RESEARCH-ARTICLE

Enhancing Graph Neural Networks in Large-scale Traffic Incident Analysis with Concurrency Hypothesis

XIWEN CHEN, Clemson University, Clemson, SC, United States

SAYED PEDRAM BOROUJENI, Clemson University, Clemson, SC, United States

XIN SHU, Northeastern University, Boston, MA, United States

HUAYU LI, The University of Arizona, Tucson, AZ, United States

ABOLFAZL RAZI, Clemson University, Clemson, SC, United States

Published: 29 October 2024

SIGSPATIAL '24: The 32nd ACM International Conference on Advances in Geographic Information Systems
October 29 - November 1, 2024
GA, Atlanta, USA

Conference Sponsors:
SIGSPATIAL

https://doi.org/10.1145/3678717.3691256
ISBN: 9798400711077

Enhancing Graph Neural Networks in Large-scale Traffic Incident Analysis with Concurrency Hypothesis

Xiwen Chen
Sayed Pedram Haeri Boroujeni
{xiwenc,shaerib}@g.clemson.edu
Clemson University
Clemson, SC, USA

Xin Shu
shu.xin@northeastern.edu
Northeastern University
Boston, MA, USA

Huayu Li
hl459@arizona.edu
University of Arizona
Tucson, AZ, USA

Abolfazl Razi
arazi@clemson.edu
Clemson University
Clemson, SC, USA

Abstract

Despite recent progress in reducing road fatalities, the persistently high rate of traffic-related deaths highlights the necessity for improved safety interventions. Leveraging large-scale graph-based nationwide road network data across 49 states in the USA, our study first posits the Concurrency Hypothesis from intuitive observations, suggesting a significant likelihood of incidents occurring at neighboring nodes within the road network. To quantify this phenomenon, we introduce two novel metrics, Average Neighbor Crash Density (ANCD) and Average Neighbor Crash Continuity (ANCC), and subsequently employ them in statistical tests to validate the hypothesis rigorously. Building upon this foundation, we propose the Concurrency Prior (CP) method, a powerful approach designed to enhance the predictive capabilities of general Graph Neural Network (GNN) models in semi-supervised traffic incident prediction tasks. Our method allows GNNs to incorporate concurrent incident information, as mentioned in the hypothesis, via tokenization with negligible extra parameters. The extensive experiments, utilizing real-world data across states and cities in the USA, demonstrate that integrating CP into 12 state-of-the-art GNN architectures leads to significant improvements, with gains ranging from 3% to 13% in F1 score and 1.3% to 9% in AUC metrics. The code is publicly available at https://github.com/xiwenc1/Incident-GNN-CP[1].

CCS Concepts

• **Information systems** → **Location based services**; **Geographic information systems**; • **Computing methodologies** → **Semi-supervised learning settings**.

Keywords

Road Network Analysis, Graph Analysis, Graph Neural Network

[1]We tend to use the term `incident` rather than `accident` according to the preference of the Department of Transportation. They may be interchangeably used in our paper.

SIGSPATIAL '24, October 29-November 1, 2024, Atlanta, GA, USA

ACM ISBN 979-8-4007-1107-7/24/10
https://doi.org/10.1145/3678717.3691256

ACM Reference Format:
Xiwen Chen, Sayed Pedram Haeri Boroujeni, Xin Shu, Huayu Li, and Abolfazl Razi. 2024. Enhancing Graph Neural Networks in Large-scale Traffic Incident Analysis with Concurrency Hypothesis . In *The 32nd ACM International Conference on Advances in Geographic Information Systems (SIGSPATIAL '24), October 29-November 1, 2024, Atlanta, GA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3678717.3691256

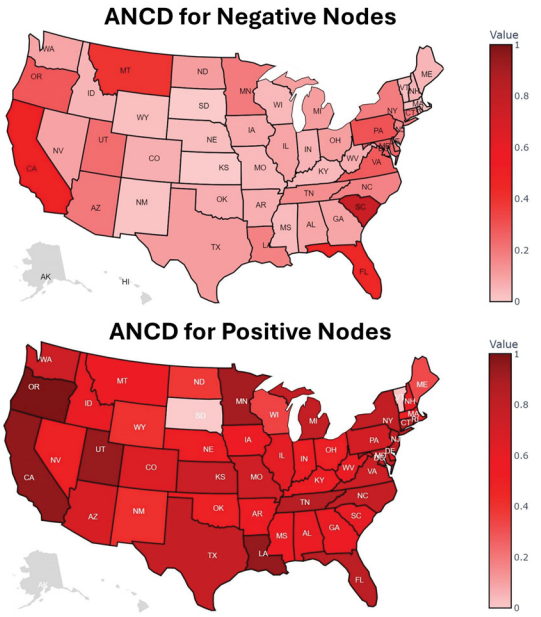

Figure 1: The state-wise Average Neighbor Crash Density (ANCD) for Top: negative nodes (i.e. nodes without incident records) and Bottom: positive nodes (i.e. nodes with incident records) when $k = 1$. For a specific class of nodes (i.e. positive/negative nodes), a deeper color denotes a higher density of their neighbor nodes have incident. It is observed that, *within the same state*, the neighbor nodes of positive nodes often exhibit a higher crash density than those of negative, which supports our Concurrency Hypothesis.

1 Introduction

The significance of traffic safety is underscored by recent statistics, which highlight the ongoing challenges and successes in reducing road fatalities. In the early months of 2023, the National Highway Traffic Safety Administration (NHTSA) [34] reported a decrease in traffic fatalities, estimating that 9,330 people lost their lives in traffic crashes in the first quarter, marking a 3.3% decline compared to the previous year. This trend continued into the first half of the year, with fatalities dropping to 19,515, also down by about 3.3% from the prior year. However, despite the positive trend in these numbers, they still reflect a high rate of traffic-related deaths, especially compared to the earlier years (2005-2019). This situation emphasizes the ongoing need for enhanced traffic safety measures and interventions to sustain and accelerate the reduction in road fatalities.

The importance of modeling traffic incident risks is well recognized in the field of urban planning and public safety [11, 40]. Accurate predictions of where and when incidents are likely to occur can significantly contribute to the development of more effective traffic management strategies and infrastructure improvements. As urban areas continue to grow and also traffic volume increases, the need for powerful analytical tools to assess risk and prevent incidents becomes increasingly critical. It requires the integration of comprehensive datasets and advanced analytical techniques to understand the complex dynamics of road traffic and enhance safety measures. By leveraging detailed geospatial data and traffic incident records, researchers and city planners can identify high-risk areas and implement specific interventions to mitigate the possible risks, while enhancing road safety [36]. Recently, many studies have analyzed the effect of road features for predicting incident occurrences, such as [37, 35, 7, 33, 61, 60]. More recently, Deep Learning (DL)-based methods have gained significant attention in traffic safety analysis since their powerful ability to characterize the inherent complex features of large-scale data [54, 41, 58, 40]. Due to the nature that both the road network and traffic flow can be viewed as graph structure data, Graph Neural Network (GNN) [23, 47, 16, 10, 51, 26, 30, 57] is the rational choice to characterize the relations in a network and has been adopted in recent works [27, 56, 62].

Our work is motivated by two intuitive observations in traffic incident occurrences in road networks: When people are driving and notice a traffic incident, there is a high probability that they observe another incident has occurred nearby. Another observation is that there are always some accident-prone sections, meaning the continuous areas included in the sections are likely to have incidents even if they have not occurred necessarily at the same time. We then make a unified hypothesis for them:

Concurrency Hypothesis. *There is a high probability when a node has an incident occurred, some of its neighbors have an incident occurred.*

Subsequently, we propose two novel metrics, the Average Neighbor Crash Density (ANCD) and Average Neighbor Crash Continuity (ANCC), to quantify these observations, and apply standard statistical tests for these quantitative results to validate the proposed hypothesis. An exemplary visualization of ANCD for each state is shown in Fig. 1, which underscores the difference of the nodes between different categories for each state. We then conjecture that this hypothesis indicates that there may be some important but difficult-to-capture information and features that have not been fully collected by the general datasets. Accordingly, in this work, we proposed an enhancement method called *Concurrency Prior* (CP) that explores the hidden information beyond the common features from the crash label for semi-supervised traffic incident prediction. This problem is built on a single monolithic graph representing an entire state or city. Entire edge features, entire node features and partial nodes' labels are known. Our goal is to utilize the known information to learn a model and predict the label for the rest of the nodes with unknown labels. The formal problem description is in Section 4. Our method is compatible and complementary with general graph neural networks, such as Graph Convolutional Networks [23], Graph Attention Networks [47], and Graph Transformers [44]. Our investigation is based on the nationwide real-world road network data provided by [18]. This large-scale data source contains the incident record from 49 states of the USA and provides various edge features, such as length, type, number of lanes, max speed, and road direction and angular information. We provide the details of the data acquisition in Section 3.

