Noname manuscript No.
(will be inserted by the editor)

Boosting House Price Predictions using Geo-Spatial Network Embedding

Sarkar Snigdha Sarathi Das · Mohammed Eunus Ali · Yuan-Fang Li · Yong-Bin Kang · Timos Sellis

the date of receipt and acceptance should be inserted later

Abstract Real estate contributes significantly to all major economies around the world. In particular, house prices have a direct impact on stakeholders, ranging from house buyers to financing companies. Thus, a plethora of techniques have been developed for real estate price prediction. Most of the existing techniques rely on different house features to build a variety of prediction models to predict house prices. Perceiving the effect of spatial dependence on house prices, some later works focused on introducing spatial regression models for improving prediction performance. However, they fail to take into account the geo-spatial context of the neighborhood amenities such as how close a house is to a train station, or a highly-ranked school, or a shopping center. Such contextual information may play a vital role in users' interests in a house and thereby has a direct influence on its price. In this paper, we propose to leverage the concept of graph neural networks to capture the geo-spatial context of the neighborhood of a house. In particular, we present a novel method, the Geo-Spatial Network Embedding (GSNE), that learns the embeddings of houses and various types of Points of Interest (POIs) in the form of multipartite networks, where the houses and the POIs are represented as attributed nodes and the relationships between them as edges. Extensive experiments with a large number of regression techniques show that the embeddings produced by our proposed GSNE technique consistently and significantly improve the performance of the house price prediction task regardless of the downstream regression model. Relevant source code for GSNE is available at: `https://github.com/sarathismg/gsne`.

Keywords Geo-spatial network embedding, graph neural networks, real estate queries, house-price predictions

Sarkar Snigdha Sarathi Das
Pennsylvania State University (Work done while at Bangladesh University of Engineering and Technology)
E-mail: sfd5525@psu.edu

Mohammed Eunus Ali
Bangladesh University of Engineering and Technology
E-mail: mohammed.eunus.ali@gmail.com

Yuan-Fang Li
Monash University E-mail: yuanfang.li@monash.edu

Yong-Bin Kang
Swinburne University of Technology E-mail: ykang@swin.edu.au

Timos Sellis
Facebook, USA (Work done while at Swinburne University of Technology) E-mail: tsellis@fb.com

1 Introduction

The price of a house is one of the most critical factors in the decision-making process of buying a house. Determining which house to buy is a challenging task as it is influenced by a multitude of other factors: features of houses as well as the complex geo-spatial relationships with their neighborhoods.

Precise prediction of house prices is of paramount importance to prospective buyers and homeowners who are planning to purchase or sell their houses. An effective prediction system can greatly assist the buyers to plan their funds ahead of purchase. Besides, property investors can leverage these prediction systems to plan their schemes for increasing revenue. Most importantly, a house price prediction system places homeowners and customers in a level playing field with investors, developers, real estate companies, mortgages, and insurance companies by providing them with reasonable price estimates. Zillow's *Zestimate* [1] is a prominent example of such a prediction scheme that has helped even first time customers to get a clear picture of the U.S. housing market over the years. Thus, the task of house price prediction has received significant attention in both academia and stakeholders for decades. Over the years, researchers have used different techniques to build effective models for house price predictions. For example, typical Hedonic Price models [38,43] have been extensively studied to model the relationship between prices and housing features. While the early models based on ordinary least square (OLS) context did not include spatial awareness, researchers gradually realized the impact of regional submarkets in house price prediction. To incorporate the locational influence, several works [7, 18,16,6] used spatial statistical methods since simple hedonic models are not much effective in handling spatial dependence in regression residuals. However, these spatial statistical methods require explicit feature engineering by domain experts. Following the success of machine learning in different fields, in recent years, SVM [51], Convolutional Neural Networks [41], and Recurrent Neural Networks [13] have been employed to better capture the preferences for more accurate house price prediction. House images have also been leveraged for better price estimation [55]. Unlike OLS based hedonic models, these modern learning algorithms can effectively capture the spatial dependence from location attributes which obviates the need of explicit feature engineering.

Although most of the recent prior works leverage the detailed housing and location features, they overlook the geo-spatial contexts such as "how close is this house to the train station?", or "is there any good school in walking distance from the house?". These geo-spatial contexts based on neighbourhood facilities can greatly influence user preferences on buying a house and hence can be key determining factors for the price of the house. To illustrate, suppose that there are two houses with the same set of features such as the number of bedrooms, house areas, etc., in the same suburb. However, there may have variations in their prices. For example, a house next to the train station will likely have a much higher price than the house which is far away (e.g. 3km distance) from the train station. Similarly, houses in a certain suburb with good schools in the neighborhood and a train station for commuting to the city will have higher prices than those houses in a nearby suburb that does not have a good school and a train station in its neighborhood. Hence, points of interests (POIs) such as train stations, schools, shopping centers, etc. in the neighborhood can play key roles in determining the house prices. To the best of our knowledge, none of the prior works capture the important features related to neighborhood POIs and their relationship with houses.

It is not straightforward to capture complex latent interactions between houses and POIs as it involves connectivity among different entities (e.g. how close a house is to the train station) as well as heterogeneous sets of features of these entities (e.g. how good a school is). Recently, an

[1] https://www.zillow.com/how-much-is-my-home-worth/

approach by [24] used satellite image, taxi mobility data, and the existence of different categories of point of interests for generating an embedding for a region, which are later used to a get a coarse outline of price distribution per sqft for houses in that region. Though this work can capture the regional features using complex sets of data, they neither consider detailed houses and POIs features nor the relationships between neighborhood POIs and the corresponding house, which is our main focus in this paper.

In this paper, we propose Geo-Spatial Network Embedding (GSNE) that accurately captures highly useful spatial features (both connectivity and the content) of key neighborhood POIs such as schools, train stations, etc. and their relations with the houses. We leverage the key concept of graph embedding that essentially learns low dimensional feature representations of a given attributed graph. GSNE employs Gaussian-based embedding methods [4, 23, 56] as they have been shown to be effective and robust against noises and uncertainties that are inherent in many real-world graphs such as house-neighborhood networks in real estate.

We propose to represent houses and their neighborhood POIs as a multipartite graph, in which nodes of different partitions (types) represent houses and POIs (e.g. regions and schools), and edges represent the spatial proximity relation between nodes. In our case, the graph is attributed and weighted, where nodes are attributed with their own features and edge weights represent the distance between two nodes. The key intuition of our proposed approach, GSNE, is to project the nodes in a *spatial network* into a Gaussian feature space.

Prior works in Gaussian-based network embeddings [4, 23, 56] only consider homogeneous or bipartite network, and it is not straightforward to embed heterogeneous multipartite networks in the same Gaussian space. The heterogeneity poses several challenges that include projecting different categories of node features into the same Gaussian space and developing effective sampling strategies and training schemes. We address all of these challenges in our proposed GSNE framework.

We evaluate our GSNE framework on the house price prediction task on a large real-estate and POI datasets of Melbourne, Australia. We concatenate the learned embedded vectors from GSNE, which capture essential spatial information about houses and their neighborhood context, with the raw house features vectors as features to predict house prices. Compared with raw features only, the concatenated features achieve the best prediction performance on a large number of regression models, demonstrating the effectiveness and robustness of our GSNE model.

In summary, our contributions are as follows:

- We propose a novel geo-spatial network embedding (GSNE) framework that can accurately capture the geo-spatial neighborhood context in terms of different types of POI and their features, and the relationships among these POIs in a weighted, attributed multipartite graph.
- We adopt and extend the Gaussian embedding methods to realize our GSNE framework, which is highly efficient and can work with heterogeneous types of nodes and features.
- Our comprehensive evaluation on a large real-estate dataset shows that for the house prediction task, combining geo-spatial embedded vectors learned by GSNE with the housing features results in consistently better prediction performance than raw feature only, regardless of the downstream regression model.

2 Related Work

In this section, we primarily discuss the existing works on house price predictions. Based on the working methodologies, we divide these works three categories: housing feature centric traditional approaches, machine learning based approaches, and location centric approaches, which

are presented in Section 2.1. Later we discuss major existing works on networking embedding in Section 2.2

2.1 House Price Predictions

Housing Feature-Centric Traditional Approaches: Most of the earlier price prediction models were based on Hedonic Regression [43]. Later this model has been extensively studied predicting prices of different areas and analyzing the effects of different factors [47,54,29,37]. In Hedonic price model, houses are considered as aggregation of different attributes, where customers purchase this package of bundled attributes. Although it simplifies the prediction task, there are some notable shortcomings. It has been found that hedonic price coefficients of some attributes are not stable between locations, property types and age [19]. Furthermore, issues like model specification procedures, independent variable interactions, non-linearity and outlier data points inhibit price prediction performance in hedonic price models [31]. Genetic algorithms have also been used in the study of house price prediction problem. In [36], the authors used a hybrid of genetic algorithm and SVM to predict house prices from different sets of features. In another work, [33] studied the potential of genetic algorithm in this problem domain. They also study the effect of geographical location from the viewpoint of genetic algorithm. In a recent work [35] used evolutionary polynomial regression to model house price prediction.

Machine Learning (ML) Approaches: Following the success of ML models in different prediction tasks, researchers started to employ different machine learning techniques for estimating housing prices. [50] and [30] used SVM based regression for determinig the house price.[52] used Lasso and Ridge Regression for predicting house prices. In [31], the authors found that Artifical Neural Network (ANN) based model outperforms hedonic price models in out of sample predictions. Later, in another study [42], the authors experimented with a wide varieties of ML techniques that include Artificial Neural Networks (ANN), AdaBoost, Random forest, Gradient boosted trees, Multi Layer Perceptron, and Ensemble learning algorithms. They found that Gradient boosted trees yield the best performance in predicting house prices. Researchers also used modern deep learning based approaches to improve house price prediction performance. In [41], the authors used convolutional neural network (CNN) for feature selection as well as price prediction. [17] compared multi level modeling (MLM) approaches with ANN and found the MLM methods to be much superior compared to ANN. In another recent work [55], the authors utilized property images alongside original tabular features, where they used CNN to extract features from those images and combined them with the transformed tabular features. They achieved an improved performance in price prediction by channeling this new set of features through XGBoost algorithm.

