lambda_{max} = 10k$, $S = 64$. (g)(h)(i)(j)(k)(l) are the clustering results of the same models in the polar-distance space using $\log(\| \Delta \mathbf{x}_{ij} \| + 1)$. All models use 1 hidden ReLU (except $wrap$) layers of 512 neurons. Most models except $wrap$ can capture a shift when distance is around $e^5 - 1 \approx 150$ meters.

Table 2: The evaluation results of different spatial context models on the validation and test dataset. All encoders contains a 1 hidden layer FFN. All grid cell encoders set λ_{min}=10, λ_{max}=10k.

	Train	Validation			Testing	
$Space2vec$	NLL	NLL	MRR	HIT@5	MRR	HIT@5
$none$	1.163	1.297	0.159 (0.002)	22.4 (0.5)	0.167 (0.006)	23.4 (0.7)
$direct$	1.151	1.282	0.170 (0.002)	24.6 (0.4)	0.175 (0.003)	24.7 (0.5)
$polar$	1.157	1.283	0.176 (0.004)	25.4 (0.4)	0.178 (0.006)	24.9 (0.1)
$tile$ $(c = 50)$	1.163	1.298	0.173 (0.004)	24.0 (0.6)	0.173 (0.001)	23.4 (0.1)
$polar\_tile(F = 64)$	1.161	1.282	0.173 (0.003)	25.0 (0.1)	0.177 (0.001)	24.5 (0.3)
$wrap$ (h=2,o=512)	1.167	1.291	0.159 (0.001)	23.0 (0.1)	0.170 (0.001)	23.9 (0.2)
rbf $(\sigma = 50)$	1.160	1.281	**0.179** (0.002)	25.2 (0.6)	0.172 (0.001)	25.0 (0.1)
$scaled\_rbf$ (σ=40,β=0.1)	1.150	1.272	0.177 (0.002)	**25.7** (0.1)	0.181 (0.001)	25.3 (0.1)
$grid$(λ_{min}=10)	1.172	1.285	0.178 (0.004)	24.9 (0.5)	0.181 (0.001)	25.1 (0.3)
$hexa$ (λ_{min}=10)	1.156	1.289	0.173 (0.002)	24.0 (0.2)	0.183 (0.002)	25.3 (0.2)
$theorydiag$ $(\lambda_{min} = 10)$	1.156	1.287	0.168 (0.001)	24.1 (0.4)	0.174 (0.005)	24.9 (0.1)
$theory$(λ_{min}=200)	1.168	1.295	0.159 (0.001)	23.1 (0.2)	0.170 (0.001)	23.2 (0.2)
$theory$(λ_{min}=50)	1.157	1.275	0.171 (0.001)	24.2 (0.3)	0.173 (0.001)	24.8 (0.4)
$theory$(λ_{min}=10)	1.158	1.280	0.177 (0.003)	25.2 (0.3)	**0.185** (0.002)	**25.7** (0.3)

farther spatial context $cont$ which might explain its lower performance[5]. $polar\_tile$ (Fig. 3d) mostly responds to distance information. Interestingly, $scaled\_rbf$ and $theory$ have similar representations in the polar coordinate system (Fig. 3k, 3l) and similar performance (Table 2). While $scaled\_rbf$ captures the gradually decreased distance effect with a scaled kernel size which becomes larger in farther distance, $theory$ achieves this by integrating representations of different scales.

6 Conclusion

We introduced an encoder-decoder framework as a general-purpose space representation model inspired by biological grid cells' multi-scale periodic representations. The model is an inductive learning model and can be trained in an unsupervised manner. We conduct two experiments on POI type prediction based on 1) POI locations and 2) nearby POIs. The evaluation results demonstrate the effectiveness of our model. Our analysis reveals that it is the ability to integrate representations of different scales that makes the grid cell models outperform other baselines on these two tasks. In the future, we hope to incorporate the presented framework to more complex GIS tasks such as social network analysis, and sea surface temperature prediction.

[5]Note that $wrap$ is original proposed by Mac Aodha et al. (2019) for location modelling, not spatial context modelling. This results indicates $wrap$ is not good at this task.