In summary, our contribution is two-fold: **(i)** We are the first to statistically validate the *Concurrency Hypothesis* in nationwide graph-based data by using our proposed metrics; and **(ii)** We propose an enhancement method called *Concurrency Prior* that enables boosting broad variations of graph neural networks in the semi-supervised traffic incident prediction task by introducing negligible parameters. Our intensive experiments on 12 state-of-the-art graph neural networks demonstrate a 3%-13% and 1.3%-9% gain in F1 and AUC, respectively.

2 Related Work

It is known that most traffic-related data can be viewed as network/-graph structure data. For example, vehicles involved in an incident can be treated as a network, where each node is a vehicle, and the edge denotes the interaction between every two vehicles. Besides, the road network can be treated as a graph, where the nodes denote different physical locations, and the edges denote roads between them. Hence several previous works focus on network analysis [8, 50, 12, 3, 24] and many works analysis it based on knowledge from the complex network theories, such as small-world networks [48] and random scale-free networks [1]. However, this type of data is not as easy to process as usual data (e.g., images) in the machine learning community. This is due to the fact the topology of graph data is often variable and enormous [38]. For example, considering a city as a graph, different cities apparently have totally different topologies and may have massive intersections as nodes. Therefore, graph Neural Networks (GNNs) have gained significant attention for processing graph data.

The early concept of GNNs can be traced back to 2008 [42], when the authors proposed a framework that leverages a recurrent neural network (RNN) structure for graph data. Afterward, authors in [5, 17] apply spectral approaches to GNNs, where they perform convolutions via graph Laplacian. However, spectral approaches introduce an intensive computation cost and lack generalizability

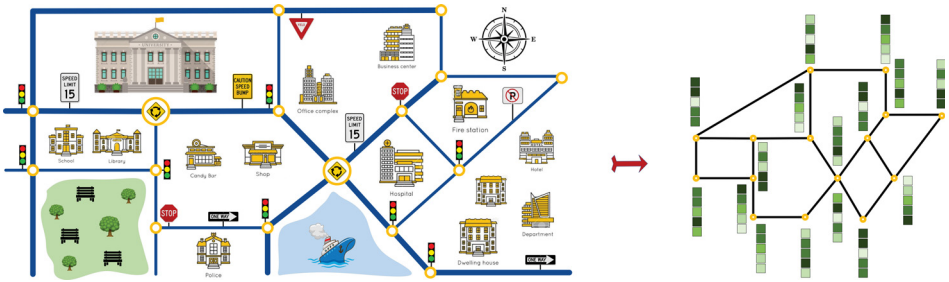

Figure 2: The graph-based data is obtained from real-world road networks.

across different graphs. To tackle these issues, authors in [23] propose Graph Convolutional Networks (GCNs), which significantly reduce computational complexity while maintaining performance. This method then becomes the cornerstone in the development of GNNs, and subsequently, there are a large number of variants [47, 16, 10, 51, 26]. For example, authors in [47] introduce attention mechanisms to model the importance of nodes' neighbors, while authors in [16] propose an inductive framework that learns node embeddings by sampling and aggregating features from a node's local neighborhood. Authors in [10] propose a topology adaptive mechanism for graph convolution. In 2020, authors in [26] developed a framework for training very deep GCNs using differentiable generalized aggregation functions and a novel normalization layer called MsgNorm, effectively addressing vanishing gradients, over-smoothing, and over-fitting issues in very deep GCN models. More recently, authors in [44] propose an adaptation of the transformer architecture to graph-structured data, providing an alternative to convolution-based methods.

GNNs illustrate the obvious superiority of capturing the dependencies of graph-based data in non-euclidean space, while these dependencies are challenging to learn by the algorithms designed for Euclidean space (e.g,. Convolutional Neural Networks (CNN)). Therefore, GNNs have been widely used in traffic and intelligence transportation fields [53, 39]. For instance, authors in [59, 15, 45, 20] employ GNN for traffic flow prediction. Demand Prediction is also a popular task that can benefit from GNN. These demands include ride-hailing demand forecasting [13, 22, 19], bike sharing systems [28, 25], and passenger flow prediction [29]. Additionally, GNN is used in point-cloud-based perception [43, 21], motion prediction [46], and planning [6] in the studies of autonomous vehicles. Likewise, there are several works aiming to predict the incident occurrence. Specifically, [62] develops a novel differential time-varying GCN to dynamically capture traffic variations and [56] proposes a spatio-temporal GCN and employs the embedding layer to remove noises and better extract semantic representations of external information.

The most related work is [18], which performs the traffic analysis with nationwide coverage and real-world network topology and tries to solve the classification problem solely based on a single monolithic graph. It is noteworthy that none of the previous works mentioned above has performed analysis on such a large data scale. Our work substantially enhances the prediction performance over this work, as well as several popular aforementioned GNN variants, by imposing our proposed concurrency prior to neural networks. The proposed training strategy mentioned in Section 5.3 is related to attribute masking used in [55, 30]; however, we use it to mimic the real node-wise inference in the training phase when incorporating the concurrency information (Eq. 9). Hence, our training strategy is essentially different and orthogonal from theirs, since they only employ it as a common data augmentation method.

3 Data Acquisition

In our study, we use nationwide traffic incident data consists over 1,000 U.S. city-level datasets and 49 U.S. state-level datasets [18]. In this section, we delve into the key concept behind the creation of graph-based traffic incident benchmarks with datasets that contain real-world geospatial features. The traffic incident processing repository is developed by collecting a comprehensive set of raw data on traffic incidents [32]. It involves detailed information about incident records, the geographical layout of streets, and the relational structure of these locations represented in graph form. To enhance the utility of the incident location data, a reverse geocoding process is employed to convert geographic coordinates into more accessible address formats. Afterward, the crash information is integrated with the graph-structured data and geographical attributes to create a cohesive and structured dataset, as shown in Fig. 2. The foundational data for these datasets are sourced from Microsoft Bing Map Traffic [31], extracted explicitly from the US-Accidents benchmarks. These datasets serve as a valuable source of information, documenting around 2.8 million traffic-related incidents over a period from January 2016 through December 2021. They provide a detailed account of traffic events during this time frame, offering insights into patterns and trends.

In these datasets, OpenStreetMap (OSM) [4] is employed as the primary resource for obtaining geospatial data information. The collected data from OSM are enriched with a variety of geographical information, including roads, trails, railway stations, land Use, land cover, transport networks and natural landmarks like forests and rivers. This information is tagged under different OSM classes, which serve to present the specific characteristics of the geographic elements in the database, such as nodes (defining points like intersections), ways (paths or open areas), and edges (logical or physical relationships between elements). Hence, the datasets now have rich features, including but not limited to the type of road, the length of a road, the number of lanes, one-way indication, max speed, tunnel indication, junction type, etc. Besides, directional and angular features of road networks are identified, enhancing the dataset with unique geometric insights.

Afterward, incident data is first reverse geocoded to pinpoint exact locations, and then systematically organized based on settlement hierarchies from villages to states. Datasets are divided into two main subsets: city-level and state-state, each tailored to different scales of traffic analysis. The city-level datasets focus on urban areas where traffic incident frequency is higher, reflecting the denser road networks and population distribution. In contrast, state-level datasets provide a broader perspective, suitable for regional traffic trends and policy planning. Eventually, the integration process involves sophisticated data processing techniques like one-hot encoding and spatial analysis used to correlate accident sites with nearby road network nodes.

In summary, each dataset (either a city or a state) is a single monolithic graph, which refers to a unified and comprehensive graph structure that includes all data points (nodes) and relationships (edges) contained within one comprehensive graph without division into subgraphs. Suppose a dataset (can be a specific state or city) is a large graph that has N nodes and E edges, then the dataset can be presented by three matrices $\boldsymbol{A} \in \mathbb{R}^{N\times N}$, $\boldsymbol{X} \in \mathbb{R}^{N\times D_1}$, and $\boldsymbol{E} \in \mathbb{R}^{E\times D_2}$, and $\boldsymbol{Y} \in \mathbb{R}^{N}$, denoting the adjacency matrix, node embedding, edge embedding, and node labels, respectively. Here, D_1 and D_2 denote the number of dimensions of node and edge features, respectively. It also should be noted that these datasets are significantly unbalanced, and a very low ratio of points is positive (i.e. nodes with crash records). The statistics of the node labels are shown in Fig. 3 and Table 2. We realize this may substantially challenge most machine learning algorithms.