Location-Centric Approaches: As researchers realized the impact of locations on house prices, several works focused on the spatial awareness of the prediction models in order to amplify the location effect. Standard Hedonic regression models assume the residuals to be independent of each other, yet it is found that those residuals show significant spatial dependency [39]. Over the years, several techniques have been proposed to introduce spatial awareness in the prediction models. In [7], the authors experimented with a set of spatial submarkets defined by real estate appraisers. They compared different approaches of including neighbouring properties' residuals, separate submarket equations, etc. to take spatial dependence into account. On the other hand, [19] found that prediction from a model with postcode dummies perform slightly better than separate equations for each postcode. Geo-statical approaches have also been taken in some works [16,3], where it was found to do well compared to ordinary least square (OLS) regression models. [5] found that geo-statistical model with dis-aggregated submarket variables performed

the best in predicting price while considering spatial dependency. On the other hand, [46] found that in neighbourhood level geo-statical models perform only slightly better compared to OLS model. Besides geo-statical approaches, lattice approaches have also been tried out. In [6], the authors found that in mass appraisal context, lattice models which include SAR (simultaneous autoregressive) and CAR (conditional autoregressive) models performed poorly compared to even simple OLS models which disregard spatial dependence. They concluded that including submarket variable in OLS context gives much better gain in accuracy compared to any other geo-statical or lattice methods. They also argued that it is much more practical given that hedonic models with submarket dummies are much easier to implement compared to geo-statical or lattice approaches while also giving better gains. In [9], the authors experimented with OLS with location variables, geo-statical and several spatial statistics methods. They found that when accounted for the neighbouring residuals, all of those models showed similar results. [18] on the other hand, showed that property characteristics as well as cartesian coordinates and submarket dummies were able to capture most of spatial dependence and gives significant improvement in prediction accuracy. Similar models have also been used in other research domains to reveal spatial relations. [10] investigated different sparse structures for vector autoregressive model with LASSO framework for wind power forecasting. On the other hand, [11] proposed learning artificial neural network with respect to entropy based criteria for resolving temporal and spatial autocorrelation in forecasting renewable energy. Again, [21] proposed using Tucker tensor decomposition for detecting community structure in temporal network. [15] also leveraged similar technique to extract spatio-temporal autocorrelation in renewable energy forecasting.

In another work, [34] considered parametric and semi-parametric spatial hedonic model variants to capture spatial auto-correlation. [44] recently proposed to use Singular Value Decomposition and kriged its residual to better capture the spatial correlation. On the other hand, an unpublished technical report by [20] partitioned their dataset for different task definitions based on different schemes such as distance to station, schools, etc and used multi-task learning approach for the partitioned dataset. A major limitation of these approaches is that it requires in-depth domain analysis for choosing the submarket definition/partitioning scheme as effective partitioning scheme vary widely from dataset to dataset. Furthermore, to induce spatial lag in price prediction, traditional spatial statistics (e.g. Geostatical, SAR, CAR) approaches relied on heavy amount of feature engineering, which is required to be done by real estate appraisers.

Recently, [24] used multimodal data such as Satellite images, taxi mobility data, and categories of point of interests for generating embedding for different grid-partitioned region, which are later used to find overall price per sqft for houses in that region. This approach only provided an overview of the region and considers price per sqft. of houses, which is a coarse outline of price distribution. None of these methods could capture the intrinsic relationships between neighborhood POIs and the corresponding house, which is our key contribution of this paper.

2.2 Network Embedding

To represent complex high dimensional network information in a low dimensional feature space, several techniques have been adopted in recent times [8]. Earlier works used random walk based methods to encode these structural information. [40] first proposed short random walks and apply neural language modeling techniques for node representation. Later [22] proposed to improve [40] by biased random walking for effective exploration of neighborhood. [26] used spectral graph convolution for learning network representation. Later the authors used this graph convolution as encoder for a variational autoencoder [27]. To further improve the performance, [48] exploited masked self-attentional layers. On the other hand, [45] proposed a different approach where prox-

imity objectives were leveraged for embedding both global and local network structures. In some recent works [4,56,23], Gaussian embedding has been exploited to handle the uncertainties in network embedding. However, these Gaussian embedding methods only works with homogeneous or bipartite networks.

In this paper, we propose GSNE to incorporate network embedding in geo-spatial network. We extend the notion of network embedding to handle a wide variety of POIs having completely different sets of features. Also, by mapping the POI nodes in a high dimensional global embedding space, GSNE generates POI representations that also incorporate the features of neighborhood POIs. GSNE preserves both local and global structures, while dealing with the high degree of uncertainties in these networks originating from irregular structures. These properties of GSNE make it useful for inducing spatial awareness in the representation of POIs.

3 Methodology

In this section, we present our proposed geo-spatial network embedding (GSNE) approach. Since GSNE utilizes the notion of network embedding, given the housing data, GSNE converts the house and POI data into a geo-spatial network that is represented as an attributed, multipartite network. This attributed, multipartite network is then channeled through a neural network embedding pipeline, which embeds the nodes of the network in a Gaussian feature space, as inspired by previous work [4,23,56]. The high-level architecture of GSNE is depicted in Figure 1, which will be discussed in subsequent sections. Next, we discuss the problem formulation of geo-spatial network embedding (Section 3.1). Then we present the details of heterogeneous node attribute encoding (Section 3.2). After that we present the structure embedding learning through both first- and second-order proximity (Section 3.3). We discuss our model optimization (Section 3.4), which will be followed by model training details for GSNE (Section 3.5).

3.1 Problem Definition

Let $G = \{\mathcal{V}, \mathcal{X}, E, W\}$ represent our attributed, weighted, multipartite geo-spatial network, where, $\mathcal{V}$, $\mathcal{X}$, E, and W represent nodes (vertices), node attribute matrices, edges, and edge weights, respectively. The set of nodes $\mathcal{V}$ comprises K mutually exclusive subsets of nodes, i.e., $\mathcal{V} = V_1 \cup V_2 \cup \ldots \cup V_K$, where each subset V_k denotes a partition of nodes of a single type of POI (e.g. school or train Station), and $V_{k_1} \cap V_{k_2} = \emptyset$ where $k_1, k_2 \in \{1, 2, \ldots, K\}$ and $k_1 \neq k_2$. The set of attribute matrices is represented as $\mathcal{X} = \{\boldsymbol{X}_1, \boldsymbol{X}_2, \ldots, \boldsymbol{X}_K\}$, such that each $\boldsymbol{X}_k \in \mathbb{R}^{D_k \times |V_k|}$ represents the attribute matrix for node partition V_k, where D_k is the dimension of the attributes and $|V_k|$ represents the number of nodes in this partition. Since these attributes can vary greatly from POI to POI (e.g. for schools, number of students, rankings etc. while for train stations, train frequency, time to reach other stations etc.), the dimension of the $\mathcal{X}_k$ will be different for each of the POIs. Again, $E = \bigcup_{k_1,k_2} E_{k_1k_2}$, where $k_1, k_2 \in \{1, 2, \ldots, K\}$, $k_1 \neq k_2$ and each $E_{k_1k_2}$ represents the set of edges between the nodes in partition V_{k_1} and nodes in partition V_{k_2}. Also, $W \in \mathbb{R}^{|E|}$ is a weight vector such that each edge $e_{ij} \in E$ between a pair of nodes i and j has a weight $w_{ij} \in W$. Note that without the loss of generality we use the terms 'attributes' and 'features' interchangeably.

We exploit the spatial information (latitude, longitude) of houses and POIs to generate the set of edges E. Intuitively, an edge is created between a pair of nodes only if they are located less than a certain user-specified distance threshold δ_{max} .

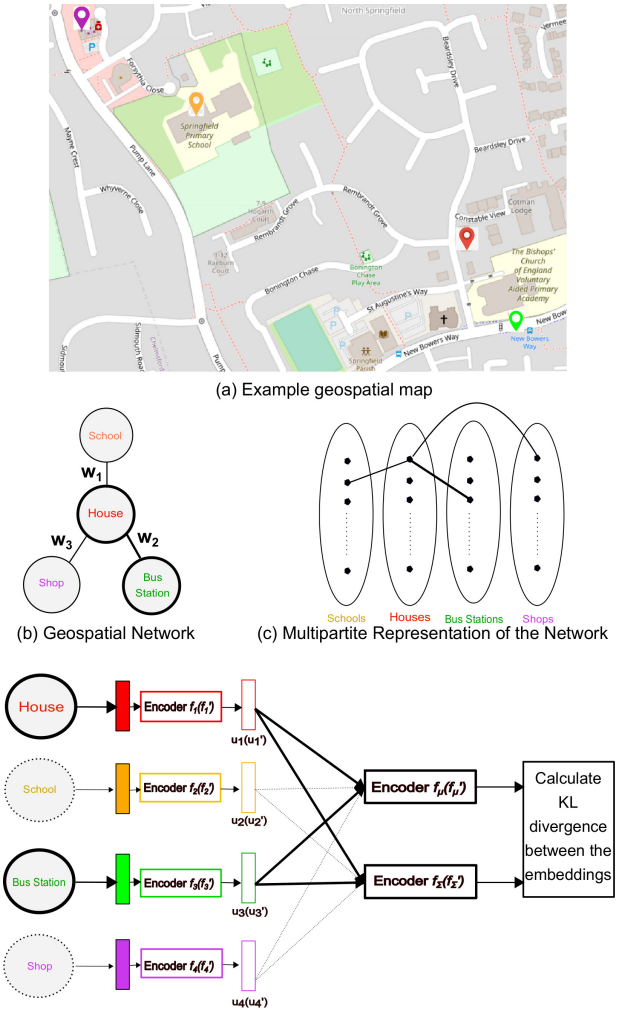


Fig. 1: (a) An example showing the location of a house (red location pin) and its nearby school (yellow pin), shop(purple pin), and bus-station (green pin) on the map. (b) Geo-spatial network centred around the house in (a) where w_1, w_2, w_3 represents the weights of the corresponding undirected edges. (c) Illustration of the multipartite nature of the network, where each node partition represents either houses or a specific category of POI nodes i.e. schools, shops, or bus stations, (d) The overall architecture of GSNE, where the node attribute encoding is represented using $\{f_1, f_2, f_3, f_4\}$, $\{u_1, u_2, u_3, u_4\}$, and $\{f_\mu, f_\Sigma\}$, and the second-order context encoding is represented as $\{f'_1, f'_2, f'_3, f'_4\}$, $\{u'_1, u'_2, u'_3, u'_4\}$, and $\{f'_\mu, f'_\Sigma\}$. Here the house and bus station nodes in (a) are bolded to represent sampled house-bus station edge and how it is channeled through the GSNE framework.