REFERENCES

Alison Abbott and Ewen Callaway. Nobel prize for decoding brain's sense of place. *Nature News*, 514(7521):153, 2014.

Benjamin Adams, Grant McKenzie, and Mark Gahegan. Frankenplace: interactive thematic mapping for ad hoc exploratory search. In *Proceedings of the 24th international conference on world wide web*, pp. 12–22. International World Wide Web Conferences Steering Committee, 2015.

Andrea Banino, Caswell Barry, Benigno Uria, Charles Blundell, Timothy Lillicrap, Piotr Mirowski, Alexander Pritzel, Martin J Chadwick, Thomas Degris, Joseph Modayil, et al. Vector-based navigation using grid-like representations in artificial agents. *Nature*, 557(7705):429, 2018.

G Baudat and F Anouar. Kernel-based methods and function approximation. volume 2, pp. 1244 – 1249 vol.2, 02 2001. ISBN 0-7803-7044-9. doi: 10.1109/IJCNN.2001.939539.

Thomas Berg, Jiongxin Liu, Seung Woo Lee, Michelle L Alexander, David W Jacobs, and Peter N Belhumeur. Birdsnap: Large-scale fine-grained visual categorization of birds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2011–2018, 2014.

Herman J. Bierens. The nadaraya–watson kernel regression function estimator. *Topics in Advanced Econometrics*, 16:212–247, 1994.

Hugh T Blair, Adam C Welday, and Kechen Zhang. Scale-invariant memory representations emerge from moire interference between grid fields that produce theta oscillations: a computational model. *Journal of Neuroscience*, 27(12):3211–3229, 2007.

Grace Chu, Brian Potetz, Weijun Wang, Andrew Howard, Yang Song, Fernando Brucher, Thomas Leung, and Hartwig Adam. Geo-aware networks for fine grained recognition. *arXiv preprint arXiv:1906.01737*, 2019.

Christopher J Cueva and Xue-Xin Wei. Emergence of grid-like representations by training recurrent neural networks to perform spatial localization. *arXiv preprint arXiv:1803.07770*, 2018.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Philip M Dixon. R ipley's k function. *Wiley StatsRef: Statistics Reference Online*, 2014.

A Stewart Fotheringham and David WS Wong. The modifiable areal unit problem in multivariate statistical analysis. *Environment and planning A*, 23(7):1025–1044, 1991.

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 *Proceedings of ICLR 2019*, 2019.

Nataliia Kussul, Mykola Lavreniuk, Sergii Skakun, and Andrii Shelestov. Deep learning classification of land cover and crop types using remote sensing data. *IEEE Geoscience and Remote Sensing Letters*, 14(5):778–782, 2017.

Alex M Lechner, William T Langford, Simon D Jones, Sarah A Bekessy, and Ascelin Gordon. Investigating species–environment relationships at multiple scales: Differentiating between intrinsic scale and the modifiable areal unit problem. *Ecological Complexity*, 11:91–102, 2012.

Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. *arXiv preprint arXiv:1707.01926*, 2017.

Oisin Mac Aodha, Elijah Cole, and Pietro Perona. Presence-only geographical priors for fine-grained image classification. *arXiv preprint arXiv:1906.05272*, 2019.

Gengchen Mai, Bo Yan, Krzysztof Janowicz, and Rui Zhu. Relaxing unanswerable geographic questions using a spatially explicit knowledge graph embedding model. In *AGILE: The 22nd Annual International Conference on Geographic Information Science*, pp. 21–39. Springer, 2019.

V Maz'ya and G Schmidt. On approximate approximations using gaussian kernels. *IMA Journal of Numerical Analysis*, 16:13–29, 01 1996.

Grant McKenzie, Krzysztof Janowicz, Song Gao, Jiue-An Yang, and Yingjie Hu. Poi pulse: A multi-granular, semantic signature–based information observatory for the interactive visualization of big geosocial data. *Cartographica: The International Journal for Geographic Information and Geovisualization*, 50(2):71–85, 2015.