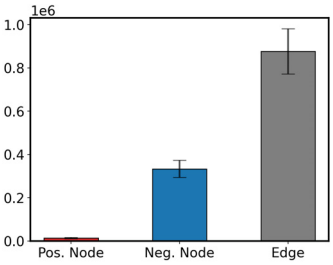

Figure 3: The statistics of the graph-based datasets on 49 states.

4 Problem Formulation

Here, we give the problem formulation of our study. Suppose the label for a node i is Y_i. Our problem is semi-supervised and in a transductive setting. It means some nodes have known labels ($Y_i, \forall i \in \mathcal{V}_{train}$), and others are unknown ($Y_i, \forall i \in \mathcal{V}_{test}$) that we aim to predict them. Note that $\boldsymbol{A}$, $\boldsymbol{X}$, and $\boldsymbol{E}$ are fully known in both the training and inference phases. We use $Y_i = 1$ to denote the positive nodes that there is an incident occurred while $Y_i = 0$ denotes the negative nodes that nothing happened here. An illustration of the problem is shown in Fig. 4 (**Left**), where red and blue denote the nodes with known labels (crash/no crash), and the question marks denote the unknown labels.

5 Methodology

In this section, we first adopt statistical tools to validate the concurrency hypothesis (Section 5.1), and then motivated by the effectiveness of this hypothesis, we propose the concurrency prior, which is an enhancement method for general graph neural networks in crash prediction (Section 5.2 and Section 5.3).

5.1 Statistical Analysis of the Concurrency Hypothesis

Recap our **Concurrency Hypothesis**: *There is a high probability when a node has an incident occurred, some of its neighbors have an incident occurred.*

To validate this hypothesis, we propose two quantitative tools, and we expect to demonstrate that there is a statistical difference between the incident occurrence of the neighbors of positive nodes and negative nodes. The metrics are calculated state by state as each state is a monolithic graph.

Average Neighbor Crash Density (ANCD). This metric is first calculated for each node of a dataset (i.e. a state here) as,

$$NCD_i = \frac{\sum_{j\in neighbor_k(i)} \mathbb{I}(Y_j = 1)}{|neighbor_k(i)|}. \tag{1}$$

where $neighbor_k(i)$ denotes the set of neighbor nodes of node i that can arrive in most k hops through the connected edges. $\mathbb{I}(\cdot)$ and $|\cdot|$ denote the indicator function and the cardinality of a set, respectively. Then, ANCD can be computed for positive nodes ($z = 1$) and negative nodes ($z = 0$) of the dataset, respectively,

$$ANCD_z = \frac{\sum_{i\in\{i|Y_i=z\}} NCD_i}{|\{i|Y_i = z\}|}, \tag{2}$$

where $z \in \{0, 1\}$. ANCD can be interpreted as the average density of the neighbor nodes that have crashes for a specific class of nodes.

We are also interested in the distance of the nearest positive nodes of nodes in a specific class. However, in a super-large graph, computing the distance in the form of hops is challenging due to the computation cost and memory issues; therefore, we propose the surrogate metrics to estimate it. **Average Neighbor Crash Continuity (ANCC).** The metric aims to calculate how the continuity of neighbor crash nodes.

$$NCC_i = \begin{cases} 1, & \exists Y_j = 1, j \in neighbor_k(i), \\ 0, & \text{Else.} \end{cases} \tag{3}$$

The NCC_i can be interpreted as follows: if NCC_i is equal to 1, the nearest positive node of node i is at most k hops. In contrast, if NCC_i is equal to 0, the nearest positive node of node i requires at least $k + 1$ hops. Subsequently, we can compute $ANCC_z$ similar to Eq. 2,

$$ANCC_z = \frac{\sum_{i\in\{i|Y_i=z\}} HR_i}{|\{i|Y_i = z\}|}. \tag{4}$$

Hence $ANCC_z$ is able to evaluate the average distance to the nearest positive nodes of the class z nodes from a different perspective.

To comprehensively evaluate the difference between negative nodes and positive nodes, after computing the two metrics for each state, *paired t-test* is used to offer support from the standard hypothesis test. To perform a paired t-test, we first compute the difference between paired observations, denoted as $d^j = M_0^j - M_1^j$,

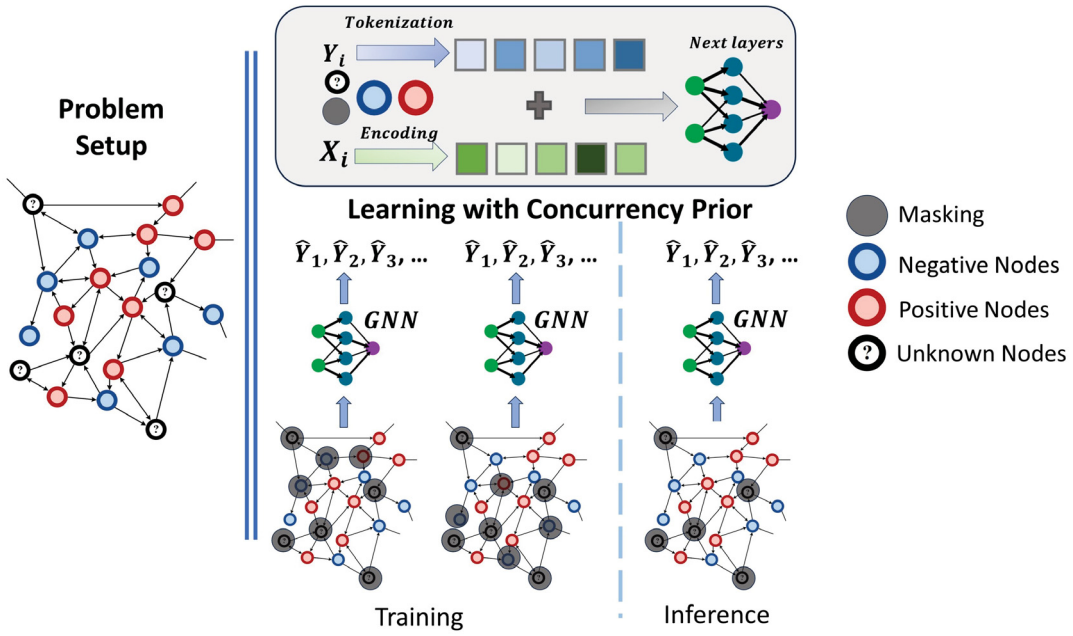

Figure 4: Left: The problem is formulated in a single large graph. Nodes' labels are known if the node is marked in colors, i.e., red (positive)/ blue (negative). The nodes with question marks are expected to be predicted. Middle: In the training phase, we keep all unknown nodes with the uncertain token 'o', and in each iteration, we also randomly mask some nodes with known labels to 'o' to mimic the prediction process. Right: In the inference phase, we only keep to nodes to be predicted with the uncertain token 'o'. Top: Imposing concurrency prior to the neural network.

where M denotes one of the proposed metrics with predefined k, and j denotes the state index. Suppose the μ_d represents the population mean difference. The null hypothesis (H_0) and alternative hypothesis (H_a) for the paired t-test are typically defined as follows:

- Null hypothesis (H_0): There is no significant difference between the paired observations, i.e., $\mu_d = 0$.
- Alternative hypothesis (H_a): There is a significant difference between the paired observations, i.e., $\mu_d < 0$.

If there is statistical significance, the incident occurrence in a node's neighbors that has a correlation with the status of this node. We present the results in Section 6.1.