For each edge $e_{ij} \in E$, its weight w_{ij} is calculated as $w_{ij} = \frac{1}{\delta(i,j)}$, where $\delta(i,j)$ represents the euclidean distance between the two nodes. Moreover, we design our network to be undirected, thus $e_{ij} = e_{ji}$ and $w_{ij} = w_{ji}$.

For our geo-spatial network, the partition V_1 denotes houses, and the other partitions $V_2, \ldots, V_K$ denote different types of POIs such as regions, schools and train stations. Figure 1 (a), (b), and (c)

Table 1: Basic notation

Symbol	Description
G	$G = \{\mathcal{V}, \mathcal{X}, E, W\}$ is our attributed, weighted, undirected geo-spatial network
$\mathcal{V}$	The set of nodes $\mathcal{V} = V_1 \cup V_2 \cup \ldots \cup V_K$ having K subsets of nodes
$\mathcal{X}$	The set of attribute matrices $\mathcal{X} = \{\boldsymbol{X}_1, \boldsymbol{X}_2, \ldots, \boldsymbol{X}_K\}$, each $\boldsymbol{X}_k \in \mathcal{X}$ represents the attribute matrix of node partition V_k
E	Set of edges of geo-spatial network
W	Weight vector of the edges
w_{ij}	$w_{ij} = \frac{1}{\delta(i,j)}$, $\delta(i,j)$ represents the distance between nodes $i, j \in \mathcal{V}$
L	Output embedding dimension
l	Heterogeneous node attribute encoder output dimension
$\{\boldsymbol{W}_{k_1}, \boldsymbol{W}_{k_2}\}$ $(\{\boldsymbol{W}'_{k_1}, \boldsymbol{W}'_{k_2}\})$	Weights of the attribute encoder of node partition k for first(second) order proximity
$\{\boldsymbol{b}_{k_1}, \boldsymbol{b}_{k_2}\}$ $(\{\boldsymbol{b}'_{k_1}, \boldsymbol{b}'_{k_2}\})$	Bias vectors of the attribute encoder of node partition k for first(second) order proximity
$f_k(f'_k)$	Attribute encoder for first(second) order proximity
$\boldsymbol{u_i}(\boldsymbol{u'_i})$	Intermediate representation of node i after attribute encoding stage for first(second) order proximity
$\boldsymbol{W}_\mu, \boldsymbol{W}_\Sigma (\boldsymbol{W}'_\mu, \boldsymbol{W}'_\Sigma)$	Mean and covariance encoder weights for first(second) order proximity
$\boldsymbol{b}_\mu, \boldsymbol{b}_\Sigma (\boldsymbol{b}'_\mu, \boldsymbol{b}'_\Sigma)$	Mean and covariance encoder biases for first(second) order proximity
$f_\mu, f_\Sigma (f'_\mu, f'_\Sigma)$	Mean and covariance encoder for first(second) order proximity
$\boldsymbol{\mu_i}, \boldsymbol{\Sigma_i}$ $(\boldsymbol{\mu'_i}, \boldsymbol{\Sigma'_i})$	Mean and covariance embedding representation of node i for first order proximity(second order proximity)
$\boldsymbol{h_i}(\boldsymbol{h'_i})$	$\boldsymbol{h_i} = \mathcal{N}(\boldsymbol{\mu_i}, \boldsymbol{\Sigma_i})$ is the first order ($\boldsymbol{h'_i} = \mathcal{N}(\boldsymbol{\mu'_i}, \boldsymbol{\Sigma'_i})$ is the second order) proximity embedding of node i
$D(\boldsymbol{h_i}, \boldsymbol{h_j})$	Asymmetric KL-divergence between embeddings $\boldsymbol{h_i}$ and $\boldsymbol{h_j}$
$d(\boldsymbol{h_i}, \boldsymbol{h_j})$	Symmetric KL-divergence between embeddings $\boldsymbol{h_i}$ and $\boldsymbol{h_j}$

show an example of geo-spatial entities, geo-spatial network, and the multipartite representation of the network, respectively.

The GSNE model learns low-dimensional Gaussian embeddings $\boldsymbol{h}_i$ for each node $v_i \in \mathcal{V}$, i.e. $\boldsymbol{h}_i = \mathcal{N}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$, where $\boldsymbol{\mu}_i \in \mathbb{R}^L$, $\boldsymbol{\Sigma}_i \in \mathbb{R}^{L \times L}$, where L is the embedding dimension and $L \ll |\mathcal{V}|$ and $L \ll D_k$ for each $k \in 1, \ldots, K$. Intuitively, nodes that are similar to each other and are close in the original multipartite network are also close in the embedding space.

3.2 Heterogeneous Node Attribute Encoding

We employ neural network based encoders to learn the embeddings of node features of the K partitions

in the network G. A straightforward approach for encoding node features in a multipartite network is to concatenate feature vectors of different partitions with zero padding to create a *global* feature vector, which is then fed into a single encoder to produce the initial node

embeddings. This approach has been shown to work well for bipartite networks (where $K = 2$) in prior work [23]. However, this strategy may face issues in a complex multipartite network like ours, as the dimension of the combined feature vector becomes exceedingly large and sparse that it induces suboptimal model convergence as found in our experiments.

To tackle this challenge, GSNE uses separate encoders to handle different partitions of nodes, which is followed by a global Gaussian encoder that projects distinct types of POI nodes in the same Gaussian embedding space. By doing so, we solve the sparsity issue while projecting the final embedding into the same Gaussian embedding space. Specifically, we employ K different encoders $\{f_k \in \boldsymbol{X}_k \rightarrow \mathbb{R}^{l \times |V_k|}\}_{k \in \{1,2,\ldots,K\}}$, where each encoder f_k projects attributes $\boldsymbol{X}_k$ of nodes in partition V_k to l-dimensional embedding space $\mathbb{R}^{l \times |V_k|}$.

Each encoder f_k generates encoded representations $\boldsymbol{u}_i$ for a feature vector $\boldsymbol{x_i} \in \boldsymbol{X}_k$ using rectified linear units (ReLU) as follows:

$$\begin{aligned} \boldsymbol{u}_i &= f_k(\boldsymbol{x_i}) = ReLU(\boldsymbol{W}_{k_2}\boldsymbol{z}_i + \boldsymbol{b}_{k_2}), \text{ where} \\ \boldsymbol{z}_i &= ReLU(\boldsymbol{W}_{k_1}\boldsymbol{x_i} + \boldsymbol{b}_{k_1}), \end{aligned} \tag{1}$$

where $\boldsymbol{W}_k = \{\boldsymbol{W}_{k_1}, \boldsymbol{W}_{k_2}\}$ are the weight matrices and $\boldsymbol{b}_k = \{\boldsymbol{b}_{k_1}, \boldsymbol{b}_{k_2}\}$ are the bias vectors for partition $V_k \in \mathcal{V}$. Here, $\boldsymbol{W}_{k_1} \in \mathbb{R}^{l' \times D_k}$, $\boldsymbol{W}_{k_2} \in \mathbb{R}^{l \times l'}$, $\boldsymbol{b}_{k_1} \in \mathbb{R}^{l'}$, and $\boldsymbol{b}_{k_2} \in \mathbb{R}^{l}$. Here, l' and l denote the first and second (ouput) layer dimensions of heterogeneous node attribute encoder respectively.

The attribute embeddings $\boldsymbol{u}_i$ are then channeled through a Gaussian encoder to obtain the final Gaussian embedding $\mathcal{N}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$. Although we use separate encoders for each node type to handle different partitions, we employ a common global Gaussian encoder (f_μ, f_Σ) for learning the final representations of all nodes. As a result, the common Gaussian encoder allows us project all nodes in the same L-dimensional Gaussian space. Since we are projecting nodes of different partitions in the same Gaussian space, we are essentially allowing the nodes to know their own neighbourhood while learning from the global network structure. Here we generate Gaussian embedding $\boldsymbol{h}_i = (\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$ from the intermediate representations as follows:

$$\boldsymbol{\mu}_i = f_\mu(\boldsymbol{u}_i) = ReLU(\boldsymbol{W}_\mu \boldsymbol{u}_i + \boldsymbol{b}_\mu) \tag{2}$$

$$\boldsymbol{\Sigma}_i = f_\Sigma(\boldsymbol{u}_i) = ELU(\boldsymbol{W}_\Sigma \boldsymbol{u}_i + \boldsymbol{b}_\Sigma) + 1 \tag{3}$$

where $\boldsymbol{W}_\mu, \boldsymbol{W}_\Sigma \in \mathbb{R}^{L \times l}$ and $\boldsymbol{b}_\mu, \boldsymbol{b}_\Sigma \in \mathbb{R}^{L}$, and the exponential linear unit (ELU) [14] is the activation function in the covariance encoder.

3.3 Network Structure Embedding

GSNE learns from the structure of the multipartite network by considering both first-order and second-order proximity between each pair of connected partitions.

First-order proximity learns from the direct connections of a pair of nodes across partitions, while second-order proximity learns from nodes that are connected through an intermediate node. The aim of this learning is to capture local neighbourhood context as well as global connectivity in the whole network. In the following subsections we describe how such structural information is captured in GSNE.