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

Justin Moat, Steven P Bachman, Richard Field, and Doreen S Boyd. Refining area of occupancy to address the modifiable areal unit problem in ecology and conservation. *Conservation biology*, 32 (6):1278–1289, 2018.

Stan Openshaw. The modifiable areal unit problem. *Concepts and techniques in modern geography*, 1984.

Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543, 2014.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*, 2018.

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.

Kevin Tang, Manohar Paluri, Li Fei-Fei, Rob Fergus, and Lubomir Bourdev. Improving image classification with location context. In *Proceedings of the IEEE international conference on computer vision*, pp. 1008–1016, 2015.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pp. 5998–6008, 2017.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. In *ICLR 2018*, 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 International Conference on Advances in Geographic Information Systems*, pp. 35. ACM, 2017.

Bo Yan, Krzysztof Janowicz, Gengchen Mai, and Rui Zhu. xnet+ sc: Classifying places based on images by incorporating spatial contexts. In *10th International Conference on Geographic Information Science (GIScience 2018)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018.

Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Ruslan R Salakhutdinov, and Alexander J Smola. Deep sets. In *Advances in neural information processing systems*, pp. 3391–3401, 2017.

Wenyun Zuo, Ni Lao, Yuying Geng, and Keping Ma. Geosvm: an efficient and effective tool to predict species' potential distributions. *Journal of Plant Ecology*, 1(2):143–145, 2008.

A APPENDIX

A.1 BASELINES

To help understand the mechanism of distributed space representation we compare multiple ways of encoding spatial information. Different models use different point space encoder $Enc^{(x)}()$ to encode either location $\mathbf{x}_i$ (for location modeling loc) or the displacement between the center point and one context point $\Delta\mathbf{x}_{ij} = \mathbf{x}_i - \mathbf{x}_{ij}$ (for spatial context modeling $cont$)[6].

- $random$ shuffles the order of the correct POI and N negative samples randomly as the predicted ranking. This shows the lower bound of each metrics.
- $direct$ directly encode location $\mathbf{x}_i$ (or $\Delta\mathbf{x}_{ij}$ for $cont$) into a space embedding $\mathbf{e}[\mathbf{x}_i]$ (or $\mathbf{e}[\Delta\mathbf{x}_{ij}]$) using a feed forward neural networks (FFNs)[7], denoted as $Enc^{(x)}_{direct}(\mathbf{x})$ without decomposing coordinates into a multi-scale periodic representation. This is essentially the GPS encoding method used by Chu et al. (2019). Note that Chu et al. (2019) is not open sourced and we end up implementing the model architecture ourselves.
- $tile$ divides the study area A_{loc} (for loc) or the range of spatial context defined by λ_{max}, A_{cont}, (for $cont$) into grids with equal grid sizes c. Each grid has an embedding to be used as the space encoding for every location $\mathbf{x}_i$ or displacement $\Delta\mathbf{x}_{ij}$ fall into this grid. This is a common practice by many previous work when dealing with coordinate data (Berg et al., 2014; Adams et al., 2015; Tang et al., 2015).
- $wrap$ is a location encoder model recently introduced by Mac Aodha et al. (2019). It first normalizes $\mathbf{x}$ (or $\Delta\mathbf{x}$) into the range $[-1, 1]$ and uses a coordinate wrap mechanism $[\sin(\pi\mathbf{x}^{[l]}); \cos(\pi\mathbf{x}^{[l]})]$ to convert each dimension of $\mathbf{x}$ into 2 numbers. This is then passed through an initial fully connected layer, followed by a series of h residual blocks, each consisting of two fully connected layers (o hidden neurons) with a dropout layer in between. We adopt the official code of Mac Aodha et al. (2019)[8] for this implementation.
- rbf randomly samples M points from the training dataset as RBF anchor points $\{\mathbf{x}^{anchor}_m, m = 1...M\}$ (or samples M $\Delta\mathbf{x}^{anchor}_m$ from A_{cont} for $cont$) [9], and use gaussian kernels $\exp\big(-\dfrac{\| \mathbf{x}_i - \mathbf{x}^{anchor}_m \|^2}{2\sigma^2}\big)$ (or $\exp\big(-\dfrac{\| \Delta\mathbf{x}_{ij} - \Delta\mathbf{x}^{anchor}_m \|^2}{2\sigma^2}\big)$ for $cont$) on each anchor points, where σ is the kernel size. Each point p_i has a M-dimension RBF feature vector which is fed into a FNN to obtain the spatial embedding. This is a strong baseline for representing floating number features in machine learning models.
- $grid$ as described in Section 4.1 inspired by the position encoding in Transformer (Vaswani et al., 2017).
- $hexa$ Same as $grid$ but use $sin(\theta)$, $sin(\theta + 2\pi/3)$, and $sin(\theta + 4\pi/3)$ in $PE^{(g)}_{s,l}(\mathbf{x})$.
- $theory$ as described in Section 4.1, uses the theoretical models (Gao et al., 2019) as the first layer of $Enc^{(x)}_{theory}(\mathbf{x})$ or $Enc^{(x)}_{theory}(\Delta\mathbf{x}_{ij})$.
- $theorydiag$ further constrains $\mathbf{NN}()$ as a block diagonal matrix, with each scale as a block.

