HAGEN: Homophily-Aware Graph Convolutional Recurrent Network for Crime Forecasting

Chenyu Wang[1*], Zongyu Lin[1*], Xiaochen Yang[1], Jiao Sun[1], Mingxuan Yue[1], Cyrus Shahabi[1†]

[1] University of Southern California, Los Angeles, CA, USA
{cy-wang18, lin-zy17}@mails.tsinghua.edu.cn, {xiaochey, jiaosun, mingxuay, shahabi}@usc.edu

Abstract

The goal of the crime forecasting problem is to predict different types of crimes for each geographical region (like a neighborhood or censor tract) in the near future. Since nearby regions usually have similar socioeconomic characteristics which indicate similar crime patterns, recent state-of-the-art solutions constructed a distance-based region graph and utilized Graph Neural Network (GNN) techniques for crime forecasting, because the GNN techniques could effectively exploit the latent relationships between neighboring region nodes in the graph if the edges reveal high dependency or correlation. However, this distance-based pre-defined graph cannot fully capture crime correlation between regions that are far from each other but share similar crime patterns. Hence, to make a more accurate crime prediction, the main challenge is to learn a better graph that reveals the dependencies between regions in crime occurrences and meanwhile captures the temporal patterns from historical crime records. To address these challenges, we propose an end-to-end graph convolutional recurrent network called HAGEN with several novel designs for crime prediction. Specifically, our framework could jointly capture the crime correlation between regions and the temporal crime dynamics by combining an adaptive region graph learning module with the Diffusion Convolution Gated Recurrent Unit (DCGRU). Based on the homophily assumption of GNN (i.e., graph convolution works better where neighboring nodes share the same label), we propose a homophily-aware constraint to regularize the optimization of the region graph so that neighboring region nodes on the learned graph share similar crime patterns, thus fitting the mechanism of diffusion convolution. Empirical experiments and comprehensive analysis on two real-world datasets showcase the effectiveness of HAGEN.

1 Introduction

Accurate crime forecasting can better guide the police deployment and allocation of infrastructure, resulting in great benefits for urban safety. Previous studies have designed several spatiotemporal deep learning frameworks for crime forecasting, including MiST (Huang et al. 2019) and DeepCrime (Huang et al. 2018). However, they assume a grid-based partitioning of the underlying geographic region to utilize CNN-based models which ignore the geographical extents of neighborhoods. Based on the intuition that nearby regions (i.e., geographical neighborhoods which collect criminal records within a period of time) have similar crime patterns, Sun et al. (2020) leveraged a distance-based region graph to explore the connection between spatial and temporal crime patterns, and utilized a graph neural network (GNN) technique to model the spatial dependencies between regions. In this setting, the crime forecasting problem is equivalent to learning a mapping function to predict the future graph signals (i.e., crime occurrences) given the historical graph signals (i.e., historical crime records) over a distance-based region graph.

However, the geographical distance does not always reveal the real correlation of crime patterns because other shared socioeconomic factors between (possibly far apart) regions may result in similar crime patterns in the regions. As illustrated in Figure 1, we observe that even though the distance between regions a and b is much farther than that of regions a and c, region a shares similar crime patterns with c but totally different from b. One may build another graph on which the edges are defined by the similarity of POIs between regions (Xu et al. 2020), nevertheless, the mismatching phenomenon still remains. In fact, the local government deploys heavy policing for the university in region a, thus a has a different crime pattern from c despite both of them being urban regions. Obviously, one can construct customized graphs for each and every case, but it is hard to predefine a graph that adapts to all cases. Towards this end, we propose to learn an adaptive graph that could dynamically capture the crime-specific correlations among regions from real-world data. In particular, we utilize a graph learning layer that learns the graph structure adaptively and jointly with the training process of crime forecasting. Nevertheless, the adaptive learning of such graph structure is non-trivial and usually requires some heuristics to regularize the graph to avoid over-fitting. For example, in the traffic forecasting domain, Graph WaveNet (Wu et al. 2019) uses the node similarity to construct the graph, and MTGNN (Wu et al. 2020) is the state-of-the-art spatiotemporal graph neural network that integrates a graph learning module to extract the uni-

*Both authors contributed equally to this work. The work was conducted while both authors were interns at USC's IMSC.

†This is the corresponding author.

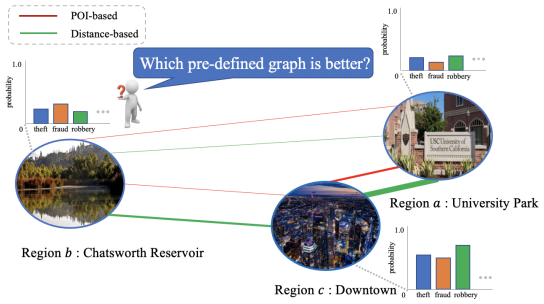


Figure 1: The width of lines represents the geographical distance or POI based similarity (the wider, the closer or more similar). As elaborated in Section 1, there are mismatches between crime patterns and either geographical distances or POI similarity (e.g. University Park and Downtown are close in both distance and POI similarity, while they have different crime patterns; on the other hand, although Chatworth Reservoir and University Park are far away both geographically and in POI similarity, they have similar crime patterns).

directed relations among sensors while preserving the sparsity of the graph. Although previous studies have considered some general graph properties in constructing the adaptive graph structure, they failed to incorporate the graph properties that are highly effective to the performance of GNN models. On the other hand, recent studies (Zhu et al. 2020; Chien et al. 2021) observed that the homophily of a graph, the property that the neighboring nodes share the same labels, significantly affects the performance of GNN. Consider our crime forecasting task as an example, the nodes in our graph are regions and the edges between nodes should indicate a high correlation of crime patterns. If the region nodes on the learned graph share totally different crime patterns with their neighboring nodes, GNN will aggregate and propagate the information of these dissimilar neighboring nodes resulting in an inaccurate inference of crime rates. Inspired by these studies, our goal is to preserve the homophily of crime patterns among neighboring nodes of our graph.