3.3.1 First-order Proximity in Geospatial Network

As indicated in previous work [45,49], the *first-order proximity* represents the local pairwise proximity between two nodes. In other words, if an edge connects a pair of nodes (i, j), they have a positive first-order proximity. While in any network it is intuitive to generate similar embeddings for two nodes with positive first-order proximity, in the geospatial domain it has additional significance. Consider the following scenario. If a house i has an edge with a nearby school j, the first-order proximity essentially tries to generate similar embeddings for them. If this same house i also has an edge with a train station k in its neighbourhood, it also tries to keep them close in the embedding space. The outcome is that, the first-order proximity lets the model learn from the geospatial connectivity of a node's *local* neighbourhood amenities. Thus, the model essentially learns to embed the neighbourhood context of a house, which customers may likely take into consideration while making purchase decisions of a house and thus influence the price of the house.

3.3.2 Second-order Proximity in Geospatial Network

While the first-order proximity can effectively capture the local neighbourhood context of a node, it fails to capture latent similarity of two nodes beyond their immediate neighbourhood when there are no direct edges between them.

However, consider the following scenario. If two houses are located in two geographically distant locations, yet both of them are connected to the same highly-rated schools and transportation facilities (e.g. train stations) nearby, their geo-spatial embedding should also be similar.

The *second-order proximity* between two nodes in a network essentially capture the similarity between their neighbourhood network structure. Thus, to take advantage of this fact, GSNE uses second-order proximity so that geographically distant houses having similar neighbourhood structure are located closer in the embedding space. In other words, it helps the model learn the **global** neighbourhood connectivity of the network.

3.3.3 Proximity Objectives

We adopt the widely used strategy of LINE [45] to compute our first- and second-order proximities. As we embed nodes as Gaussian distributions, we employ KL-divergence as our dissimilarity measure. Since our network is undirected and KL-divergence is asymmetric in nature, we consider both directions of the edges. i.e. we compute $D(\boldsymbol{h}_i, \boldsymbol{h}_j) + \mathrm{D}(\boldsymbol{h}_j, \boldsymbol{h}_i)$ the KL-divergence from both directions as suggested in previous work [4]. Let $\boldsymbol{h}_i$ and $\boldsymbol{h}_j$ represent the Gaussian representations of nodes i and j, and $d(h_i, h_j) = D(\boldsymbol{h}_i, \boldsymbol{h}_j) + \mathrm{D}(\boldsymbol{h}_j, \boldsymbol{h}_i)$ represent their KL-divergence dissimilarity measure.

First-Order Proximity: For each $(i, j) \in E$, we take the joint probability between i and j as:

$$p_1(i, j) = \frac{1}{1 + \exp\left(d(\boldsymbol{h}_i, \boldsymbol{h}_j)\right)} \tag{4}$$

With this joint probability, we take the first-order proximity objective O_1 as follows.

$$O_1 = -\sum_{(i,j)\in E} w_{ij} \log p_1(i, j) \tag{5}$$

where w_{ij} is the edge weight.

Second-order Proximity: For second-order proximity, each node $i \in \mathcal{V}$ requires both attribute embedding $\boldsymbol{h}_i$ and context embedding $\boldsymbol{h}'_i$, which is treated as a context of other nodes.

However, the traditional definition of second-order proximity [45] is defined only on homogeneous and bipartite networks, but not multipartite networks like ours. This causes a problem in negative node sampling (discussed in Section 3.4) which inhibits the model from convergence. Thus, we modify the second-order proximity as follows.

For each directed edge $(i, j) \in E$ (undirected edges can be treated as two edges in opposite directions), where $i \in V_p$ and $j \in V_q$ are nodes from two partitions V_p and V_q respectively, the probability of the context of node j generated by node i is:

$$p_2(j|i) = \frac{\exp\left(-d(\boldsymbol{h}_i, \boldsymbol{h}'_j)\right)}{\Sigma_{\hat{i} \in V_q} \exp\left(-d(\boldsymbol{h}_i, \boldsymbol{h}'_{\hat{i}})\right)} \tag{6}$$

where $\hat{i}$ range over all nodes in partition V_q.

With this new definition of p_2, we define our second-order proximity objective as in LINE [45]:

$$O_2 = -\sum_{(i,j) \in E} w_{ij} \log p_2(j|i) \tag{7}$$

In order to generate the context embedding $\boldsymbol{h}'_i = (\boldsymbol{\mu}'_i, \boldsymbol{\Sigma}'_i)$, we need a separate set of encoders. We define $f'_k = (\boldsymbol{W}'_k, \boldsymbol{b}'_k)$ where $\boldsymbol{W}'_k = \{\boldsymbol{W}'_{k_1}, \boldsymbol{W}'_{k_2}\}$ and $\boldsymbol{b}'_k = \{\boldsymbol{b}'_{k_1}, \boldsymbol{b}'_{k_2}\}$ are the weight matrices and bias vectors for partition $V_k \in \mathcal{V}$. For Gaussian context embedding, with model parameters $f'_\mu = (\boldsymbol{W}'_\mu, \boldsymbol{b}'_\mu)$, $f'_\Sigma = (\boldsymbol{W}'_\Sigma, \boldsymbol{b}'_\Sigma)$, we generate context embedding as following:

$$\begin{aligned}
\boldsymbol{u}'_i &= f'_k(\boldsymbol{x}_i) = ReLU(\boldsymbol{W}'_{k_2} \boldsymbol{z}'_i + \boldsymbol{b}'_{k_2}) \text{ where} && (8)\\
\boldsymbol{z}'_i &= ReLU(\boldsymbol{W}'_{k_1} \boldsymbol{x}_i + \boldsymbol{b}'_{k_1}) \\
\boldsymbol{\mu}'_i &= f'_\mu(\boldsymbol{u}'_i) = ReLU(\boldsymbol{W}'_\mu \boldsymbol{u}'_i + \boldsymbol{b}'_\mu) && (9)\\
\boldsymbol{\Sigma}'_i &= f'_\Sigma(\boldsymbol{u}'_i) = ELU(\boldsymbol{W}'_\Sigma \boldsymbol{u}'_i + \boldsymbol{b}'_\Sigma) + 1 && (10)
\end{aligned}$$

Similar to the encoders in Section 3.2, here $\boldsymbol{W}'_{k_1} \in \mathbb{R}^{l' \times D_k}$, $\boldsymbol{W}'_{k_2} \in \mathbb{R}^{l \times l'}$, $\boldsymbol{b}'_{k_1} \in \mathbb{R}^{l'}$, $\boldsymbol{b}'_{k_2} \in \mathbb{R}^{l}$, $\boldsymbol{W}'_\mu, \boldsymbol{W}'_\Sigma \in \mathbb{R}^{L \times l}$ and $\boldsymbol{b}'_\mu, \boldsymbol{b}'_\Sigma \in \mathbb{R}^{L}$

3.4 Model Optimization

Since the objective function in Eq. 7 requires the summation over entire set of nodes of the same type while calculating p_2, it is computationally prohibitive. To alleviate this issue, we use negative sampling [45]. With this negative sampling technique employed, objective O_2 becomes:

$$\begin{aligned}
O_2 = \sum_{(i,j) \in E, i \in V_p, j \in V_q} & \Big(\log\ \sigma(-d(\boldsymbol{h}_i, \boldsymbol{h}'_j)) + \\
& \sum_{n=1}^{N} \mathbb{E}_{v_n \sim P_{qn}(v)} \log\ \sigma(d(\boldsymbol{h}_i, \boldsymbol{h}'_{v_n})) \Big)
\end{aligned} \tag{11}$$

Here the first term optimizes the positive edges whereas the second term is concerned with negative edges drawn from a noise distribution $P_{qn}(v) \propto d_v^{\frac{3}{4}}$, where $v \in V_q$ and d_v is the degree of the node(since the network is undirected). We use this negative sampling strategy also in the calculation of O_1 in Eq. 5, which is similar to Eq.11 except $\boldsymbol{h}'$ in the equation will be changed to $\boldsymbol{h}$.

When O_2 is optimized, the gradient gets multiplied by the edge weight. Since the edge weights are set to be the inverse of geographical distance, edges in our geo-spatial network may have high weight variance. This may induce exploding gradient during training phase. A naive solution to this problem is unwrapping an edge of weight w units into w-binary unweighted edges so that the whole graph can be regarded as unweighted. Yet, it is very inefficient from the memory perspective of the network.

To avoid exploding gradient without compromising memory efficiency, we use edge sampling from an alias table as in [45], where we efficiently sample positive edges according to the distribution of the weights of the edges using alias table.

3.5 Model Training

The training algorithm of GSNE is presented in Algorithm 1. Here, in lines 1-4, we first define and initialize the parameters of our network described previously. Lines 5-19 show the main training loop, where in each iteration we select an edge set $\mathcal{E} \subseteq E$ (line 6) where $\mathcal{E}$ represents connections between nodes between two partitions. Subsequently, we sample a batch of positive edges from $\mathcal{E}$ as in [45] (line 7). For each of the positive edges, we also sample N negative edges. (line 9). Later we calculate the Gaussian embeddings of the node pairs of sampled edges (lines 10-14). Finally, we calculate the batch loss (lines 15-18) and optimize through the back propagation and updating the parameters θ (line 19).