We also have the following baselines which are specific to the spatial context modeling task.

- $none$ the decoder $Dec_c()$ does not consider the spatial relationship between the center point and context points but only the co-locate patterns such as Place2Vec (Yan et al., 2017). That means we drop the $\mathbf{e}[\Delta\mathbf{x}_{ij}]$ from the attention mechanism in Equ. 7 and 8.
- $polar$ first converts the displacement $\Delta\mathbf{x}_{ij}$ into polar coordinates (r, θ) centered at the center point where $r = log(\| \Delta\mathbf{x}_{ij} \| + 1)$. Then it uses $[r, \theta]$ as the input for a FFN to obtain the spatial relationship embedding in Equ. 7. We find out that it has a significant performance improvement over the variation with $r = \| \Delta\mathbf{x}_{ij} \|$.

[6]We will use meter as the unit of $\lambda_{min}, \lambda_{max}, \sigma, c$.
[7]we first normalizes $\mathbf{x}$ (or $\Delta\mathbf{x}$) into the range $[-1, 1]$
[8]`http://www.vision.caltech.edu/~macaodha/projects/geopriors/`
[9]these anchor points are fixed in both loc and $cont$.

- $polar\_tile$ is a modified version of $tile$ but the grids are extracted from polar coordinates (r, θ) centered at the center point where $r = log(\parallel \Delta\mathbf{x}_{ij} \parallel +1)$. Instead of using grid size c, we use the number of grids along θ (or r) axis, F, as the only hyperparameter. Similarly, We find that $r = log(\parallel \Delta\mathbf{x}_{ij} \parallel +1)$ outperform $r = \parallel \Delta\mathbf{x}_{ij} \parallel$ significantly.
- $scaled\_rbf$ is a modified version of rbf for $cont$ whose kernel size is proportional to the distance between the current anchor point and the origin, $\parallel \Delta\mathbf{x}_m^{anchor} \parallel$. That is $\exp\big(-\frac{\parallel \Delta\mathbf{x}_{ij} - \Delta\mathbf{x}_m^{anchor} \parallel^2}{2\sigma_{scaled}^2}\big)$. Here $\sigma_{scaled} = \sigma + \beta \parallel \Delta\mathbf{x}_m^{anchor} \parallel$ where σ is the basic kernel size and β is kernel rescale factor, a constant. We developed this mechanism to help RFB to deal with relations at different scale, and we observe that it produces significantly better result than vanilla RBFs.