In this work, we introduce HAGEN, a Homophily-Aware Graph convolutional rEcurrent Network with several novel designs for crime forecasting. Specifically, we model intra-region and inter-region dependencies of crime patterns by introducing an adaptive graph learning layer with region and crime embedding layer. By observing that the homophily ratio is highly correlated with model performance, we propose a homophily-aware constraint to boost the ability of region graph learning. Subsequently, we utilize a weighted graph diffusion layer to simulate crime diffusion and capture the crime-specific dependencies between regions. Finally, we integrate the Gated Recurrent Network in the graph diffusion layer to capture temporal crime dynamics. Our empirical studies on two real-world crime datasets show that HAGEN outperformed both state-of-the-art crime forecasting methods (CrimeForecaster (Sun et al. 2020) and MiST (Huang et al. 2019)) and generic spatiotemporal GNN methods (MTGNN (Wu et al. 2020) and GraphWaveNet (Wu et al. 2019)). Case studies revealed some insights for both the research community and urban planners. *

2 Problem Definition

In this paper, we focus on the task of crime forecasting given previous crime records. Our setting is the same as CrimeForecaster's (Sun et al. 2020).

Definition 1 ***Crime Record.*** *For the inputs of forecasting model, we consider the crimes occurring in the past sequence of non-overlapping and consequent time slots $T = (t_1, \ldots, t_K)$, where K is the time sequence length. For each region r_i, we use $\mathcal{Y}_i = (y^1_{i,1}, \ldots, y^k_{i,l} \ldots, y^K_{i,C}) \in \mathbb{R}^{C \times K}$ to denote all C types of crimes that occurred during the past K slots' observations. Following the general settings of previous crime forecasting studies (Huang et al. 2018, 2019; Sun et al. 2020), we set each element $y^k_{i,l}$ to 1 if crime type l happens at region i in time slot t, and 0 otherwise. Given the sparsity of crime intensity data in public domain, i.e., the number of a specific crime type at each fine grained unit (by time and space) are mostly 1 or 0, we believe this classification model is more appropriate than a regression model.*

To utilize the advanced graph neural network on crime forecasting, we define a region graph with a weight matrix describing the relationships between region nodes. Also, we introduce Homophily Ratio (Zhu et al. 2020) of a graph with node labels which is related to our model design.

Definition 2 ***Region Graph.*** *A region graph is formulated as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{A}_r)$, where $\mathcal{V}$ is a set of region nodes with $|\mathcal{V}| = N$ where N is the number of region nodes in the graph, and $\mathcal{E}$ is a set of directional edges between region nodes, $\mathcal{A}_r$ is the weight matrix in Def 3.*

Definition 3 ***Weight Matrix.*** *Weight matrix is a representation of a directed graph, denoted as $\mathcal{A}_r \in \mathbb{R}^{N \times N}$ with $\mathcal{A}_r(i,j) = w > 0$ if $(v_i, v_j) \in \mathcal{E}$ and $\mathcal{A}_r(i,j) = 0$ if $(v_i, v_j) \notin \mathcal{E}$. The $\mathcal{A}_r(i,j)$ reflects the strength of influence from region i to region j.*

Definition 4 ***Homophily Ratio.*** *For a given graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, A)$ with node label $\mathcal{Y}$, Zhu et al. (2020) defines the graph's homophily ratio $\mathcal{H}(\mathcal{G}, \mathcal{Y})$ to represent the probability that neighboring nodes share the same label as follows:*

$$\mathcal{H}(\mathcal{G}, \mathcal{Y}) = \frac{1}{|\mathcal{V}|} \sum_{v \in \mathcal{V}} \frac{|\{u : u \in \mathcal{N}_v \wedge y_u = y_v\}|}{|\mathcal{N}_v|}$$

where $\mathcal{N}_v$ denotes the neighboring nodes of region node v and y_v is the label of node v.

HAGEN Goal. Given the historical crime records across all regions from time slot t_1 to t_K: $\{\mathcal{Y}^1, \mathcal{Y}^2, \ldots, \mathcal{Y}^k, \ldots, \mathcal{Y}^K\}$, and the predefined region graph $\hat{\mathcal{G}}(\mathcal{V}, \hat{\mathcal{E}}, \hat{\mathcal{A}}_r)$, we aim to learn a function $p(\cdot)$ which can learn the adaptive region graph structure $\mathcal{G}(\mathcal{V}, \mathcal{E}, \mathcal{A}_r)$ and finally forecast the crime occurrences $\mathcal{Y}^{K+1}$ of all crime categories for all regions at time t_{K+1}, which can be formulated as follow:

$$\{\mathcal{Y}^1, \mathcal{Y}^2, \ldots, \mathcal{Y}^k, \ldots, \mathcal{Y}^K, \hat{\mathcal{G}}\} \xrightarrow{p(\cdot)} \{\mathcal{Y}^{K+1}, \mathcal{G}\},$$

where $\mathcal{Y}^k \in \mathbb{R}^{N \times C}$ denotes the crime records across all N regions and C crime categories in time slot t_k.

*The code and data will be released in the final version.

3 HAGEN Model

3.1 Framework Overview

We first explain the general framework of our model. As illustrated in Figure 2, HAGEN at its highest level consists of the homophily-aware graph learning layer, the diffusion graph convolution module, GRU-based temporal module, and the MLP-based decoder module. To discover hidden associations among region nodes, a graph learning layer with homophily-aware constraint learns a weight matrix, which is used as an input to the diffusion convolution modules. Diffusion convolution modules are interwoven with GRU networks to capture temporal dependencies. To obtain final predictions, the MLP-based decoder module projects the hidden features to the desired output dimension. Each core component of our model is discussed in turn as follows.

3.2 Region Graph Learning

A well-defined graph structure that is suitable for crime forecasting is essential for graph-based methods. As we illustrated in Figure 1, a pre-defined graph cannot fully capture the real connectivity in terms of crime patterns, thus affecting the final performance of crime forecasting. Therefore, we aim to learn the graph adaptively instead of using a fixed graph to reflect the crime patterns shared by regions.