Algorithm 1: The GSNE algorithm

Input: Network $G = \{\mathcal{V}, \mathcal{X}, E, W\}$, proximity, number of negative samples N, batch size b, total iterations T
Output: Embedding $\boldsymbol{h}_i$ for each node $i \in \mathcal{V}$

1 Let $\theta = \{f_i\}_{i\in\{1,2,\ldots,K\}} \bigcup \{f_\mu, f_\Sigma\}$
2 **if** *proximity = second-order* **then**
3 $\theta = \theta \bigcup \{f'_i\}_{i\in\{1,2,\ldots,K\}} \bigcup \{f'_\mu, f'_\Sigma\}$
4 Initialize θ
5 **for** *iterations* $= 0 \rightarrow T$ **do**
6 **for** ***each*** *edge set* $\mathcal{E} \subseteq E$ **do**
7 $batch = FetchBatch(\mathcal{E}, b)$
8 **for** ***each*** *edge* $(i, j) \in batch$ **do**
9 $Neg(i) = NegativeSampling((i, j), N, \mathcal{E})$
10 Calculate $\boldsymbol{h}_i$ (Eq. 2, 3)
11 **if** *proximity = first-order* **then**
12 Calculate $\boldsymbol{h}_j, \{\boldsymbol{h}_{\nu_k}\}_{\nu_k \in Neg(i)}$ ▷ (Eq. 2, 3)
13 **else**
14 Calculate $\boldsymbol{h}'_j, \{\boldsymbol{h}'_{\nu_k}\}_{\nu_k \in Neg(i)}$ ▷ (Eq. 9, 10)
15 **if** *proximity = first-order* **then**
16 Calculate O_1 ▷ (Eq. 5)
17 **else**
18 Calculate O_2 ▷ (Eq. 11)
19 Update the parameters θ in the backpropagation stage

4 Experiments

To evaluate the efficacy of our Geo-spatial Network Embedding (GSNE) method, we apply it with a number of state-of-the-art regression models for the house price prediction task on a large real-estate and POI datasets of Melbourne, Australia. Specifically, our GSNE model is trained to obtain embeddings of houses. Then, we train each of the house price prediction (regression) models by concatenating our generated GSNE embeddings with the raw housing features (referred to as Raw + GSNE). We compare the prediction performance of our method (Raw + GSNE) against the same regression models trained on raw housing features only as described in Section 4.1 (referred to as "Raw"). To ensure the baseline, i.e., "Raw" method contains spatial lag in the modeling, location details are also included along with the core housing features.

As the downstream regression models, we have chosen some of the best regression models for house price prediction competition in Kaggle [1], the recent house prediction models in [52, 53, 42], and well known regression models such as LightGBM [25], XGBoost [12] and Gradient Boosting [42].

In the following, we first present the details of the dataset and the generation of geo-spatial network in Section 4.1. We then discuss our performance metrics for evaluating different algorithms in Section 4.3, followed by a discussion on the experimental setup in Section 4.4. We analyze our house price prediction results in Section 4.5, including an ablation study on the effect of various components in our method. Finally, a qualitative analysis of the embeddings are presented using visualizations in Section 4.6.

4.1 Dataset Description

We conducted our experiments on the house transaction records obtained from a real-estate web site[2] for Melbourne, which is the second largest city in Australia by population. We extracted a total of the 52,851 house transaction records of years from 2013 to 2015. Our dataset also includes the three types of POIs: *regions*, *schools*, and *train stations* and their corresponding features. Houses are situated in regions which capture the geographical contextual information about houses. Intuitively, information about nearby schools and train stations may influence house prices. Our dataset contains information of the 13,340 regions, 709 schools, and 218 train stations.

4.1.1 House and POI Features

Housing Features: Our dataset contains information about a wide range of housing features. In total, we consider **43** housing features for each house for in depth exploration of the effect of GSNE. To the best of our knowledge, none of the prior works considered such a wide range of feature sets in a large dataset like ours for house price prediction task. Although the dataset in Kaggle competition [1] has 86 features, it has only 3000 samples in total and lots of columns are highly sparse rendering only a few of those columns truly useful. Besides, no information regarding neighbourhood amenities is available in that dataset. In our dataset, each house record contains information ranging from basic housing features like number of bedrooms, number of bathrooms, number of parking spaces, location, type of property, etc. to detailed facility features like air-conditioning, balcony, city-view, river-view, swimming, tennis-court, etc. These features are listed in detail in Table 2.

[2] https://www.realestate.com.au/

Table 2: Housing Features

Number of bedrooms	Fireplace
Number of bathrooms	Fully fenced
Parking	Gas heating
Property type	Gym
Transaction date	Heating
Agency	Intercom
Latitude	Laundry
Longitude	Mountain
Air Conditioning	Park
Alarm	Swimming pool
Balconey	Renovated
BBQ	River view
City view	Rumpus room
Adjacency to schools	Sauna
Adjacency to shops	Study rooms
Adjacency to transport	Sun room
Courtyard	System heating
Number of dining rooms	Tennis court
Dish wash	Water views
Ducted	Wordrobe
Ensuite	Total additional features
Family rooms	

Region Features: Our dataset contains Melbourne region information at SA1 level[3]. SA1 is the most granular unit for the release of census data of Australia [2]. The SA1 data typically has a population of 200 to 800 people with an average of 400 people per region. For each region, our dataset contains comprehensive information about the number of residents, average age, median personal income, percentage of Australian citizens, educational qualification, median house rent, location as the centroid of the region, etc. Since these aspects can be useful for determining house prices, we consider all of them as the features for regions.

School Features: The schools in our dataset are attributed with the type of school (primary or secondary), school category by gender(single gender or co-ed), ranking, location, number of students, zone restrictions, number of students enrolled in Victorian Certificate of Education(VCE), percentage of students securing 40% marks, etc.[4]

Train Stations: The train stations in the dataset contain information about their location and average time to reach to other stations.[5]

4.1.2 Dataset Pre-processing

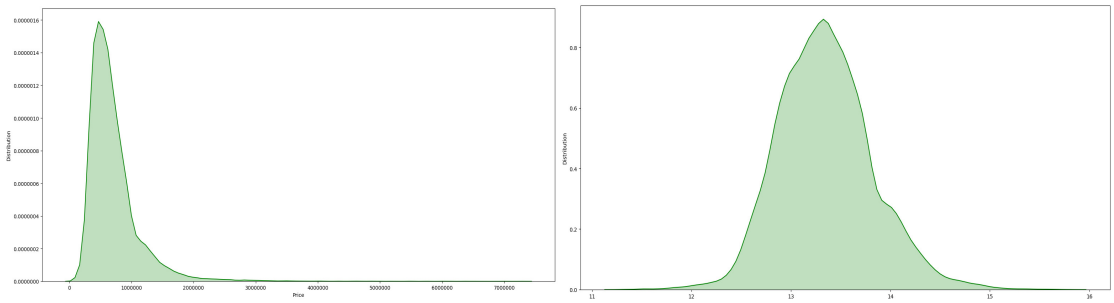
From Figure 2, we can see that the price distribution is skewed to the right. Since these skewed data may induce higher influence on the error calculation of the more expensive houses, we apply the widely-used log-normalization to the prices as done in previous works [20, 53]. All performance measures are calculated on these normalized prices.

For *train stations* and *schools*, we filled the missing values with the mean of the corresponding feature, since there are only 709 schools and 218 train stations. Categorical variables have been handled by one-hot encoding. For feature standardization, we use zero-mean, unit-variance on our dataset.

[3] https://www.abs.gov.au/

[4] https://bettereducation.com.au/

[5] http://developers.google.com/maps/



(a) Before log transformation (b) After log transformation

Fig. 2: Price distribution of the houses before and after log-normalization.

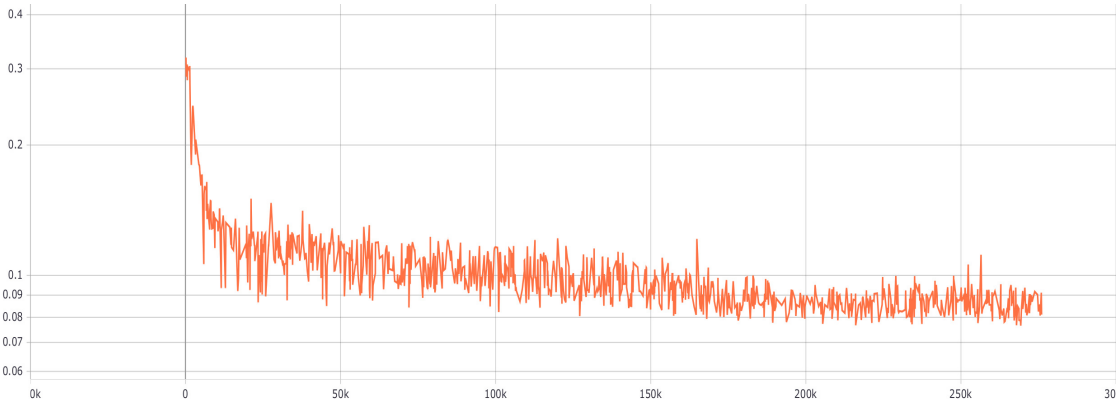


Fig. 3: Training Loss Curve for GSNE.

4.1.3 Geo-spatial Network Generation

From the dataset, we build a geo-spatial network using houses and the three different types of POIs: regions, schools, and train stations. We generate the network by considering the following edges: *House-Region, House-School, House-Train Station, and School-Train Station.* In House-Region edges, every house in a region will be connected to the corresponding region. For House-School edges, we connect a house to all schools which are located in one kilometer radius. If there is no school found within a kilometer range we connect the house to the nearest school. Similarly, we form the edges between Houses and Train Stations. Apart from these three types of house-POI connections, we also include *School-Train Station* edges as these can help to model the transport options to a school from a house via train. While our network is defined to be multi-partite, we allow *Train Station-Train Station* edges to keep the whole geo-spatial network connected. Weights for all types of edges are based on the Euclidean distance between the two nodes of the network. Since our primary focus is to create geo-spatial embeddings for house price prediction, we disregard the edges among POIs that seemingly do not have much impact in our problem domain.

4.2 Model Training

We have built the GSNE model by following the steps as described in Algorithm 1. Though most of the training steps are straightforward, the selection of a bipartite edge set $\mathcal{E} \subseteq E$ in each iteration is an important factor to consider here as it may impact model performance. We experiment with three strategies: (1) randomly selecting pairs of connected partitions, (2) iteratively alternating between pairs of connected partitions in every iteration, and (3) selecting a pair of connected partitions per 100 iterations and alternate. We find that strategy (2), iterative alternation between bipartite edge sets, is the strategy yielding the best result overall. Even though iterative alternation causes the loss to be jumpy, the overall trend of the loss decreases as shown in Figure 3. On the other hand, the other two strategies lead towards early convergence to local optima, resulting in higher training loss.

4.3 Performance Metrics

Following the prior works in this area, we use 'mean absolute error' (MAE) and 'root mean squared error' (RMSE) as our metrics to evaluate our embedding model.