5.2 Graph Neural Networks with Concurrency Prior

As the Concurrency Hypothesis exists, we conjecture that there is information included in the label Y_i in addition to the original features $\boldsymbol{X}$ and $\boldsymbol{E}$. Hence, employing this information may enhance the model's learning ability. Here, a prediction for a node i by a conventional graph neural work G in the problem is presented as,

$$\hat{Y}_i = G_i(\boldsymbol{A}, \boldsymbol{X}, \boldsymbol{E}). \tag{5}$$

In our method, we want to explicitly adopt the information, which results in a prediction as,

$$\hat{Y}_i = F_i(\boldsymbol{A}, \boldsymbol{X}, \boldsymbol{E}, \{Y_j | j \in \mathcal{V}_{train}\}). \tag{6}$$

Theorem 1. *If we use the mutual information $I(\cdot;\cdot)$ to denote the upper bound of the learning ability of a network, apparently, F_i should have a stronger potential of learning ability. This is because,*

$$I(Y_i; \boldsymbol{A}, \boldsymbol{X}, \boldsymbol{E}) \leq I(Y_i; \boldsymbol{A}, \boldsymbol{X}, \boldsymbol{E}, \{Y_j | j \in \mathcal{V}_{train}\}). \tag{7}$$

Since the concurrency information is usually presented as discrete labels, it is now impossible to directly present any semantic information to the neural network. Therefore, we tokenize labels as a learnable dictionary, and each instance (one vector) of the dictionary represents the latent feature of each category. This strategy is much more friendly for learning the neural network because, with tokenization, all operations are in continuous space, which allows us to optimize a neural network with concurrency information just like training a common network. Additionally, we use an efficient way to embed the concurrency information without introducing considerable parameters. Thereby, the *Concurrency Prior* can be imposed to the neural network as,

$$\boldsymbol{X}_i \leftarrow encode(\boldsymbol{X}_i) + Tokenization(Y_i). \tag{8}$$

We impose Concurrency Prior in the embedding space because the original features consist of data from different concepts (see Section 3), and the $encode(\boldsymbol{X}_i)$ can be viewed as these feature after fusion, which offers a better representation. We aggregate the feature information ($\boldsymbol{X}_i$) and Concurrency Prior ($Tokenization(Y_i)$) by summation because this way allows us not to change the original network architecture and hence not introduce extra parameters (except the few parameters by tokenization). For example, if the original network has one linear layer with $d_1 \times d_2$ parameters. Alone with the parameters introduced by tokenization, if aggregating by concatenating, the architecture should be modified and has $((d_1 + d_{cp}) \times d_2$

parameters accordingly, where d_1,d_2, and d_{cp} are the number of the original input, output, and CP dimensions. In contrast, the total parameters in our method are consistently $d_1 \times d_2$. Hence, the total number of introduced parameters by imposing Concurrency Prior is $(C+1) \times d$ for the set of learnable vectors for tokenization, where C and d denote the number of classes and the embedding size of the feature in the original architecture, respectively. The additional one (i.e. 1 in $C+1$) in the term denotes the uncertain class, which we will discuss in the next section. When $C = 2$ in our case, the introduced parameters are negligible.

5.3 How to train the neural network?

To train the neural network, for each node $i \in \mathcal{V}_{train}$, we anticipate minimizing the loss for each node,

$$\min_{F_i} \mathcal{L}(Y_i, F_i(\boldsymbol{A}, \boldsymbol{X}, \boldsymbol{E}, \{Y_j | j \in \mathcal{V}_{train} \setminus Y_i\})), \tag{9}$$

where F_i denotes the classifier for node i. Another challenge is posed here since a general graph neural network is designed to process a graph with arbitrary shapes, and training a classifier for each node is inefficient due to the massive number of nodes (e.g., 1169400 nodes in California dataset); therefore, the network often has a unified classifier for all nodes. A general training strategy is feeding $\boldsymbol{A}, \boldsymbol{X}, \boldsymbol{E}$ to the network to predict all Y_i, which assumes the prediction for all nodes uses the same input (i.e. $\boldsymbol{A}, \boldsymbol{X}, \boldsymbol{E}$), where our network is not fulfilled. In our case, we need to feed the feature and label of a node to the network; however, the label of the nodes from the test set is unknown, and we also need to exclude the label information of the target node (i.e. $\hat{Y}_i = F_i(\boldsymbol{A}, \boldsymbol{X}, \boldsymbol{E}, \{Y_j | j \in \mathcal{V}_{train} \setminus Y_i\})$) during training. To tackle these issues, we first introduce the uncertain token o as a placeholder for the nodes without knowing the label information (i.e. test set). Then, in each iteration, we randomly set the labels of partial training nodes to o to mimic the prediction processing that excludes the label information of the target nodes (shown in Fig. 4 (**Middle**)). With these proposed methods, we can train any graph neural network with Concurrency Prior in the same way as a common network. During inference, we will feed all known labels of training nodes to the GNN for the prediction (shown in Fig. 4 (**Right**)). A summary of our proposed method is presented in Algorithm 1.

Algorithm 1 PyTorch Code for a general GNN with CP.

```
#input:  Hidden dim: d, number of classes: C, Node
     feature: X (N*D1), Adj. Matrix: A  (N*N), Edge
     Feature: E  (N*D2), hard label: Y (shape N*1), and
     train/valid/test indices: V_train, V_valid, V_test.
     Y[V_test] is unknown. Mask rate: R (0<R<1).

#output: Predicted probability for each node.
#The loss is only computed for all training nodes.


class GNNwithCP(torch.nn.Module):
    def __init__(self, hidden_dim=d,number_class =C):
        super(GNNwithCP, self).__init__()
        #define CP embedding
        self.CP_embeeding = nn.Embedding(dataset.
     num_classes+1,hidden_dim) #one class for masked
     nodes


        self.encoder_1 = ...  #output size should be
     hidden_dim
        self.encoder_other = ...
        self.fc = nn.Linear(..., C)#The unified
     classifier for all nodes

    def forward(self,X,A,E,Y,M):

        token = torch.zeros_like(Y).to(Y.device)
        token[V_train] =  Y[V_train]+1 # We only know
     training nodes' labels, and others set to 0 meaning
     unknown. Original label 0->1, 1->2.

        if self.train: #only masking during training.
            select_index = random.sample(range(len(Y)),
     int(R*len(Y)))
            token[select_index]=0

        token_embeeding = self.CP_embeeding(token)
        X,E = self.encoder_1(X,A,E)
        X = X+token_embeeding #Eq. 8

        X,E = self.encoder_other(X,A,E)
        X = self.fc(X)
        return F.log_softmax(X, dim=1)
```

6 Experiment

6.1 Statistical Analysis Results

We consider the number of available hops k to $k \in \{1, 2, 4, 8, 10\}$ in our experiments. The results of these paired t-tests conducted in Section 5.1 are shown in Table 1, where p-values are tiny for all tests (i.e. less than 1E-18). These results exhibit that we have very high confidence to conclude: in each state, the metrics ANCD and ANCC computed for positive nodes are statically higher than those for negative nodes, which supports our hypothesis that if a node has an incident occurred, its neighbors are likely to have incidents. A summary of the metrics is presented in Fig. 5, which illustrates another interesting observation. We find that as the k increases, the difference between the value of positive nodes and negative nodes decreases, which may suggest that the concurrency hypothesis has a high locality that indicates a label of a node is related to its closer neighbors. We also present the metrics for each state in Fig. 1 to highlight their difference.

Table 1: The p-value of the paired-test. k denotes the number of available hops.

k	1	2	4	8	10
ANCD	1.13E-29	1.90E-28	5.84E-26	2.69E-21	4.53E-19
ANCC	6.20E-32	2.13E-37	3.42E-34	1.56E-23	1.05E-19

6.2 Main Experiment

Datasets. We evaluate our proposed methods both city-wise and state-wise. Following [18], we choose six representative cities, including Miami, Los Angeles, Orlando, Dallas, Houston, and New York. We also choose six representative states, including California,

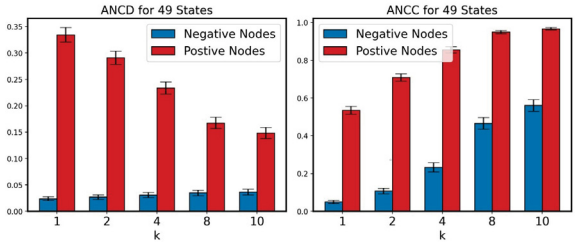

Figure 5: The statistics of ANCD and ANCC computed for the available 49 states.

Table 2: The description of the selected datasets in our experiments.

Dataset	California	Oregon	Utah	Maryland	Minnesota	Connecticut
# of Nodes	1169400	217619	142478	234475	370383	120642
# of Edges	2915853	544776	362667	557249	965962	304417
positive Ratio	0.106	0.072	0.059	0.057	0.052	0.042
Dataset	**Miami (Florida)**	**Los Angeles (California)**	**Orlando (Florida)**	**Dallas (Texas)**	**Houston (Texas)**	**New York (New York)**
# of Nodes	8461	49251	7513	36150	59711	55404
# of Edges	22648	135547	18216	92348	148937	140005
positive Ratio	0.133	0.130	0.302	0.258	0.221	0.083

Oregon, Utah, Maryland, Minnesota, and Connecticut. A summary of these datasets is presented in Table 2. All datasets are stratified split into 60% training/20% validation/20% testing.