Besides, with crime forecasting, we often assume that the change of a region's security condition indicates the change of another region's security condition (Morenoff and Sampson 1997), which can be dubbed as crime flow. Hence, we suppose the connections between region nodes in our crime-specific graph are uni-directional (i.e., if $\mathcal{A}_r(i,j) > 0$, $\mathcal{A}_r(j,i)$ must be zero), which leads to better model performance in practice. Rather than refining the graph structure directly during the training process, we achieve the learning of the graph by updating the node embeddings adaptively and then change the graph structure accordingly. Similar to (Wu et al. 2020), we design the graph learning layer based on two region embeddings: source region embedding $\mathbf{E_s} \in \mathbb{R}^{N\times D}$ and target region embedding $\mathbf{E_t} \in \mathbb{R}^{N\times D}$, where N is the number of regions and D is the dimension of the embedding space. We leverage pairwise region node similarity to compute the uni-directional adaptive weight matrix $\mathcal{A}_r$ as follows:

$$\mathbf{Z_s} = \tanh(\alpha \mathbf{E_s}\mathbf{\Theta_1}) \tag{1}$$

$$\mathbf{Z_t} = \tanh(\alpha \mathbf{E_t}\mathbf{\Theta_2}) \tag{2}$$

$$\mathcal{A}_r = \mathrm{ReLU}(\tanh(\alpha(\mathbf{Z_s}\mathbf{Z_t}^T - \mathbf{Z_t}\mathbf{Z_s}^T))) \tag{3}$$

where α is the hyper-parameter to control the saturation rate of the hyperbolic tangent function and $\mathbf{\Theta}_1$ and $\mathbf{\Theta}_2$ denote linear transformation weights. The ReLU activation function leads to the asymmetric property of matrix $\mathcal{A}_r$. To ensure graph sparsity, we connect each node with the top k nodes with the largest similarity. The weights between disconnected nodes are set to be zero.

To incorporate the geographical proximity between regions and have meaningful initial graph for faster convergence, we initialize both the source and target region embeddings with the pre-trained embeddings on graphs defined by geospatial and POI proximity. For simplicity, we utilize Node2Vec (Grover and Leskovec 2016) (a classical graph embedding method) to pre-train the region embedding $\mathbf{E_{pre}}$ based on the distance-based graph and the POI similarity-based graph, respectively, and concatenate them together.

3.3 Homophily-aware Constraint

The remaining question is that what factors we should consider to learn a good region graph in addition to preserving node similarity and controlling sparsity (Wu et al. 2020, 2019). Following the basic homophily assumption of graph neural network (Zhu et al. 2020), we aim at learning a region graph where neighboring region nodes share similar crime patterns. Therefore, we propose a homophily constraint to the learning process for explicit heterophily reduction (i.e., reduce the probability that neighboring region nodes share totally different crime patterns).

Since the original definition of homophily ratio (see Def. 4) is defined on static graphs for node classification under the semi-supervised setting, we need to extend it to fit our case. For our crime forecasting problem, historical crime observations in region r_i and time slot t_k across different crime categories $y^k_{i,1}, \ldots, y^k_{i,C}$ have the additional temporal dimension k and the edges are defined by the weight matrix $\mathcal{A}_r$ with element $\mathcal{A}_r(u,v) \in [0,1]$ representing the edge weight between region u and region v. Therefore, given time slot t_k and crime category l, we extend the definition of homophily ratio $\mathcal{H}(\mathcal{A}_r, \mathcal{Y}^k_l, l)$ as:

$$\mathcal{H}(\mathcal{A}_r, \mathcal{Y}^k_l, l) = \frac{1}{|\mathcal{V}|}\sum_{v\in\mathcal{V}} \frac{\sum_{u\in N(v), y^k_{u,l}=y^k_{v,l}} \mathcal{A}_r(u,v)}{\sum_{u\in N(v)} \mathcal{A}_r(u,v)} \tag{4}$$

The extended homophily ratio is the sum of edge weight where connected region nodes share the same crime label divided by the sum of the edge weight considering whole neighbor nodes. Intuitively, the extended homophily ratio measures the probability that neighbor region nodes share similar crime patterns, thus optimizing the homophily ratio to 1 (the maximum of homophily ratio) of our graph would boost the effectiveness of information propagation of graph convolution in Section 3.4 and finally contribute to crime forecasting. Therefore, with definition of the extended homophily ratio, we formally define loss $\mathcal{L}_{homo}$ to regularize homophily of the learnt graph from time t_1 to t_K as follows:

$$\mathcal{L}_{homo} = \Sigma^K_{k=1}\Sigma^C_{l=1}[\mathcal{H}(\mathcal{A}_r, \mathcal{Y}^k_l, l) - 1]^2 \tag{5}$$

where C denotes the number of crime categories and $\mathcal{Y}^k_l$ is the records of crime category l at time slot t_k.

3.4 Graph Diffusion Convolution

In the graph diffusion convolution module, we propose a direction-aware diffusion convolution layer to simulate real-life crime diffusion and capture crime dependencies between regions. We concern the propagation of crime patterns between regions as a diffusion process from one region's first order neighbors to its M-th order neighbors or in the reversed way. To learn such diffusion pattern, similar to (Li et al. 2018), we characterize graph diffusion on $\mathcal{G}$ with the