They are defined as:

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|z_i - \hat{z}_i| \tag{12}$$

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(z_i - \hat{z}_i)^2} \tag{13}$$

Here z_i represents the ground truth price and $\hat{z}_i$ represents the predicted house price. N represents the number of samples. However, these metrics introduce a problem as higher prices influence the metrics much higher than the others[53]. To mitigate this issue, we use the *logarithmic* form of *sales (sold) price* in prediction as described in Section 4.1.2.

4.4 Experimental Setup and Model Building

All experiments were performed on a server with 16GB memory, and a 12GB NVIDIA Tesla P100 GPU.

The dimension of our Gaussian embedding is set as $L = 32$. The total number of iterations is set as $T = 3,00,000$. The batch size is set as $b = 128$, and the number of negative samples is set as $N = 5$. The above sets of parameters are chosen based on the empirical evaluation with our dataset. We divided our dataset into two parts by using stratified random sampling, into 80% for unsupervised training of the embedding model GSNE and 20% for testing the embedding model. Since the test set is completely unseen in the training phase of the model, a good performance on the test set usually indicates that the embedding model successfully generalizes to unseen nodes, i.e., the model is *inductive*.

4.5 House Price Prediction Results

We evaluate the effectiveness of our embeddings by using them as features to train various regression models. Specifically, we compare the performance of each regression model trained with

Table 3: Housing Price Prediction Performance Comparison. Best performed is bolded for each model.

Metric	Method	Lasso	Elastic Net	Kernel Ridge	Gradient Boosting	XGBoost (XGB)	LGBM	Avg.(KRR, GBoost, XGB)	Meta-model Stacking[53]
MAE	Raw	0.251	0.245	0.149	0.136	0.143	0.135	0.140	0.135
	Raw+ GSNE(1^{st})	0.220	0.216	0.141	0.128	0.133	0.129	0.130	0.129
	Raw+ GSNE(2^{nd})	0.247	0.241	0.137	0.126	0.133	**0.127**	0.130	0.129
	Raw+GSNE ($1^{st} + 2^{nd}$)	**0.209**	**0.205**	**0.135**	**0.125**	**0.132**	**0.127**	**0.128**	**0.128**
RMSE	Raw	0.333	0.331	0.206	0.195	0.200	0.190	0.197	0.190
	Raw+ GSNE(1^{st})	0.295	0.291	0.196	0.184	0.188	0.182	0.185	0.185
	Raw+ GSNE(2^{nd})	0.339	0.334	0.191	0.182	0.188	**0.180**	0.184	0.184
	Raw+GSNE ($1^{st} + 2^{nd}$)	**0.290**	**0.289**	**0.190**	**0.181**	**0.187**	**0.180**	**0.182**	**0.183**

two sets of features: with the house feature only (referred to as "Raw") and with the concatenation of the Raw features and our embeddings (referred to as "Raw+GSNE"). We also consider three variants of the embeddings: first-order proximity only (1^{st}), second-order proximity only (2^{nd}), and both the first- and second-order ($1^{st} + 2^{nd}$) proximities. In each case, we concatenate these embeddings with the original raw features and use them to train the downstream regression models.

To ensure our comparison baseline "Raw" also include sufficient spatial lag in it's price model, we add the location details for the houses. We take this as representative of spatially aware prediction model as in [18] since modern learning algorithms can effectively capture spatial dependence without any complex feature engineering required in OLS hedonic price models. We also experimented by adding postcode dummies as in [19], although it did not provide any improvement in performance over location details. Hence, we did not include postcode features in "Raw" feature set to ensure optimal performance in our comparison baseline while accurately capturing spatial dependence.

We consider a wide range of learning models to compare the performance. We use a number of widely-used regression models, including Lasso and Ridge regression as in [52], and Random Forest Regression, Elastic Net Regression, and Kernel-Ridge Regression. Furthermore, we also train a number of state-of-the-art models including Gradient Boosting, XGBoost [12], LightGBM [25], which have recently been shown to yield good performance in this problem. Finally, we also evaluate the ensembles of these models, including Averaging and Stacking with meta-model[53]. For Stacking, we used Gradient Boosting, XGBoost, and Kernel-Ridge Regression as first stage models, and Kernel-Ridge Regression as our meta model.

4.5.1 Result Summary

Table 3 summarizes our house prediction results, where we observe that for all models, the Raw+GSNE-based house prediction results consistently outperform that with the Raw features(including location information) only. We also observe that the less expressive variants of our embeddings, first-order (1^{st}) or second-order(2^{nd}) proximity, also outperform Raw in all cases.

Table 4: Comparison against spatial statistical approaches.

	MAE	RMSE
Raw GBoost	0.136	0.195
GBoost + Kriging	0.157	0.268
GBoost + GSNE	**0.125**	**0.181**
SVD + Kriging (two features)	0.524	0.898
GBoost + GSNE (two features)	0.168	0.231

From Table 3, we observe that with MAE, our best performing embeddings, i.e., Raw+GSNE (1^{st}+2^{nd}), outperform Raw with different downstream models by a notable margin ranging from 5.2% to 16.73%. Moreover in the best performing model, Gradient Boosting, Raw+GSNE outperforms Raw by 8.1%. For RMSE, Raw+GSNE(1^{st}+2^{nd}) outperforms different Raw based versions by a margin ranging from 3.7% to 12.91% . These results indicate the efficacy of GSNE in modeling neighbourhood preference, which significantly improves price prediction performance.

Performance against Spatial Statistical Approach: Since GSNE utilizes the neighborhood amenity features for prediction of the prices, it is interesting to see how the performance stacks up against spatial statistical approach. To experiment that, we first use Kriging [28] on the regression residual of Gradient Boosting Regression to compare it against GSNE. Besides, we also compared our performance against a recent work by [44], where the authors used Kriging on the output of SVD based rating model to better model the spatial characteristics. To utilize SVD, they built a house price matrix where rows are populated with unique latitude longitude combinations, and columns with unique non-location feature combinations. This strategy cannot handle high dimensional housing features due to extreme sparseness of the matrix (only 0.005% available values in our dataset). Thus to make a fair comparison, we "dumbed down" our model to utilize only two housing features (number of bedrooms and number of bathrooms) as done in [44]. These results are shown in Table 4.

Table 4 clearly shows that, spatial method like Kriging does not help in gaining performance advantage over raw ML approaches. In fact, the residuals of powerful ML models show no spatial dependency which causes worse performance with Kriging than standalone ML techniques. On the other hand, even in constrained features, GSNE performs significantly better than SVD-RK [44]. These results clearly show that GSNE extracts useful neighborhood information from nearby POIs which gives GSNE based approach a performance.

Confidence Interval: We also calculate the 95% confidence interval of MAE on both Raw and Raw+GSNE. For this comparison, we use Gradient Boosting Regression as our downstream model, since it gives the best result in Table 3. We bootstrap sampled the dataset 500 times, each time using 80% of the data to train, and 20% of the data to test the model. Results in the test set are then used to generate the density plot for confidence interval shown in Figure 4. From the result, we can observe that 95% of the time the MAE lies within 0.124 and 0.128 for our embeddings (Raw+GSNE), whereas for Raw features only it lies within 0.1338 and 0.139. This analysis statistically validates the efficacy of our proposed geo-spatial embedding in performance improvement.

4.5.2 Varying Price Partitions

To gain a better insight of the performance of our approach, we analyze the price prediction performance in different price quartiles of our dataset. We summarize these results in Table 5.

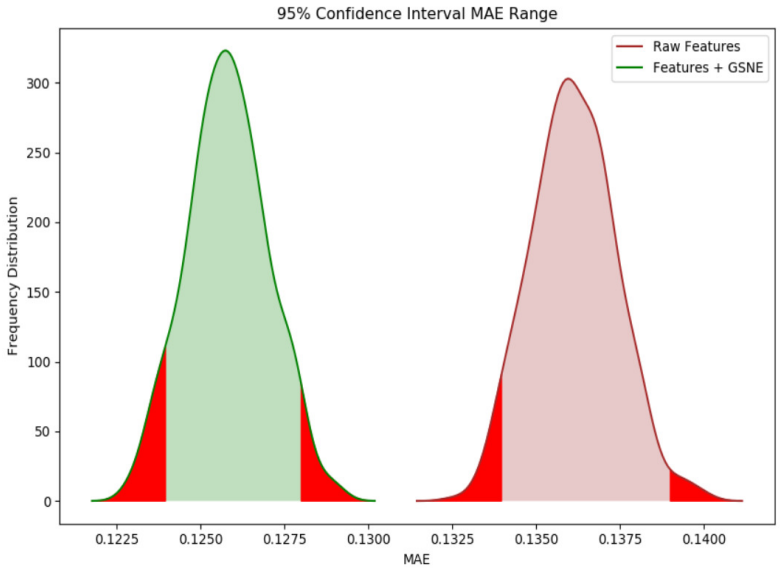


Fig. 4: Bootstrapped confidence interval comparison in Gradient Boost Regression. Here with *95%* confidence level, we can observe that the true MAE lies within *0.124 and 0.128* when our spatial embedding is used with housing features, whereas for raw features only, this range lies within *0.1338 and 0.139*.

From Table 5 we observe that, in every price quartile, Raw+GSNE outperforms Raw. We also separately tested the performance on outliers: houses with prices outside of the 3σ range of the distribution, where σ denotes the standard deviation of house price distribution. Outliers represent houses that are significantly more challenging to predict. Even on these outlier data, the features augmented with our embeddings outperform the Raw features only.

From Table 5, we can also observe an interesting aspect of our geo-spatial embeddings. While the generated embeddings always improve prediction performance over Raw features, the improvement is even more pronounced in the 1^{st} and 4^{th} quartile price partitions. Intuitively, houses in the highest price category usually enjoy modern amenities in the neighborhood, whereas these facilities are somewhat limited for the cheapest houses. This explains why neighborhood context might be more influential in the two ends of price partitions.

Table 5: Performance comparison in different price partitions using the Gradient Boosting regressor.