Baselines. We select 12 state-of-the-art GNN models, including (1) **GCN**: Graph Convolutional Networks [23], (2) **ChebNet**: Chebyshev spectral graph convolution networks [9], (3) **ARMANet**: Graph neural networks with convolutional auto-regressive moving average (ARMA) filters [2]. (4) **GraphSAGE**: A general framework for inductive representation learning on graphs [16], (5) **TAGCN**: Topology adaptive graph convolutional networks [10], (6) **GIN**: Graph Isomorphism Networks [52, 49], (7) **GAT**: Graph attention networks [47], (8) **MPNN**: Message Passing Neural Network [14], (9) **CGC**: Crystal graph convolutional neural network [51], (10) **GEN**: GENeralized graph convolutional neural networks [26], (11) **Graphformer**: Graph transformers [44], and (12) **TRAVEL**: a GNN designed for road network analysis [18].

Evaluation Metrics. We use F1 score and Area Under the Receiver Operating Characteristic Curve (AUC) to evaluate the performance, since these datasets are obviously unbalanced, as presented in Section 3 as well as Table 2.

Implementation Detail. Our training implementation strictly follows [18] and keeps the exact same structure and training hyperparameters (e.g., Optimizer, learning rate, weight decay, dropout, etc.). More details can be found in our source code. We find the reported results in [18] are highly reproducible; hence, we directly use their results in our comparison. For our methods, we report the average result and its standard deviation over 10 runs.

Main Results. The numerical results are shown in Table 3 and Table 4. The main observation is that by imposing our proposed Concurrency Prior, all twelve GNN methods exhibit a considerable improvement across the cities and states. Specifically, in the city-wise datasets shown in Table 3, GNN can obtain a 1.31% to 12.05% gain in F1 score and a 1.81% to 5.48% gain in AUC by imposing our prior. We note that GCN and its early variants, including ChebNet, ARMANet, GraphSAGE, TAGCN, GIN, and GAT are significantly boosted to 46.58%, 44.94%, 42.99%, 48.06%, 46.82%, and 45.24% in F1 score, resulting in their performance being comparable with the original version of Graphformer (45.13%). More importantly, our methods can still improve the previous best method (TRAVEL) with 1.31% and 2.01% gain in F1 score and AUC, respectively. Likewise, as shown in Table 4, our method can consistently enhance all GNN methods in state-wise datasets, which is on a larger geographic scale. We can observe a 3.26% to 13.67% and 1.33% to 9.07% gain in F1 and AUC, respectively. Similarly, the previous best method, TRAVEL, is boosted to 90.96% in AUC, which is a high enough performance in such unbalanced datasets. These results underscore the value of integrating the proposed Concurrency Prior enhancements into GNNs for traffic incident prediction tasks. The visualization of the prediction for different GNN models on different cities and states is presented in Figs. 7 and 8, showcasing a remarkable visual enhancement. Due to the page limit, we provide more visualization in our GitHub repository: https://github.com/xiwenc1/Incident-GNN-CP.

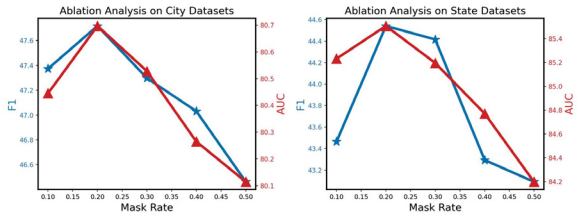

Figure 6: The ablation analysis for different masking rate.

Ablation Analysis. The only hyperparameter in our proposed method is the rate of masking; hence here, we conduct the experiment to test the effect by different $R \in (0, 0.5)$. We present the average F1 and AUC across all selected states and cities in Fig. 6.

7 Conclusion

In this paper, we perform the traffic road network analysis using a large-scale graph-based nationwide data source, including incident records across 49 states in the USA. we first propose two metrics, Average Neighbor Crash Density (ANCD) and Average Neighbor Crash Continuity (ANCC), to statically validate the intuitive concurrency hypothesis, where there is a high probability of incidents occurring in neighboring nodes of a road network. Based on this validation, we then propose our novel Concurrency Prior (CP) method that can incorporate this concurrency information into various GNN models with neglectable extra parameters. Our experiment showcases a remarkable improvement in the semi-supervised graph-based traffic incident prediction tasks. We expect our contributions will be able to offer promising directions for future research and practical applications in urban planning and public safety.

Acknowledgements

This material is based upon the work supported by the National Science Foundation under Grant Number 2204721 and partially supported by our collaborative project with MIT Lincoln Lab under Grant Number 7000612889.

Table 3: City-wise accident occurrence prediction results in terms of F1 score and AUC. Δ denotes the gain obtained by imposing our proposed Concurrency Prior (with the suffix "-CP") in the neural networks.