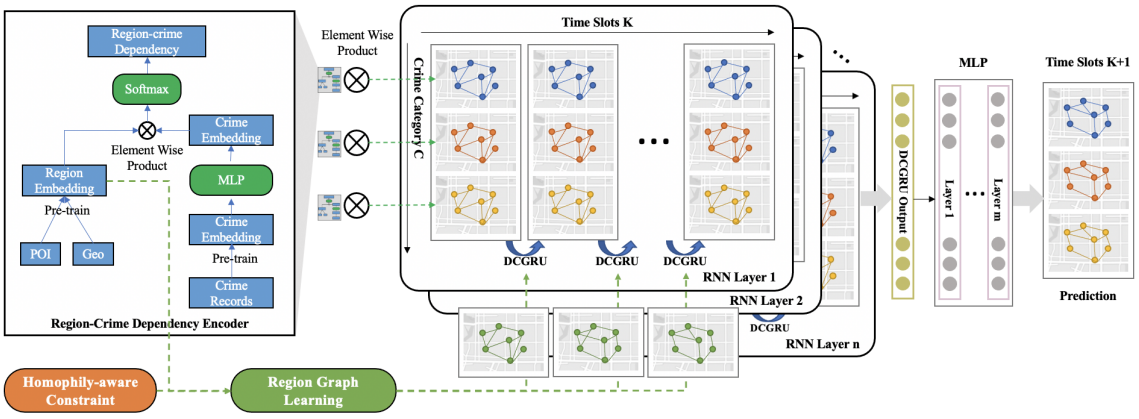


Figure 2: HAGEN Framework. HAGEN consists of homophily-aware graph learning layer, the diffusion graph convolution module, GRU-based temporal module, and the MLP-based decoder module. Details are explained in Section 3.

m-th order transition matrices $\mathbf{S}_m^O, \mathbf{S}_m^I$ on out-degree and in-degree basis respectively:

$$\mathbf{S}_m^O = (\mathbf{D}_O^{-1}\mathcal{A}_r)^m \tag{6}$$

$$\mathbf{S}_m^I = (\mathbf{D}_I^{-1}\mathcal{A}_r^\top)^m \tag{7}$$

where $\mathbf{D}_O$ and $\mathbf{D}_I$ are diagonal matrices of nodes out-degree and in-degree respectively. Different from (Li et al. 2018), the weight matrix $\mathcal{A}_r$ is calculated with unidirectional constraints. Thus the diffusion matrices $\mathbf{S}_m^O, \mathbf{S}_m^I$ will be sparse and also follow the unidirectional constraint during the m-step diffusion.

Given the above transition matrices, we then define a direction-aware diffusion convolution function that transform the input $\mathbf{X} \in \mathbb{R}^{N\times H}$ from all nodes into (hidden) outputs via the diffusion process. Here if $\mathbf{X}$ is the input, H equals to number of crime categories C, otherwise it is the dimension of the hidden states. Specifically, the direction-aware diffusion convolution function $f_*(\mathbf{X}; \mathcal{G}, \mathbf{\Theta}, \mathbf{D_W})$ aggregates the input $\mathbf{X}$ on graph $\mathcal{G}(\mathcal{V}, \mathcal{E}, \mathcal{A}_r)$ with parameters $\mathbf{\Theta}$ and $\mathbf{D_W}$ in the following way:

$$f_*(\mathbf{X}; \mathcal{G}, \mathbf{\Theta}, \mathbf{D_W}) = \sum_{m=0}^{M} (\mathbf{S}_m^O \mathbf{X} \mathbf{\Theta}_{:,:,\mathbf{m,1}} + \mathbf{D_W} \mathbf{S}_m^I \mathbf{X} \mathbf{\Theta}_{:,:,\mathbf{m,2}}) \tag{8}$$

where M is the total diffusion step, $\mathbf{\Theta} \in \mathbb{R}^{H\times H'\times(M+1)\times 2}$ (H' is the output dimension) is the filter parameter. $\mathbf{D_W}$ is a diagonal matrix designed for measuring the direction preference of each node. Specifically, the i_{th} diagonal value of $\mathbf{D_W}$ denotes the the preference of region i on in-going diffusion. A high value means the region prefers in-going diffusion to out-going diffusion, i.e., the region is more likely to be affected by other regions than to affect other regions. Via this direction-aware diffusion convolution function, each node (region) would learn its own way to perform the aggregation of crime patterns from neighboring nodes and produce meaningful (hidden) outputs for further computation.

3.5 Temporal Module

For temporal crime dynamics, we use RNN with DCGRU (Diffusion Convolutional Gated Recurrent) units (Li et al. 2018) in HAGEN to capture the complex crime connectivity across historical time slots (from t_1 to t_K). In DCGRU, the original linear transformation in GRU is replaced by the direction-aware diffusion convolution in Equation 8, which can incorporate global information from the whole graph and enable the learning process of a node to be based on not only its previous state but also its neighbors' previous state with similar crime patterns. We formulate the updating functions for hidden state h^t at t-th time step of input signal sequence in our DCGRU-revised RNN encoder as follows:

$$\begin{aligned} r^t &= \sigma(f_*([x^t, h^{t-1}]; \mathcal{G}, \mathbf{\Theta_r}, \mathbf{D_W}) + \mathbf{b_r}) \\ u^t &= \sigma(f_*([x^t, h^{t-1}]; \mathcal{G}, \mathbf{\Theta_u}, \mathbf{D_W}) + \mathbf{b_u}) \\ c^t &= tanh(f_*([x^t, r^t \odot h^{t-1}]; \mathcal{G}, \mathbf{\Theta_c}, \mathbf{D_W}) + \mathbf{b_c}) \\ h^t &= u^t \odot h^t + (1 - u^t) \odot c^t \end{aligned} \tag{9}$$

where x^t, h^t are the input and output of the DCGRU cell at time t, and r^t, u^t, c^t are the reset gate, update gate and cell state respectively. The parameters $\mathbf{\Theta_r}, \mathbf{\Theta_u}$ and $\mathbf{\Theta_c}$ are filter parameters of diffusion convolution and $\mathbf{b_r}, \mathbf{b_u}$ and $\mathbf{b_c}$ are the bias terms. All gates share the same direction weight $\mathbf{D_W}$, indicating the proportional intensity of reverse diffusion is consistent for the same region. The parameter σ denotes the sigmoid function and $\odot$ denotes the Hadamard Product. From the encoder-decoder perspective, as shown in Figure 2, HAGEN's encoder consists of multiple stacked layers of RNNs with DCGRU units, which capture temporal transitions across regions and time. The encoder's outputs, i.e., the final hidden states, are delivered to the decoder for future crime forecasting.