	MAE		RMSE	
Price Partition	Raw	Raw+GSNE	Raw	Raw+GSNE
1^{st} Quartile	0.119	**0.110**	0.163	**0.155**
2^{nd} Quartile	0.106	**0.102**	0.138	**0.133**
3^{rd} Quartile	0.122	**0.115**	0.157	**0.149**
4^{th} Quartile	0.173	**0.159**	0.228	**0.212**
Outside 3σ	0.518	**0.473**	0.678	**0.640**

4.5.3 Impact of POIs on House Price

To investigate how different POIs impact house price prediction, we take the three types of POIs, i.e. Region, School, and Train Station, separately and train our embedding model on each of these nodes independently. To achieve this, we essentially take each bipartite partition (house-region, house-train, or house-school) separately and train GSNE on these networks. The generated embeddings are then channeled through the Gradient Boosting Regressor model. These results are presented in Table 6.

As we can observe from the table, even the consideration of each POI node separately gives us substantial performance improvements over raw features. Table 6 also reveals an interesting insights on how neighbouring transportation facilities and educational institutes are influential in predicting house prices. Nevertheless, training GSNE with all types of POIs yields the best results both in terms of MAE and RMSE.

Table 6: Comparison of the effect of different POIs on house price with the Gradient Boosting Regressor.

	MAE	RMSE
Regions	0.131	0.188
Train Station	0.127	0.184
School	0.126	0.182
GSNE (All nodes)	**0.125**	**0.181**

4.6 Visualization

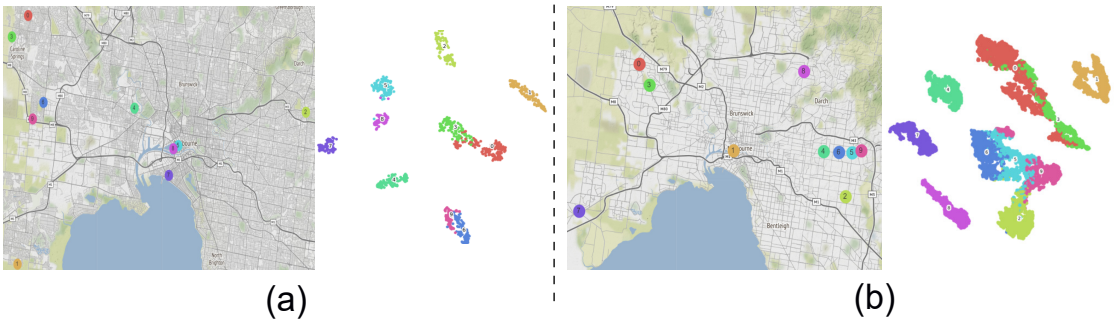


Fig. 5: 2D visualization of the generated geo-spatial embeddings. (a) Visualization of houses adjacent to 10 regions (shown in map). These regions are taken as labels of different colors for the houses. (b) Visualization of houses adjacent to 10 train stations (shown in map). In both cases, we chose 10 of the POIs that have the highest number of houses in the neighbourhood for a better visibility.

We visualize how GSNE embeds neighbourhood information for different POI category to qualitatively analyse embedding quality. In Figure 5 we show separate t-SNE[32] visualizations

for the categories *regions* and *train stations*. Since each of these categories contains a large number of POIs, we select the top 10 POIs of each category that have the highest number of houses in its neighbourhood. We use these POIs as the labels of each house in its neighbourhood. In other words, if a house i has an edge with a train station j, we use j as the label of the house. Consequently, different colours in the visualization represent different POIs in that category.

Figure 5(a) contains the t-SNE visualization for the ten *regions* as well as a map view of these regions, which shows their geographical locations. The t-SNE plots reveal that the separation is evident, such that houses in similar regions are closely clustered to each other. A deeper look at the plot reveals more insights about the efficacy of GSNE. In the map we can see that region labeled as **0** and **3** are located in the top left corner (marked with red and green respectively), and they are close to each other. Their close vicinity is reflected in the embedding, as can be seen in the t-SNE plot, where the red and green clusters are very close to each other. It can also be observed that these two clusters overlap each other. With further investigation of the dataset, we discovered that the overlapping houses have a number of identical features, including median age, median house rent, median weekly income, and median rent. In other words, even though these two regions are not immediately next to each other geographically, they share highly similar features in terms of overall regional information.

Another interesting case appears with region labeled as **5** and region **8**, coloured in cyan and purple respectively. In the map, we can see that these two regions are located close geographically. Yet they have very distinct feature sets. Houses in region **5** have almost twice the number of residents than region **8**, while only half of the median income of the residents of region **8**. Features such as median age, weekly house rent etc. are also distinct. These two cases give us an important insight about how GSNE embeds neighbourhood information in terms of the POI features. While it captures the neighbourhood information effectively, it also effectively captures the similarity of neighbourhoods in terms of their features. We see make similar observations about the other visible clusters. Regions labeled **1, 2, and 4** are very far from the other regions having distinctive features. They are also well separated from other clusters in the embedding space. Region **6 and 9** seem to be quite close geographically. Their feature sets are very similar with same median house rent, same median age, and similar median income. As can be observed, houses in these two regions are also closely clustered. From the analysis of this visualization, the efficacy of GSNE becomes quite evident.

In Figure 5(b) we show the t-SNE visualization for *train stations* and the corresponding map view. Here we can also observe good separations of clusters. From the map, it can be seen that the train stations labeled as **0** and **3** (marked red and green respectively) are located close to each other. These two stations also have identical feature sets. Their close vicinity can be observed in the t-SNE plot, showing the efficacy of our embedding model. Train station **1, 7, and 8** are far away from the other stations and each other, geographically, which is reflected in the t-SNE plot. On the other hand, station **2, 4, 5, 6, 9** are close geographically. In the embedding space, we see the clusters to be quite close for **2, 5, 6, 9**. However, the cluster for the station labeled as **4** is more separated from the above clusters. From the feature sets, we observed that the average time required to travel to other stations from station **4** is much lower compared to from stations **2, 5, 6, 9**, which is an indicator of better connectivity of station **4**. This essentially explains how the cluster of houses in the neighbourhood of station **4** achieves separation over the other nearby stations.

This visualization highlights insight about GSNE's capability to extract neighborhood characteristics based on the nearby POIs and their features. During training, as we try to project houses and their adjacent POIs in the embedding space, GSNE basically aggregates the POI information in the generated embedding. Consequently, in the t-SNE diagram we can see the embeddings that create meaningful separated clusters based on different POIs and their features.

4.7 Effect of Embedding Dimension

To check the effect of embedding dimension on overall performance, we investigate the performance on downstream GBoost model with different values of L. The results are shown in Table 7. From the results we can see even in low embedding dimension (8 or 16), GSNE performs reasonably well.

Table 7: Comparison of model performance in different embedding dimensions

Embedding Dimension	MAE	RMSE
8	0.128	0.184
16	0.127	0.182
32	**0.125**	**0.181**
40	0.131	0.187

4.8 Ablation Studies

From the performance comparison in Table 3, we can discern the effect of 1^{st}- and 2^{nd}-order proximities in the final result. We can observe that GSNE with either 1^{st}- or 2^{nd}-order proximity alone achieves noteworthy improvements over the performance of the raw features. Another notable fact is that, for regression models with higher expressive powers, we see comparable improvements for both 1^{st}- and 2^{nd}-order proximity version of GSNE. Nevertheless, in every case, the GSNE($1^{st} + 2^{nd}$) sees the highest improvement. This indicates that the **local** and **global** features extracted respectively by 1^{st}- and 2^{nd}-order proximity complement each other, and their combination results in best performance in any chosen regression model. We also observe that the geo-spatial embedding alone (without concatenating with the raw housing features) achieves 0.222 MAE and 0.306 RMSE, which also validates our claims on the importance of neighbourhood contextual information in housing preferences.

The potency of GSNE embeddings in house price prediction is also visible from the confidence interval (C.I.) plot of the Gradient Boosting Regressor in Figure 4. Here we bootstrap our dataset 500 times with 80% train and 20% test set, with which we train the Gradient Boost Regressor and examine its performance. From the plot, with 95% probability we can observe that GSNE embeddings along with raw features achieves an MAE between 0.124 and 0.128. These two tails are respectively 7.32% and 7.91% better than the raw features confidence interval results where this range lies between 0.1338 and 0.139. These observations clearly demonstrate that the performance improvements of our approach is consistent over the whole dataset.

From Table 6, we see that each of the different POI types gives different effects in house price prediction performance. Even though all of them improve performance when considered separately, we can see that the *School* POI type improves the performance by the highest margin. This may indicate that in deciding the purchase of a house, buyers may value educational institutions over other POIs. We also observe that *Train Station* POI type also gives a good house prediction performance boost as transport facilities in the neighborhood influence buyers' choices on a house. Yet, using all POI types in the embedding gives us the best performance, which indicates that all the POI types improve the performance of house price prediction.

Finally, Table 7 shows the performance of GSNE with different embedding dimensions. As the embedding dimension increases, the performance improves, where the best performance is

achieved at the dimension 32. We can see that even in the low dimensions, the performance impact is negligible, which shows the robustness of GSNE.

5 Conclusion and Future Works

In this paper, we have proposed a novel geo-spatial network embedding (GSNE) framework to accurately capture the geo-spatial neighborhood relationships between houses and surrounding POIs. The GSNE essentially learns low-dimensional Gaussian embeddings of nodes of a geo-spatial network. We have validated the efficacy of the GSNE in the house price prediction task, where our detailed experimental evaluation shows that GSNE features combined with the raw housing features can predict house prices with a higher accuracy (i.e., 8.1% lower MAE and 7.2% lower RMSE) than that of the best performing state of the art methods that only consider standalone house features. It is important to note that though we validate the proposed GSNE on the house price prediction problem in this paper, our proposed geo-spatial embedding can be highly effective in answering other real estate queries like recommending similar houses, which is of independent interest.

In future, we plan to explore how other complex house features such as textual description and images can be embedded in the multi-modal feature space to further enhance house price predictions. Another interesting area is to investigate potentiality of GSNE in other spatial domains such as tourist spot recommendation and alternative path suggestion as these applications can be greatly influenced by the nearby POIs and their attributes. Experimenting with other variants of network embedding in this framework can be another interesting area to look into. It will be also interesting to investigate how a geospatial model like GSNE learned in one city can be transferred to a geographically distant different city with minimal training data for fine-tuning.