Dataset	Mean		Miami (FL)		Los Angeles (CA)		Orlando (FL)		Dallas (TX)		Houston (TX)		New York (NY)	
Method	F1	AUC	F1	AUC	F1	AUC	F1	AUC	F1	AUC	F1	AUC	F1	AUC
GCN	34.53	72.88	20.0±3.3	68.5±3.3	40.2±1.1	80.4±0.3	51.6±0.8	73.1±1.2	39.8±1.9	73.1±0.4	16.4±1.3	66.7±0.2	39.2±3.7	75.5±0.4
GCN-CP	46.58	76.32	46.49±2.83	80.35±0.88	53.68±1.57	83.56±0.17	56.38±3.15	77.40±1.15	45.15±1.30	71.86±0.41	31.63±1.21	63.90±0.22	46.13±0.81	80.83±0.25
Δ	**+12.05**	**+3.44**	+26.49	+11.85	+13.48	+3.16	+4.78	+4.30	+5.35	-1.24	+15.23	-2.80	+6.93	+5.33
ChebNet	36.72	75.45	20.7±2.9	71.3±3.6	39.8±1.8	81.0±0.3	53.1±0.6	76.7±1.6	42.0±0.5	75.8±0.4	23.8±0.5	69.6±0.5	40.9±4.3	78.3±1.1
ChebNet-CP	44.94	79.61	41.39±3.05	81.40±1.10	53.32±1.38	85.59±0.37	60.71±1.74	80.44±0.65	47.02±0.73	77.05±0.38	22.41±0.57	70.15±0.07	44.81±1.38	83.01±0.32
Δ	**+8.22**	**+4.16**	+20.69	+10.10	+13.52	+4.59	+7.61	+3.74	+5.02	+1.25	-1.39	+0.55	+3.91	+4.71
ARMANet	36.37	74.77	19.2±3.3	69.5±3.5	40.8±1.0	80.9±0.4	51.5±1.3	75.7±1.4	41.2±0.5	75.6±0.2	23.1±0.4	69.2±0.7	42.4±1.1	77.7±0.6
ARMANet-CP	44.88	79.54	42.30±5.06	81.58±1.34	52.78±1.84	85.41±0.13	59.82±1.63	80.15±1.00	47.31±1.49	77.03±0.48	23.56±1.79	70.16±0.16	43.50±1.49	82.89±0.42
Δ	**+8.51**	**+4.77**	+23.10	+12.08	+11.98	+4.51	+8.32	+4.45	+6.11	+1.43	+0.46	+0.96	+1.10	+5.19
GraphSAGE	37.55	73.57	20.7±2.4	67.6±2.8	41.6±0.5	80.5±0.3	52.6±1.3	74.1±1.2	44.2±0.5	74.4±0.3	23.7±0.4	68.5±0.4	42.5±1.1	76.3±0.1
GraphSAGE-CP	42.99	79.05	38.27±4.18	80.26±0.56	53.16±1.51	84.99±0.20	58.50±2.03	80.14±0.52	45.16±1.88	76.56±0.54	18.96±2.79	69.63±0.20	43.89±1.68	82.72±0.40
Δ	**+5.44**	**+5.48**	+17.57	+12.66	+11.56	+4.49	+5.90	+6.04	+0.96	+2.16	-4.74	+1.13	+1.39	+6.42
TAGCN	39.85	77.40	25.2±1.1	73.5±2.4	49.5±0.7	84.7±0.2	53.3±2.5	77.2±1.2	45.4±0.4	77.0±0.5	23.7±0.6	70.5±0.3	42.0±1.1	81.5±0.2
TAGCN-CP	48.06	82.38	45.83±2.24	85.44±0.59	56.61±0.97	88.67±0.30	62.42±1.68	82.67±0.58	49.11±0.99	79.00±0.35	26.81±2.10	71.91±0.18	47.59±1.20	86.58±0.24
Δ	**+8.21**	**+4.98**	+20.63	+11.94	+7.11	+3.97	+9.12	+5.47	+3.71	+2.00	+3.11	+1.41	+5.59	+5.08
GIN	37.17	75.57	22.8±1.2	72.7±2.6	41.6±0.7	81.8±0.2	54.7±1.4	76.6±1.1	41.3±2.0	75.2±0.3	20.9±1.0	68.0±0.3	41.7±2.1	79.1±0.5
GIN-CP	46.82	77.38	43.48±3.18	80.75±1.45	54.87±1.43	85.03±0.18	57.68±1.59	78.03±0.54	47.44±2.02	73.00±0.55	30.91±1.29	64.24±0.24	46.55±0.91	83.25±0.16
Δ	**+9.65**	**+1.81**	+20.68	+8.05	+13.27	+3.23	+2.98	+1.43	+6.14	-2.20	+10.01	-3.76	+4.85	+4.15
GAT	36.93	73.47	22.6±1.5	68.3±3.0	41.6±0.4	80.9±0.2	55.3±1.3	74.1±1.0	42.1±1.5	73.6±0.3	17.8±0.8	67.3±0.3	42.2±0.5	76.6±0.4
GAT-CP	45.24	76.06	40.07±10.92	78.96±1.70	50.71±1.09	83.11±0.19	55.09±6.56	76.83±0.65	47.07±1.38	71.76±0.14	35.14±0.90	65.29±0.33	43.36±3.32	80.41±0.63
Δ	**+8.31**	**+2.59**	+17.47	+10.66	+9.11	+2.21	-0.21	+2.73	+4.97	-1.84	+17.34	-2.01	+1.16	+3.81
MPNN	44.63	81.32	38.8±2.1	82.4±1.0	46.0±1.6	83.9±0.2	61.4±2.5	81.8±0.7	48.5±1.9	79.4±0.4	28.2±1.7	73.5±0.5	44.9±0.8	86.9±0.4
MPNN-CP	45.36	83.50	39.56±5.44	86.22±0.32	51.72±4.24	87.79±0.08	63.09±1.39	84.03±0.28	47.93±1.54	80.33±0.38	23.34±2.26	73.82±0.27	46.56±3.49	88.83±0.57
Δ	**+0.73**	**+2.18**	+0.76	+3.82	+5.72	+3.89	+1.69	+2.23	-0.57	+0.93	-4.86	+0.32	+1.66	+1.93
CGC	42.47	79.83	34.4±2.7	79.5±1.5	45.0±1.2	81.5±0.2	59.0±2.1	81.1±0.8	48.5±0.5	79.2±0.7	27.3±1.9	72.3±0.1	40.6±1.2	85.4±0.8
CGC-CP	48.55	82.84	42.94±3.34	86.18±0.44	53.55±1.37	87.08±0.21	64.19±1.88	83.68±0.73	48.12±5.22	79.69±0.16	35.68±6.53	72.39±0.24	46.82±3.02	88.00±0.62
Δ	**+6.08**	**+3.01**	+8.54	+6.68	+8.55	+5.58	+5.19	+2.58	-0.38	+0.49	+8.38	+0.09	+6.22	+2.60
Graphformer	45.13	81.32	37.7±3.3	81.0±1.9	48.9±0.3	83.8±0.3	62.9±1.6	82.0±0.7	49.8±0.7	80.0±0.7	28.4±0.7	73.9±0.4	43.1±0.7	87.2±0.4
Graphformer-CP	50.14	83.36	49.18±2.74	85.58±0.80	54.82±1.53	86.93±0.26	66.08±0.66	84.19±0.56	51.76±0.93	80.59±0.36	31.01±1.91	74.04±0.24	47.97±1.75	88.85±0.47
Δ	**+5.01**	**+2.04**	+11.48	+4.58	+5.92	+3.13	+3.18	+2.19	+1.96	+0.59	+2.61	+0.14	+4.87	+1.65
GEN	49.07	80.97	44.9±3.1	81.0±2.4	48.6±6.2	82.7±0.9	63.0±1.1	81.2±0.9	56.5±1.7	79.5±0.1	34.1±6.0	73.7±0.4	47.3±1.4	87.7±0.9
GEN-CP	53.12	83.55	49.43±1.39	85.56±0.61	57.88±1.07	88.05±0.62	66.18±1.81	83.89±0.34	51.47±5.09	80.29±0.08	40.62±6.89	74.28±0.21	53.12±0.78	89.21±0.23
Δ	**+4.05**	**+2.58**	+4.53	+4.56	+9.28	+5.35	+3.18	+2.69	-5.03	+0.79	+6.52	+0.58	+5.82	+1.51
TRAVEL	54.62	82.77	51.9±1.0	84.9±0.9	55.3±0.9	85.9±0.5	65.0±0.4	82.3±0.4	58.0±0.9	80.8±0.7	46.4±0.7	74.5±0.3	51.1±0.9	88.2±0.2
TRAVEL-CP	55.93	84.78	56.84±2.57	88.05±0.41	59.56±1.11	89.02±0.44	67.85±0.97	85.15±0.58	56.63±1.23	81.35±0.24	43.55±4.49	75.59±0.22	51.18±1.56	89.52±0.07
Δ	**+1.31**	**+2.01**	+4.94	+3.15	+4.26	+3.12	+2.85	+2.85	-1.37	+0.55	-2.85	+1.09	+0.08	+1.32

Table 4: State-wise accident occurrence prediction results in terms of F1 score and AUC. Δ denotes the gain obtained by imposing our proposed Concurrency Prior (with the suffix "-CP") in the neural networks.