3.6 Region-crime Dependency Encoder

As depicted in the left part of Figure 2, in order to capture the underlying dependencies between regions and crime categories, we introduce the crime embedding $\mathbf{E}_c \in \mathbb{R}^{C\times D}$ for

all C crime categories and compute the inter region-crime dependency matrix $\mathbf{W_{inter}} \in \mathbb{R}^{N \times C}$ by calculating an element-wise product between region embedding and crime embedding with a transition matrix. Each element of the inter dependency matrix $\mathbf{W_{inter}}(i, l)$ represents the dependency between region i and crime category l. After normalizing the dependency weight via a softmax function, we perform an element-wise product between dependency weight matrix $\mathbf{W_{inter}}$ and the input crime records $\mathcal{Y}^k \in \mathbb{R}^{N \times C}$ across all regions and categories in k-th time slot to generate a weighted input $\mathbf{W_{inter}} \odot \mathcal{Y}^k \in \mathbb{R}^{N \times C}$ which is then fed into the encoder (i.e., RNN with DCGRU units) of HAGEN.

To incorporate the prior knowledge of crime patterns, we flatten the training records across all regions and time slots for each crime category and use PCA for dimension reduction to initialize the crime embedding $\mathbf{E}_c \in \mathbb{R}^{C \times D}$.

3.7 Crime Forecasting and Model Inference

In general, we employ the Encoder-Decoder architecture (Cho et al. 2014) for the crime forecasting. After encoding the temporal crime dynamics by the encoder, HAGEN utilizes a Multilayer Perceptron (MLP) based decoder to map the encoded hidden states to the outputs for crime forecasting in a non-linear way. We formulate the decoder (i.e., MLP with diffusion convolutional layers) as follows.

$$\begin{aligned} \psi_1 &= \text{ReLU}(f_*(h; \mathcal{G}, \mathbf{W}_1, \mathbf{D_W}) + b_1) \\ &... \\ \psi_P &= f_*(\psi_{P-1}; \mathcal{G}, \mathbf{W}_P, \mathbf{D_W}) + b_P) \\ y &= \sigma(f_*(\psi_P; \mathcal{G}, \mathbf{W}', \mathbf{D_W}) + b') \end{aligned} \tag{10}$$

where h is the input of the decoder and is retrieved from the output of temporal module, i.e., h^t. The output y is a matrix denoting the crime occurrence probabilities of all regions for all categories at the prediction time t_{K+1}. During the training process, we use binary cross entropy as the major learning objective:

$$\mathcal{L}_{crime} = - \sum_{\substack{i \in \{1,...,R\}, \\ l \in \{1,...,C\}}} y_{i,l} \log \hat{y}_{i,l} + (1 - y_{i,l}) \log(1 - \hat{y}_{i,l}), \tag{11}$$

where $y_{i,l}$ and $\hat{y}_{i,l}$ represent the empirical and estimated probability of the l-th crime category happening at region i at the time t_{K+1}, respectively.

As mentioned in Section 3.3, we propose a homphily-aware constraint loss during the graph learning process to conform to the homophily assumption of GNN. A natural approach is to optimize linear combination of the corresponding loss functions, which is formulated as follows:

$$\mathcal{L}_{total} = \mathcal{L}_{crime} + \lambda \mathcal{L}_{homo}, \tag{12}$$

where λ is the trade-off parameter. We minimize the joint loss function by using the Adam optimizer (Kingma and Ba 2014) with learning rate decay strategy to learn the parameters of HAGEN.

4 Experiments

We conduct our experiments on two real-world datasets with the following objectives. First, we compare HAGEN's performance to those of state-of-the-art spatial and temporal graph neural networks and crime forecasting models. We also explore HAGEN's performance for different crime categories, and how HAGEN's various components affect performance. Then, we focus on model explainability to gain meaningful insights from the learned region graph in Section 4.3.

4.1 Data and Experimental Setting

Data Description and Training Configuration We evaluated HAGEN on two real-world benchmarks in Chicago and Los Angeles by CrimeForecaster (Sun et al. 2020). In our experiments, we use the same "train-validation-test" setting as the previous work (Sun et al. 2020; Huang et al. 2019). We chronologically split the dataset as 6.5 months for training, 0.5 months for the validation, and 1 month for testing. For the vital hyperparameters in HAGEN, we use two stacked layers of RNNs. Within each RNN layer, we set 64 as the size of the hidden dimension. Moreover, we set the subgraph size of the sparsity operation as 50 and the saturation rate as 3. For the learning objective, we fix the trade-off parameter λ as 0.01, similar to the common practice of other regularizers.

Evaluation Metric Since the crime forecasting problem can be viewed as a multi-label classification problem, we utilize Micro-F1 (Grover and Leskovec 2016) and Macro-F1 (Lin et al. 2020) as general metrics to evaluate prediction performance across all crime categories, similar to CrimeForecaster (Sun et al. 2020) and MiST (Huang et al. 2019).

Baseline We choose three categories of baselines.

Traditional Methods. We use three types of traditional methods, including time series forecasting model ARIMA (Contreras et al. 2003), classical machine learning methods Epsilon-Support Vector Regression (SVR) (Chang and Lin 2011), Decision Tree (Safavian and Landgrebe 1991) and Random Forest (Verikas, Gelzinis, and Bacauskiene 2011), and traditional neural networks Multilayer Perceptron classifier (MLP) (Covington, Adams, and Sargin 2016), Long Short-term Memory (LSTM) (Gers, Schmidhuber, and Cummins 1999) and Gated Recurrent Unit (GRU) (Chung et al. 2014).