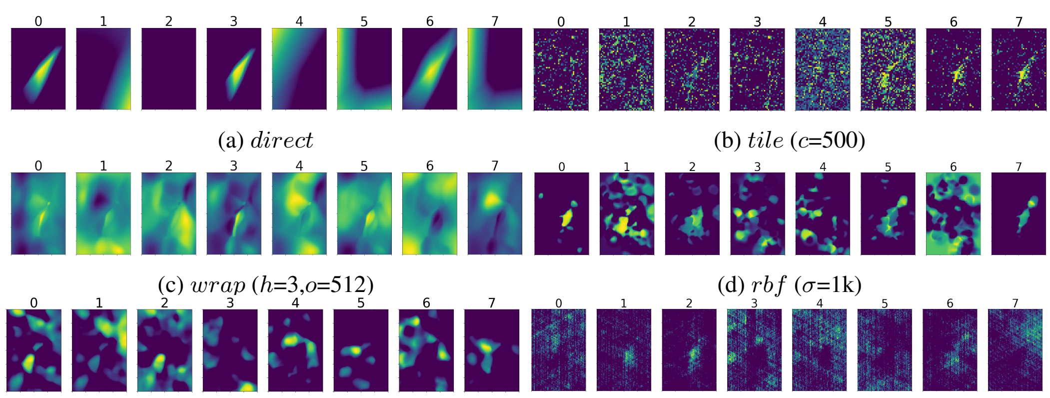
A.2 Hyper-Parameter Selection

We perform grid search for all methods based on their performance on the validation sets.

Location Modeling The hyper-parameters of $theory$ models are based on grid search with $d^{(v)} = (32, 64, 128, 256)$, $d^{(x)} = (32, 64, 128, 256)$, $S = (4, 8, 16, 32, 64, 128)$, and $\lambda_{min} = (1, 5, 10, 50, 100, 200, 500, 1k)$ while $\lambda_{max} = 40k$ is decided based on the total size of the study area. We find out the best performances of different grid cell based models are obtained when $d^{(v)} = 64$, $d^{(x)} = 64$, $S = 64$, and $\lambda_{min} = 50$. In terms of $tile$, the hyper-parameters are selected from $c = (10, 50, 100, 200, 500, 1000)$ while $c = 500$ gives us the best performance.As for rbf, we do grid search on the hyper-parameters: $M = (10, 50, 100, 200, 400, 800)$ and $\sigma = (10^2, 10^3, 10^4, 10^5, 10^6, 10^7)$. The best performance of rbf is obtain when $M = 200$ and $\sigma = 10^3$. As for $wrap$, grid search is performed on: $h = (1, 2, 3, 4)$ and $o = (64, 128, 256, 512)$ while $h = 3$ and $o = 512$ gives us the best result. All models use FFNs in their $Enc^{(x)}()$ except $wrap$. The number of layers f and the number of hidden state neurons u of the FFN are selected from $f = (1, 2, 3)$ and $u = (128, 256, 512)$. We find out $f = 1$ and $u = 512$ give the best performance for $direct$, $tile$, rbf, and $theory$. So we use them for every model for a fair comparison.

Spatial Context Modeling Grid search is used for hyperparameter tuning and the best performance of different grid cell models is obtain when $d^{(v)} = 64$, $d^{(x)} = 64$, $S = 64$, and $\lambda_{min} = 10$. We set $\lambda_{max} = 10k$ based on the maximum displacement between context points and center points to make the space encoding unique. As for multiple baseline models, grid search is used again to obtain the best model. The best model hyperparameters are shown in () besides the model names in Table 2. Note that both rbf and $scaled\_rbf$ achieve the best performance with $M = 100$.