Dataset	Mean		California		Oregon		Utah		Maryland		Minnesota		Connecticut	
Method	F1	AUC	F1	AUC	F1	AUC	F1	AUC	F1	AUC	F1	AUC	F1	AUC
GCN	28.58	73.85	24.0±0.0	71.5±0.0	20.6±0.5	68.7±0.7	32.7±0.1	76.3±0.3	26.1±1.3	79.5±0.4	28.1±0.4	70.9±0.3	40.0±0.6	76.2±0.8
GCN-CP	42.25	81.05	45.61±1.20	79.24±0.21	45.58±0.83	81.48±0.15	45.54±0.90	83.56±0.25	36.00±1.27	80.90±0.27	41.90±1.47	81.06±0.19	38.86±3.60	80.05±0.34
Δ	**+13.67**	**+7.20**	+21.61	+7.74	+24.98	+12.78	+12.84	+7.26	+9.90	+1.40	+13.80	+10.16	-1.14	+3.85
ChebNet	29.87	75.73	23.2±1.0	72.9±0.2	21.0±0.2	73.1±0.3	34.3±1.2	77.3±0.5	28.5±0.3	80.4±0.1	30.2±2.1	74.1±1.3	42.0±0.4	76.6±0.2
ChebNet-CP	38.74	82.58	38.66±0.69	81.09±0.11	35.68±9.84	82.72±0.10	42.78±1.81	84.46±0.18	35.20±1.35	83.61±0.16	38.88±1.35	82.11±0.11	41.25±1.37	81.49±0.19
Δ	**+8.87**	**+6.85**	+15.46	+8.19	+14.68	+9.62	+8.48	+7.16	+6.70	+3.21	+8.68	+8.01	-0.75	+4.89
ARMANet	29.03	75.53	23.6±2.0	72.8±0.2	18.6±3.4	72.7±0.7	34.6±0.3	77.2±0.3	28.6±1.6	80.6±0.2	26.4±1.7	72.7±1.2	42.4±1.5	77.2±0.6
ARMANet-CP	38.45	82.60	38.48±0.58	81.07±0.05	38.49±1.33	82.71±0.21	44.04±1.83	84.82±0.35	33.54±1.66	83.48±0.12	37.57±1.55	81.96±0.08	38.56±1.19	81.54±0.24
Δ	**+9.42**	**+7.07**	+14.88	+8.27	+19.89	+10.01	+9.44	+7.62	+4.94	+2.88	+11.17	+9.26	-3.84	+4.34
GraphSAGE	30.27	75.17	25.8±0.4	72.8±0.4	21.4±0.9	71.2±1.3	34.3±1.5	77.7±0.5	28.5±1.2	80.2±0.1	28.9±0.1	71.9±0.8	42.7±1.6	77.2±0.6
GraphSAGE-CP	37.67	82.23	39.52±0.51	81.01±0.09	41.53±4.20	82.32±0.12	41.23±5.77	83.93±0.18	30.64±8.28	82.83±0.06	34.91±3.06	81.50±0.23	38.18±1.73	81.78±0.39
Δ	**+7.40**	**+7.06**	+13.72	+8.21	+20.13	+11.12	+6.93	+6.23	+2.14	+2.63	+6.01	+9.60	-4.52	+4.58
TAGCN	30.37	78.10	28.7±0.4	75.9±0.1	24.7±1.0	76.6±0.1	34.1±1.0	78.9±0.4	26.1±0.9	81.8±0.4	30.8±0.9	77.2±0.8	37.8±0.8	78.2±1.3
TAGCN-CP	43.13	87.17	47.25±1.50	85.81±0.15	46.08±1.80	88.67±0.12	46.79±1.34	88.97±0.30	36.78±1.29	86.67±0.28	42.40±2.09	87.47±0.07	39.49±1.53	85.43±0.47
Δ	**+12.76**	**+9.07**	+18.55	+9.91	+21.38	+12.07	+12.69	+10.07	+10.68	+4.87	+11.60	+10.27	+1.69	+7.23
GIN	31.78	76.45	28.0±0.2	72.7±0.2	24.3±0.4	74.2±0.2	36.2±0.3	78.9±0.5	28.2±0.6	80.8±0.2	32.0±1.8	74.9±1.3	42.0±1.2	77.2±0.4
GIN-CP	43.97	82.08	45.42±1.83	79.76±0.05	48.36±2.03	83.07±0.34	46.60±1.21	84.41±0.19	37.38±1.64	82.23±0.18	44.34±1.76	82.35±0.12	41.73±1.87	80.67±0.36
Δ	**+12.19**	**+5.63**	+17.42	+7.06	+24.06	+8.87	+10.40	+5.51	+9.18	+1.43	+12.34	+7.45	-0.27	+3.47
GAT	30.12	74.47	24.5±0.3	72.2±0.1	21.8±0.4	70.2±1.0	35.6±2.2	76.8±0.6	27.8±0.2	80.4±0.2	29.1±0.2	71.3±1.1	41.9±2.6	75.9±1.5
GAT-CP	40.06	80.83	46.70±0.59	79.76±0.21	47.12±0.79	81.26±0.57	39.62±6.36	82.76±1.13	36.07±0.85	81.61±0.75	38.14±9.85	80.81±0.52	32.69±2.74	78.77±0.22
Δ	**+9.94**	**+6.36**	+22.20	+7.56	+25.32	+11.06	+4.02	+5.96	+8.27	+1.21	+9.04	+9.51	-9.21	+2.87
MPNN	41.20	87.13	33.2±1.7	79.9±0.5	56.2±1.3	89.2±0.8	43.1±2.6	88.7±0.3	32.3±0.2	89.4±0.2	38.5±1.0	89.8±0.1	43.9±0.6	85.8±0.7
MPNN-CP	44.93	89.50	43.84±1.93	84.45±0.23	57.06±1.95	90.83±0.17	45.30±4.70	91.59±0.12	34.03±2.50	90.38±0.08	45.99±2.09	91.69±0.07	43.35±2.63	88.07±0.26
Δ	**+3.73**	**+2.37**	+10.64	+4.55	+0.86	+1.63	+2.20	+2.89	+1.73	+0.98	+7.49	+1.89	-0.55	+2.27
CGC	44.80	86.50	34.4±2.8	79.0±0.1	53.1±3.3	88.5±0.1	47.1±0.8	88.3±0.2	43.0±1.9	88.6±0.4	48.1±1.7	88.7±0.5	43.1±0.2	85.9±0.7
CGC-CP	48.18	88.80	46.02±2.08	84.10±0.06	58.00±2.53	90.34±0.16	52.05±3.34	90.89±0.11	41.65±4.13	89.84±0.07	48.01±5.12	90.74±0.17	43.34±3.75	86.89±0.40
Δ	**+3.38**	**+2.30**	+11.62	+5.10	+4.90	+1.84	+4.95	+2.59	-1.35	+1.24	-0.09	+2.04	+0.24	+0.99
Graphformer	39.48	87.15	29.3±0.5	80.1±0.1	53.5±0.4	89.4±0.3	44.8±1.4	89.0±0.1	31.5±0.6	89.2±0.6	33.9±0.5	90.1±0.2	43.9±0.7	85.1±0.6
Graphformer-CP	47.77	88.48	44.68±0.38	84.25±0.11	56.96±1.04	89.71±0.14	53.18±1.72	90.35±0.25	38.97±2.81	89.33±0.14	49.43±1.83	90.13±0.09	43.43±1.37	87.13±0.21
Δ	**+8.29**	**+1.33**	+15.38	+4.15	+3.46	+0.31	+8.38	+1.35	+7.47	+0.13	+15.53	+0.03	-0.47	+2.03
GEN	48.88	84.92	43.9±0.1	77.3±0.4	59.4±0.8	87.2±0.7	53.8±0.8	86.1±0.3	40.8±2.8	88.4±0.8	52.3±1.5	88.7±0.4	43.1±1.1	81.8±0.7
GEN-CP	54.30	89.81	53.30±1.70	86.03±0.10	62.40±0.80	91.28±0.09	56.93±1.14	91.28±0.24	46.44±3.35	90.03±0.19	56.77±2.28	91.71±0.37	49.94±1.11	88.53±0.49
Δ	**+5.42**	**+4.89**	+9.40	+8.73	+3.00	+4.08	+3.13	+5.18	+5.64	+1.63	+4.47	+3.01	+6.84	+6.73
TRAVEL	51.78	88.02	46.1±0.7	81.1±1.0	60.8±0.0	90.5±0.2	55.6±0.6	89.9±1.9	46.4±1.4	90.1±0.3	55.2±1.0	91.3±0.7	46.6±0.3	85.2±0.6
TRAVEL-CP	55.04	90.96	54.85±0.58	87.22±0.18	62.46±1.03	92.24±0.23	57.42±1.84	92.49±0.22	47.44±1.17	90.99±0.16	58.93±0.25	92.87±0.17	49.17±1.62	89.93±0.38
Δ	**+3.26**	**+2.94**	+8.75	+6.12	+1.66	+1.74	+1.82	+2.59	+1.04	+0.89	+3.73	+1.57	+2.57	+4.73