Spatial-temporal graph neural network (i.e., MTGNN and Graph WaveNet). Among the spatial-temporal graph neural network models in Section 5.2, we select two methods designed for multivariate time-series forecasting which are generally used as baselines in previous works for comparison. Graph WaveNet (GW) (Wu et al. 2019) is a spatial-temporal graph neural network that combines diffusion convolutions with 1D dilated convolutions. MTGNN (Wu et al. 2020) is the state-of-the-art spatial-temporal graph neural network in previous works, which integrates the adaptive graph structure, mix-hop graph convolution layers, and dilated temporal convolution layers.

Spatial-temporal learning models for crime forecasting (i.e., MiST and CrimeForecaster). MiST (Huang et al. 2019) learns both the inter-region temporal and spatial correlations. To the best of our knowledge, CrimeForecaster (CF) (Sun et al. 2020) is the most recent crime forecasting model, which is an end-to-end framework to model the dependencies between regions by geographical neighborhoods

instead of grid partitions and captures both spatial and temporal dependencies using diffusion convolution layer and Gated Recurrent Network.

4.2 Performance Evaluation

Overall Performance Table 1 shows the crime forecasting accuracy in both Chicago and Los Angeles dataset. We evaluate the performance of crime forecasting in terms of Macro-F1 and Micro-F1 for all methods. We summarize the following key observations from Table 1:

Advantage of graph-based model. For the three types of baseline models stated in Section 4.1, graph-based methods generally outperform non-graph-based ones. We share the same conclusion with CrimeForcaster that graphs can better capture spatial relation between regions, compared with both grid-based models (i.e., MiST) and traditional methods.

Advantage of graph learning with the homophily-aware constraint. HAGEN consistently outperforms the state-of-the-art CrimeForecaster in 5 testing months on both datasets in Micro-F1 and Macro-F1, which showcases the effectiveness of our adaptively-learned graph. In addition, HAGEN outperforms other models with adaptive graph but without homophily-aware constraint (i.e., MTGNN and Graph WaveNet), verifying the importance of incorporating the homophily-aware constraint when learning the region graph.

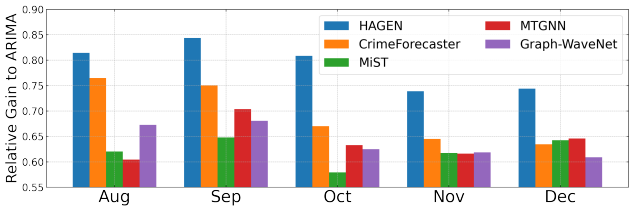


Figure 3: Relative Macro-F1 gain of models over ARIMA on LA dataset.

HAGEN performs significantly better than all other learning models. In particular, the most notable advancement occurs in crime forecasting of Los Angeles in October, November, and December. Note that the improvement of CrimeForecaster as compared to the previous state-of-the-art, MiST, is less than that of HAGEN over CrimeForecaster. HAGEN reaches a 5.50% relative improvement over the state-of-the-art (i.e., CrimeForecaster) in terms of Micro-F1 and an 8.25% relative improvement in terms of Macro-F1 in October. We take ARIMA as the base model and display the relative gain of competitive models over it in Figure 3.

Performance Comparison Across Crime Categories We also evaluate HAGEN's performance across different crime categories with those of a selected competitive baselines (i.e., CrimeForecaster, MTGNN, and Graph WaveNet) on the Chicago dataset. As shown in Figure 4, HAGEN achieves almost consistent performance gain in Micro-F1 score across various crime categories compared to the competitors, showing the effectiveness of our model.

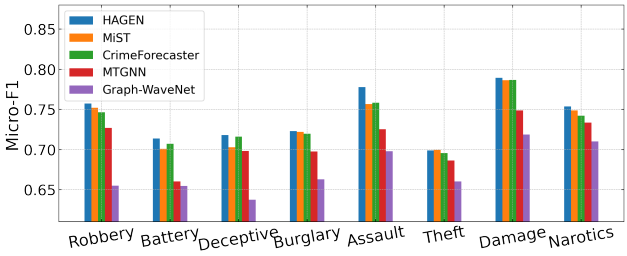
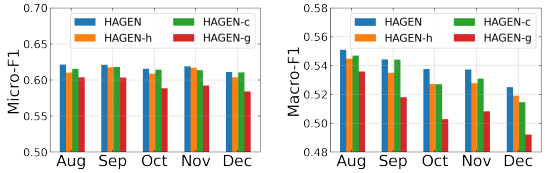


Figure 4: Micro-F1 for individual crime categories on Chicago dataset.

Ablation Study We evaluate how each key component contributes to our framework. We consider three degrade variants: HAGEN-h, HAGEN-c, HAGEN-g, which removes the homophily-aware constraint, region-crime dependency and graph learning layer from HAGEN respectively. We compare complete HAGEN with variants in Figure 5(a) and 5(b). We observe that taking out each component in HAGEN will lead to a performance drop, which indicates the importance of each component and justifies our model design. Specifically, HAGEN-g incurs the worst performance, suggesting graph learning layer to be the most impactful component.



(a) Micro-F1 (b) Macro-F1

Figure 5: Evaluation on the ablated variants of HAGEN.

4.3 Model Explainability

To have a better understanding of what HAGEN acquired in the graph learning process, we analyze the hidden relationship revealed by the learned weight matrix by comparing the neighboring nodes of some popular regions in the predefined graph (i.e. geographical distance) and in the learned graph. We take the "University Park" region of Los Angeles as an example in Figure 6 to illustrate the pattern. If we consider the graph defined by geographical distance, University Park is closely connected with regions next to it like Downtown. However, a major university resides in University Park, which has a strong public safety department, and hence it has a very different pattern from its adjacent neighbors in term of crime events.

In contrast, the adaptive region graph is more successful in capturing the crime-related proximity as depicted in the new "neighbors" of University Park in the graph adaptively learned by HAGEN in Figure 6. These learned neighbors can be categorized into two classes: remote and less-populated regions like Chatsworth Reservoir and Harbor City (framed in blue) and more secure regions like Hancock Park (framed

Table 1: Performance comparison with the state-of-the-art baselines on crime forecasting.