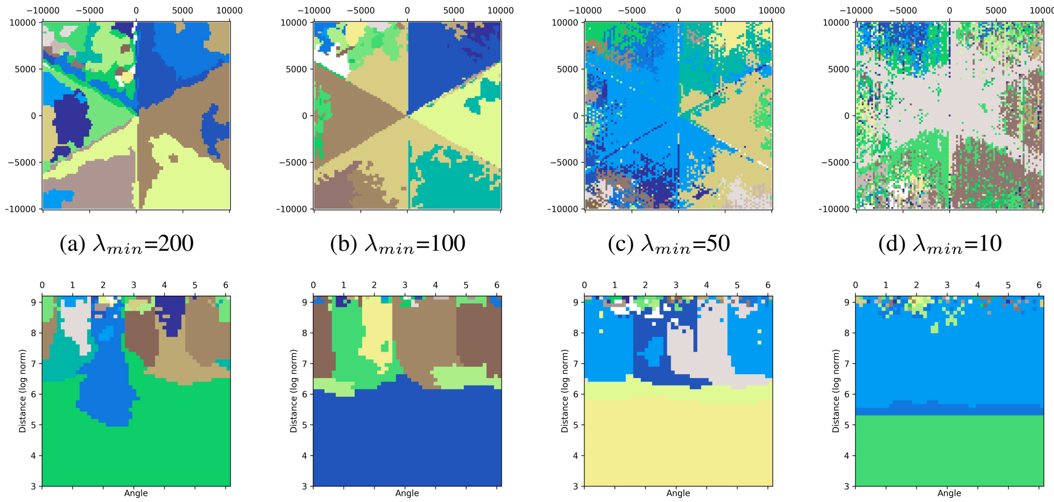
A.3 Firing Pattern for the Neurons



(a) $direct$ (b) $tile$ (c=500) (c) $wrap$ (h=3,o=512) (d) rbf (σ=1k) (e) $theory$ (λ_{min}=1k) (f) $theory$ (λ_{min}=50)

Figure 4: The firing pattern for the first 8 neurons (out of 64) given different encoders in location modeling.

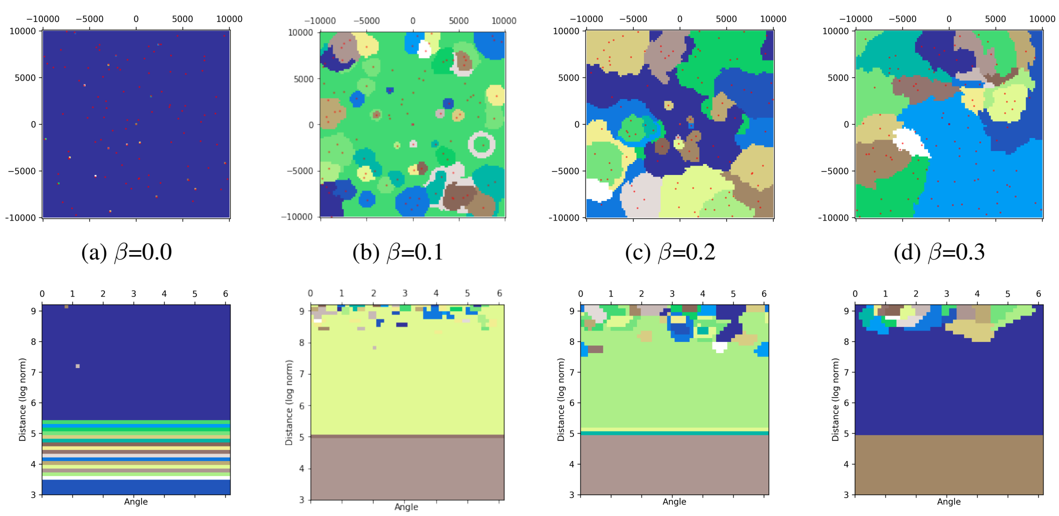
A.4 Embedding clustering of RBF and theory models



(a) λ_{min}=200 (b) λ_{min}=100 (c) λ_{min}=50 (d) λ_{min}=10

(e) λ_{min}=200 (f) λ_{min}=100 (g) λ_{min}=50 (h) λ_{min}=10

Figure 5: Embedding clustering in the original space of (a)(b)(c)(d) *theory* with different λ_{min}, but the same $\lambda_{max} = 10k$ and $S = 64$. (e)(f)(g)(h) are the embedding clustering results of the same models in the polar-distance space. All models use 1 hidden ReLU layers of 512 neurons.



(a) β=0.0 (b) β=0.1 (c) β=0.2 (d) β=0.3

(e) β=0.0 (f) β=0.1 (g) β=0.2 (h) β=0.3

Figure 6: Embedding clustering of RBF models with different kernel rescalar factor β (a)(b)(c)(d) in the original space; (e)(f)(g)(h) in the polar-distance space. Here β=0.0 indicates the original RBF model. All models use σ=10m as the basic kernel size and 1 hidden ReLU layers of 512 neurons.