6 Acknowledgements

We are grateful to Dr Zhifeng Bao, Associate Professor, RMIT University, Australia for sharing the Melbourne housing price dataset with us.

References

1. Stacked Regressions : Top 4% on LeaderBoard (2017). URL `https://kaggle.com/serigne/stacked-regressions-top-4-on-leaderboard`
2. What are SA1s, SA2s and SA3s? (2018). URL `https://communityinsightaustralia.org/what-are-sas/`
3. Basu, S., Thibodeau, T.G.: Analysis of spatial autocorrelation in house prices. The Journal of Real Estate Finance and Economics **17**(1), 61–85 (1998)
4. Bojchevski, A., Günnemann, S.: Deep gaussian embedding of attributed graphs: Unsupervised inductive learning via ranking. arXiv preprint arXiv:1707.03815 (2017)
5. Bourassa, S., Cantoni, E., Hoesli, M.: Predicting house prices with spatial dependence: a comparison of alternative methods. Journal of Real Estate Research **32**(2), 139–159 (2010)
6. Bourassa, S.C., Cantoni, E., Hoesli, M.: Spatial dependence, housing submarkets, and house price prediction. The Journal of Real Estate Finance and Economics **35**(2), 143–160 (2007)
7. Bourassa, S.C., Hoesli, M., Peng, V.S.: Do housing submarkets really matter? Journal of Housing Economics **12**(1), 12–28 (2003)
8. Cai, H., Zheng, V.W., Chang, K.C.C.: A comprehensive survey of graph embedding: Problems, techniques, and applications. IEEE Transactions on Knowledge and Data Engineering **30**(9), 1616–1637 (2018)
9. Case, B., Clapp, J., Dubin, R., Rodriguez, M.: Modeling spatial and temporal house price patterns: A comparison of four models. The Journal of Real Estate Finance and Economics **29**(2), 167–191 (2004)
10. Cavalcante, L., Bessa, R.J., Reis, M., Browell, J.: Lasso vector autoregression structures for very short-term wind power forecasting. Wind Energy **20**(4), 657–675 (2017)

11. Ceci, M., Corizzo, R., Malerba, D., Rashkovska, A.: Spatial autocorrelation and entropy for renewable energy forecasting. Data Mining and Knowledge Discovery **33**(3), 698–729 (2019)
12. Chen, T., Guestrin, C.: Xgboost: A scalable tree boosting system. In: Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining, pp. 785–794 (2016)
13. Chen, X., Wei, L., Xu, J.: House price prediction using lstm. arXiv preprint arXiv:1709.08432 (2017)
14. Clevert, D.A., Unterthiner, T., Hochreiter, S.: Fast and accurate deep network learning by exponential linear units (ELUs). arXiv preprint arXiv:1511.07289 (2015)
15. Corizzo, R., Ceci, M., Fanaee-T, H., Gama, J.: Multi-aspect renewable energy forecasting. Information Sciences **546**, 701–722 (2020)
16. Dubin, R.A.: Predicting house prices using multiple listings data. The Journal of Real Estate Finance and Economics **17**(1), 35–59 (1998)
17. Feng, Y., Jones, K.: Comparing multilevel modelling and artificial neural networks in house price prediction. In: 2015 2nd IEEE International Conference on Spatial Data Mining and Geographical Knowledge Services (ICSDM), pp. 108–114. IEEE (2015)
18. Fik, T.J., Ling, D.C., Mulligan, G.F.: Modeling spatial variation in housing prices: a variable interaction approach. Real Estate Economics **31**(4), 623–646 (2003)
19. Fletcher, M., Gallimore, P., Mangan, J.: The modelling of housing submarkets. Journal of Property Investment & Finance (2000)
20. Gao, G., Bao, Z., Cao, J., Qin, A.K., Sellis, T., Wu, Z., et al.: Location-centered house price prediction: A multi-task learning approach. arXiv preprint arXiv:1901.01774 (2019)
21. Gauvin, L., Panisson, A., Cattuto, C.: Detecting the community structure and activity patterns of temporal networks: a non-negative tensor factorization approach. PloS one **9**(1), e86028 (2014)
22. Grover, A., Leskovec, J.: node2vec: Scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 855–864 (2016)
23. Hettige, B., Li, Y.F., Wang, W., Buntine, W.: Gaussian embedding of large-scale attributed graphs. In: Australasian Database Conference, pp. 134–146. Springer (2020)
24. Jenkins, P., Farag, A., Wang, S., Li, Z.: Unsupervised representation learning of spatial data via multimodal embedding. In: Proceedings of the 28th ACM International Conference on Information and Knowledge Management, pp. 1993–2002 (2019)
25. Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., Liu, T.Y.: LightGBM: A highly efficient gradient boosting decision tree. In: Advances in neural information processing systems, pp. 3146–3154 (2017)
26. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)
27. Kipf, T.N., Welling, M.: Variational graph auto-encoders. arXiv preprint arXiv:1611.07308 (2016)
28. Krige, D.G.: A statistical approach to some basic mine valuation problems on the witwatersrand. Journal of the Southern African Institute of Mining and Metallurgy **52**(6), 119–139 (1951)
29. Król, A.: Application of hedonic methods in modelling real estate prices in poland. In: Data Science, Learning by Latent Structures, and Knowledge Discovery, pp. 501–511. Springer (2015)
30. Li, D.Y., Xu, W., Zhao, H., Chen, R.Q.: A svr based forecasting approach for real estate price prediction. In: 2009 International Conference on Machine Learning and Cybernetics, vol. 2, pp. 970–974. IEEE (2009)
31. Limsombunchai, V.: House price prediction: hedonic price model vs. artificial neural network. In: New Zealand agricultural and resource economics society conference, pp. 25–26 (2004)
32. Maaten, L.v.d., Hinton, G.: Visualizing data using t-sne. Journal of machine learning research **9**(Nov), 2579–2605 (2008)
33. Manganelli, B., De Mare, G., Nesticò, A.: Using genetic algorithms in the housing market analysis. In: International Conference on Computational Science and Its Applications, pp. 36–45. Springer (2015)
34. Montero, J.M., Mínguez, R., Fernández-Avilés, G.: Housing price prediction: parametric versus semi-parametric spatial hedonic models. Journal of Geographical Systems **20**(1), 27–55 (2018)
35. Morano, P., Tajani, F., Locurcio, M.: Multicriteria analysis and genetic algorithms for mass appraisals in the italian property market. International Journal of Housing Markets and Analysis (2018)
36. Ng, S.T., Skitmore, M., Wong, K.F.: Using genetic algorithms and linear regression analysis for private housing demand forecast. Building and Environment **43**(6), 1171–1184 (2008)
37. Ottensmann, J.R., Payton, S., Man, J.: Urban location and housing prices within a hedonic model. Journal of Regional Analysis and Policy **38**(1100-2016-89822) (2008)
38. Owusu-Ansah, A.: A review of hedonic pricing models in housing research. A Compendium of International Real Estate and Construction Issues **1**, 17–38 (2013)
39. Pace, R.K., Gilley, O.W.: Generalizing the ols and grid estimators. Real Estate Economics **26**(2), 331–347 (1998)
40. Perozzi, B., Al-Rfou, R., Skiena, S.: Deepwalk: Online learning of social representations. In: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 701–710 (2014)
41. Piao, Y., Chen, A., Shang, Z.: Housing price prediction based on cnn. In: 2019 9th International Conference on Information Science and Technology (ICIST), pp. 491–495. IEEE (2019)

42. Ravikumar, A.S.: Real estate price prediction using machine learning. Ph.D. thesis, Dublin, National College of Ireland (2017)
43. Rosen, S.: Hedonic prices and implicit markets: product differentiation in pure competition. Journal of political economy **82**(1), 34–55 (1974)
44. Sikder, A., Züfle, A.: Augmenting geostatistics with matrix factorization: A case study for house price estimation. ISPRS International Journal of Geo-Information **9**(5), 288 (2020)
45. Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., Mei, Q.: Line: Large-scale information network embedding. In: Proceedings of the 24th international conference on world wide web, pp. 1067–1077 (2015)
46. Thibodeau, T.G.: Marking single–family property values to market. Real Estate Economics **31**(1), 1–22 (2003)
47. Trojanek, R., et al.: Measuring dwelling price changes in poland with the application of the hedonic methods. Tech. rep., European Real Estate Society (ERES) (2013)
48. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y.: Graph attention networks. arXiv preprint arXiv:1710.10903 (2017)
49. Wang, D., Cui, P., Zhu, W.: Structural deep network embedding. In: Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 1225–1234 (2016)
50. Wang, T., Li, Y.Q., Zhao, S.F.: Application of svm based on rough set in real estate prices prediction. In: 2008 4th International Conference on Wireless Communications, Networking and Mobile Computing, pp. 1–4. IEEE (2008)
51. Wang, X., Wen, J., Zhang, Y., Wang, Y.: Real estate price forecasting based on svm optimized by pso. Optik **125**(3), 1439–1443 (2014)
52. Xin, S.J., Khalid, K.: Modelling house price using ridge regression and lasso regression. International Journal of Engineering & Technology **7**(4.30), 498–501 (2018)
53. Xiong, S., Sun, Q., Zhou, A.: Improve the house price prediction accuracy with a stacked generalization ensemble model. In: International Conference on Internet of Vehicles, pp. 382–389. Springer (2019)
54. Yayar, R., Demir, D.: Hedonic estimation of housing market prices in turkey. Erciyes Univ. J. Fac. Econ. Adm. Sci pp. 67–82 (2014)
55. Zhao, Y., Chetty, G., Tran, D.: Deep learning with xgboost for real estate appraisal. In: 2019 IEEE Symposium Series on Computational Intelligence (SSCI), pp. 1396–1401. IEEE (2019)
56. Zhu, D., Cui, P., Wang, D., Zhu, W.: Deep variational network embedding in wasserstein space. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 2827–2836 (2018)