Data	Month	Metric	ARIMA	LSTM	GRU	MLP	DT	SVR	RF	MiST	GW	MTGNN	CF	**HAGEN**
CHI	8	Micro-F1	0.4867	0.5032	0.5047	0.5443	0.6226	0.5822	0.6860	0.6719	0.6593	0.6876	0.7052	**0.7209**
		Macro-F1	0.4205	0.4371	0.4339	0.4515	0.4324	0.3424	0.3423	0.6176	0.6201	0.6516	0.6636	**0.6791**
	9	Micro-F1	0.4836	0.4979	0.4818	0.5077	0.6252	0.6064	0.6967	0.6892	0.6512	0.6885	0.6929	**0.7211**
		Macro-F1	0.3979	0.4155	0.4206	0.4326	0.4404	0.3288	0.3587	0.6351	0.6111	0.6524	0.6491	**0.6779**
	10	Micro-F1	0.4757	0.4834	0.4886	0.4240	0.6166	0.6395	0.6933	0.6692	0.6456	0.6832	0.6931	**0.7129**
		Macro-F1	0.3881	0.4031	0.4125	0.3944	0.4396	0.3352	0.3624	0.6211	0.6068	0.6494	0.6506	**0.6738**
	11	Micro-F1	0.4520	0.4495	0.4657	0.5449	0.6108	0.5935	0.6833	0.6766	0.6238	0.6539	0.6774	**0.6946**
		Macro-F1	0.3667	0.3988	0.4059	0.4256	0.4300	0.3296	0.3523	0.6262	0.5849	0.6153	0.6356	**0.6528**
	12	Micro-F1	0.4528	0.4891	0.4655	0.4957	0.6133	0.6261	0.6795	0.6753	0.5933	0.6480	0.6773	**0.6907**
		Macro-F1	0.3697	0.4041	0.4034	0.4146	0.4247	0.3325	0.3540	0.6138	0.5581	0.6064	0.6379	**0.6467**
LA	8	Micro-F1	0.3711	0.4159	0.3931	0.4075	0.5072	0.4569	0.5798	0.5991	0.5699	0.5561	0.6038	**0.6216**
		Macro-F1	0.3036	0.3171	0.3285	0.3236	0.3013	0.2119	0.2054	0.4920	0.5079	0.4871	0.5359	**0.5509**
	9	Micro-F1	0.3668	0.4005	0.4357	0.3636	0.5104	0.4748	0.5849	0.5998	0.5719	0.5792	0.6035	**0.6210**
		Macro-F1	0.2959	0.3118	0.3160	0.3162	0.3019	0.2005	0.2097	0.4877	0.4973	0.5042	0.5180	**0.5443**
	10	Micro-F1	0.3722	0.4010	0.3994	0.3728	0.5147	0.5302	0.5742	0.5956	0.5611	0.5590	0.5886	**0.6165**
		Macro-F1	0.3010	0.3174	0.3197	0.3225	0.3114	0.1992	0.4492	0.4754	0.4891	0.4915	0.5028	**0.5455**
	11	Micro-F1	0.3800	0.4688	0.4643	0.4639	0.5058	0.4899	0.5790	0.5393	0.5711	0.5736	0.5922	**0.6190**
		Macro-F1	0.3090	0.3774	0.3890	0.3797	0.2975	0.2048	0.4519	0.4999	0.5002	0.4994	0.5083	**0.5373**
	12	Micro-F1	0.3730	0.4482	0.4543	0.4481	0.5103	0.5294	0.5685	0.5343	0.5570	0.5663	0.5841	**0.6113**
		Macro-F1	0.3010	0.3849	0.3662	0.3856	0.3033	0.2084	0.4401	0.4944	0.4843	0.4954	0.4921	**0.5250**

in yellow). These neighborhoods better resemble University Park with respect to crime occurrence.

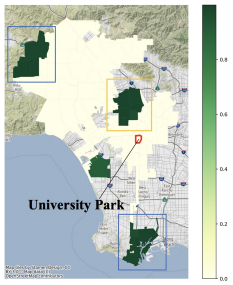


Figure 6: Adaptively learned neighbors of University Park.

5 Related Works

5.1 Forecasting Model for Crime Forecasting

As a prediction problem of the sequential data, it is natural to utilize models like ARIMA and LSTM (Mei and Li 2019; Safat, Asghar, and Gillani 2021) in crime forecasting. To further capture the spatial dependency of crime events as well as temporal dependency, Huang et al. partition the region into synthetic units and combined CNN-based models and RNN-based models in DeepCrime (Huang et al. 2018) and MiST (Huang et al. 2019). The state-of-the-art CrimeForester (Sun et al. 2020) shows that distance-based region graph structure can better capture crime correlations. However, pre-defined graph like distance based weight matrix is hard to take each case into account, unavoidably leading to cases where neighboring nodes do not share similar crime pattern, which limits the performance of GNN-based models. To handle this problem, we propose an adaptive graph structure by introducing graph learning layer into HAGEN that continuously updates during training.

5.2 Spatial-temporal Graph Neural Network

In traffic forecasting domain, many spatial-temporal graph neural network models are proposed such as DCRNN (Li et al. 2018), STGCN (Yu, Yin, and Zhu 2017), GraphWaveNet (Wu et al. 2019) and GMAN (Zheng et al. 2020) to model both spatial correlations and temporal dependencies. Adaptive graph learning is utilized by AGCRN (Bai et al. 2020), SLCNN (Zhang et al. 2020) and MTGNN (Wu et al. 2020) recently, and MTGNN is the state-of-the-art approach in previous papers. However, the current approaches of graph construction are heuristic with consideration of barely node similarity, graph sparsity and symmetry. Homophily, which is one of the fundemental assumptions for GNNs, is not taken into account. We proposed a homophily-aware graph convolutional recurrent network framework by explicitly introducing homophily constraint into our model to regularize the process of graph learning.

6 Conclusion

We presented HAGEN, an end-to-end graph convolutional recurrent network with a novel homophily-aware graph learning module for crime forecasting. In particular, HAGEN uses an adaptive graph structure to capture dependency of crime patterns between regions and incorporates direction-aware diffusion convolution layer with Gated Recurrent Network to learn spatiotemporal dynamics. The graph structure is constrained by a designed homophily-aware loss to enhance performance of graph neural network. We evaluated HAGEN on two real-world benchmarks. HAGEN consistently outperforms state-of-the-art crime forecasting model and spatiotemporal graph neural networks. In future work, we will improve our model and evaluate benchmarks for traffic forecasting to prove HAGEN's generality for geospatial multivariate time series forecasting. Furthermore, we intend to explore theoretical foundations of adaptive graph construction and how it improves multivariate time series forecasting.

Acknowledgments

This research has been funded in part by NSF grants CNS-2027794, CNS-2125530 and IIS-2128661, the USC Integrated Media Systems Center (IMSC), and an unrestricted cash gift from Google and Microsoft Research. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of any of the sponsors such as the NSF.

References

Bai, L.; Yao, L.; Li, C.; Wang, X.; and Wang, C. 2020. Adaptive Graph Convolutional Recurrent Network for Traffic Forecasting. *arXiv preprint arXiv:2007.02842*.

Chang, C.-C.; and Lin, C.-J. 2011. LIBSVM: A library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, 2(3): 1–27.

Chien, E.; Peng, J.; Li, P.; and Milenkovic, O. 2021. Adaptive Universal Generalized PageRank Graph Neural Network. In *International Conference on Learning Representations. https://openreview. net/forum*.

Cho, K.; Van Merriënboer, B.; Bahdanau, D.; and Bengio, Y. 2014. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*.

Chung, J.; Gulcehre, C.; Cho, K.; and Bengio, Y. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.

Contreras, J.; Espinola, R.; Nogales, F. J.; and Conejo, A. J. 2003. ARIMA models to predict next-day electricity prices. *IEEE transactions on power systems*, 18(3): 1014–1020.

Covington, P.; Adams, J.; and Sargin, E. 2016. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM conference on recommender systems*, 191–198.

Gers, F. A.; Schmidhuber, J.; and Cummins, F. 1999. Learning to Forget: Continual Prediction with LSTM. In *Istituto Dalle Molle Di Studi Sull Intelligenza Artificiale*. IET.

Grover, A.; and Leskovec, J. 2016. node2vec: Scalable feature learning for networks. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 855–864.

Huang, C.; Zhang, C.; Zhao, J.; Wu, X.; Yin, D.; and Chawla, N. 2019. Mist: A multiview and multimodal spatial-temporal learning framework for citywide abnormal event forecasting. In *The World Wide Web Conference*, 717–728.

Huang, C.; Zhang, J.; Zheng, Y.; and Chawla, N. V. 2018. DeepCrime: Attentive hierarchical recurrent networks for crime prediction. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, 1423–1432.

Kingma, D. P.; and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Li, Y.; Yu, R.; Shahabi, C.; and Liu, Y. 2018. Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting. In *The International Conference on Learning Representations*.

Lin, Z.; Lyu, S.; Cao, H.; Xu, F.; Wei, Y.; Samet, H.; and Li, Y. 2020. HealthWalks: Sensing Fine-grained Individual Health Condition via Mobility Data. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 4(4): 1–26.

Mei, Y.; and Li, F. 2019. Predictability comparison of three kinds of robbery crime events using LSTM. In *Proceedings of the 2019 2nd International Conference on Data Storage and Data Engineering*, 22–26.

Morenoff, J. D.; and Sampson, R. J. 1997. Violent crime and the spatial dynamics of neighborhood transition: Chicago, 1970–1990. *Social forces*, 76(1): 31–64.

Safat, W.; Asghar, S.; and Gillani, S. A. 2021. Empirical Analysis for Crime Prediction and Forecasting Using Machine Learning and Deep Learning Techniques. *IEEE Access*.

Safavian, S. R.; and Landgrebe, D. 1991. A survey of decision tree classifier methodology. *IEEE transactions on systems, man, and cybernetics*, 21(3): 660–674.

Sun, J.; Yue, M.; Lin, Z.; Yang, X.; Nocera, L.; Kahn, G.; and Shahabi, C. 2020. CrimeForecaster: Crime Prediction by Exploiting the Geographical Neighborhoods' Spatiotemporal Dependencies. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 52–67. Springer.

Verikas, A.; Gelzinis, A.; and Bacauskiene, M. 2011. Mining data with random forests: A survey and results of new tests. *Pattern recognition*, 44(2): 330–349.

Wu, Z.; Pan, S.; Long, G.; Jiang, J.; Chang, X.; and Zhang, C. 2020. Connecting the dots: Multivariate time series forecasting with graph neural networks. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 753–763.

Wu, Z.; Pan, S.; Long, G.; Jiang, J.; and Zhang, C. 2019. Graph WaveNet for Deep Spatial-Temporal Graph Modeling. In *Twenty-Eighth International Joint Conference on Artificial Intelligence IJCAI-19*.

Xu, F.; Lin, Z.; Xia, T.; Guo, D.; and Li, Y. 2020. Sume: Semantic-enhanced urban mobility network embedding for user demographic inference. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 4(3): 1–25.

Yu, B.; Yin, H.; and Zhu, Z. 2017. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. *arXiv preprint arXiv:1709.04875*.

Zhang, Q.; Chang, J.; Meng, G.; Xiang, S.; and Pan, C. 2020. Spatio-temporal graph structure learning for traffic forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 1177–1185.

Zheng, C.; Fan, X.; Wang, C.; and Qi, J. 2020. Gman: A graph multi-attention network for traffic prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 1234–1241.

Zhu, J.; Yan, Y.; Zhao, L.; Heimann, M.; Akoglu, L.; and Koutra, D. 2020. Beyond Homophily in Graph Neural Networks: Current Limitations and Effective Designs. *Conference on Neural Information Processing Systems*, 33.