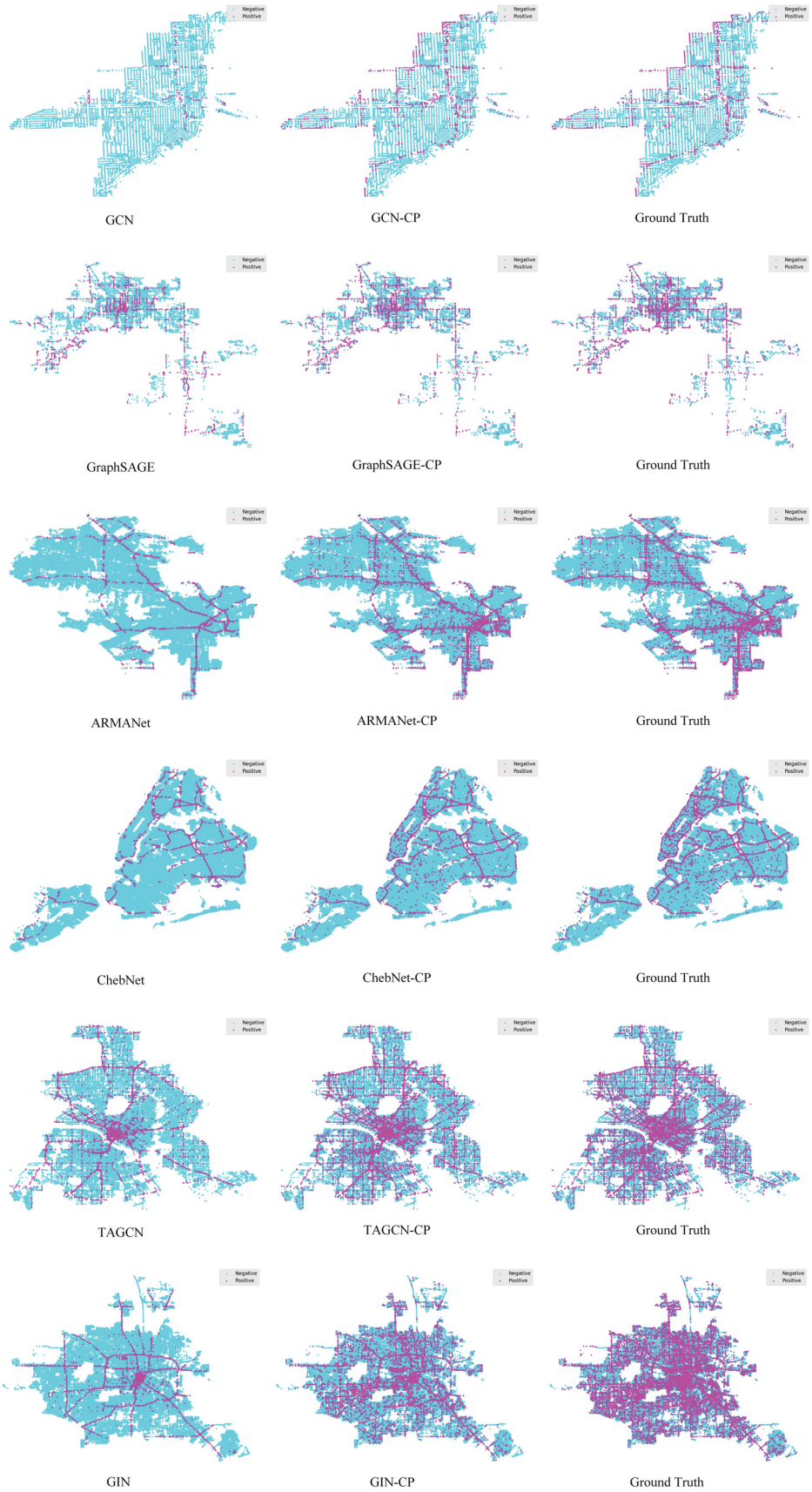

Figure 7: The visualization of the prediction by different methods on cities. Left columns: The original GNN methods. Middle columns: GNN methods with our proposed concurrency prior. Right columns: The ground truth.

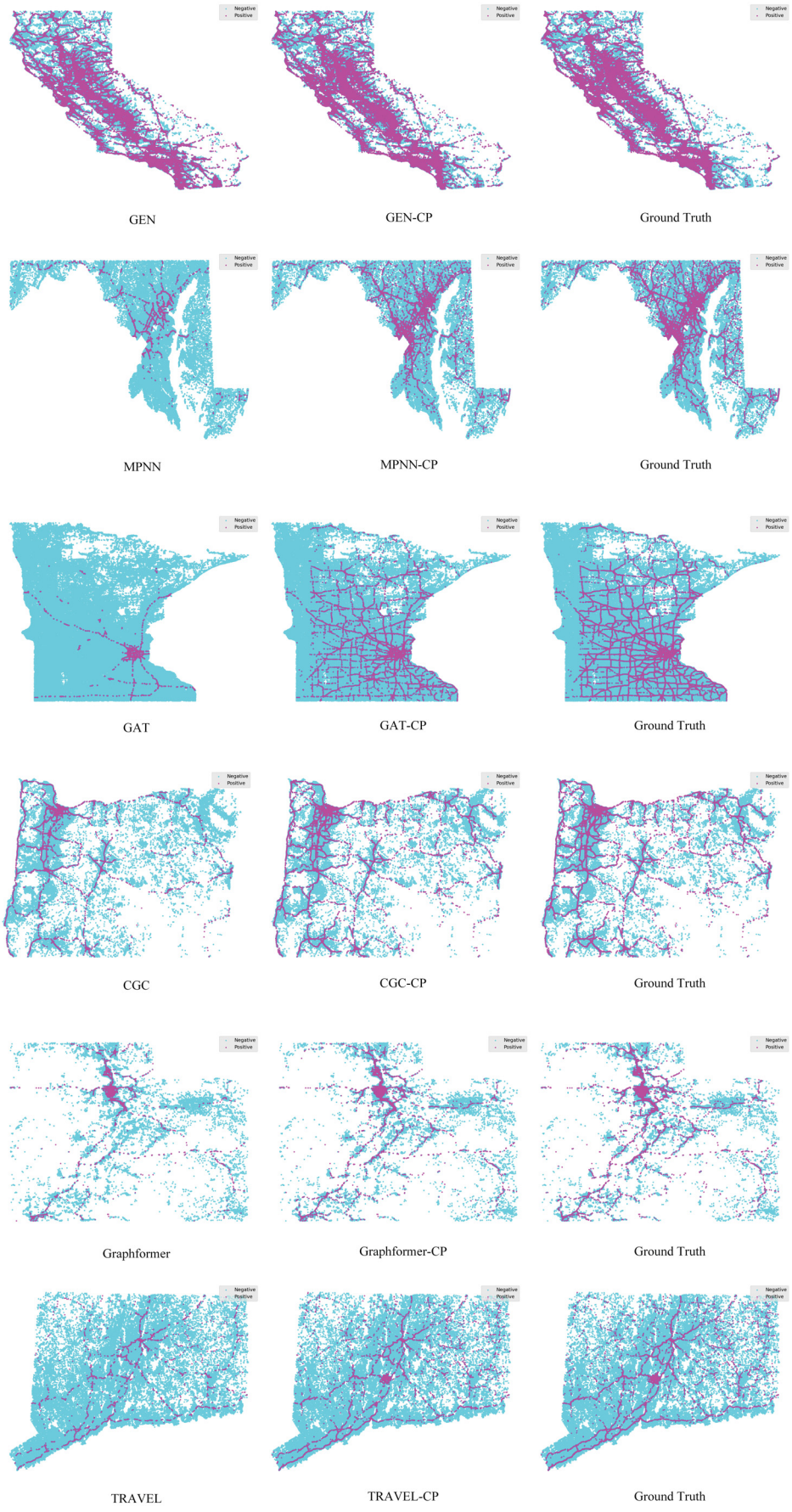

Figure 8: The visualization of the prediction by different methods on states. Left columns: The original GNN methods. Middle columns: GNN methods with our proposed concurrency prior. Right columns: The ground truth.

References

[1] Albert-László Barabási and Réka Albert. 1999. Emergence of scaling in random networks. *science*, 286, 5439, 509–512.

[2] Filippo Maria Bianchi, Daniele Grattarola, Lorenzo Livi, and Cesare Alippi. 2021. Graph neural networks with convolutional arma filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1–1. DOI: 10.1109/tpami.2021.3054830.

[3] Stefano Boccaletti, Ginestra Bianconi, Regino Criado, Charo I Del Genio, Jesús Gómez-Gardenes, Miguel Romance, Irene Sendina-Nadal, Zhen Wang, and Massimiliano Zanin. 2014. The structure and dynamics of multilayer networks. *Physics reports*, 544, 1, 1–122.

[4] Geoff Boeing. 2017. Osmnx: new methods for acquiring, constructing, analyzing, and visualizing complex street networks. *Computers, Environment and Urban Systems*, 65, 126–139.

[5] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. 2013. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*.

[6] Peide Cai, Hengli Wang, Yuxiang Sun, and Ming Liu. 2022. Dq-gat: towards safe and efficient autonomous driving with deep q-learning and graph attention networks. *IEEE Transactions on Intelligent Transportation Systems*, 23, 11, 21102–21112.

[7] Ciro Caliendo, Maurizio Guida, and Alessandra Parisi. 2007. A crash-prediction model for multilane roads. *Accident Analysis & Prevention*, 39, 4, 657–670.

[8] Xiwen Chen, Hao Wang, Abolfazl Razi, Brendan Russo, Jason Pacheco, John Roberts, Jeffrey Wishart, Larry Head, and Alonso Granados Baca. 2022. Network-level safety metrics for overall traffic safety assessment: a case study. *IEEE Access*, 11, 17755–17778.

[9] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in neural information processing systems*, 29, 3844–3852.

[10] Jian Du, Shanghang Zhang, Guanhang Wu, Jose M. F. Moura, and Soummya Kar. 2018. Topology adaptive graph convolutional networks. (2018). arXiv: 1710.10370 [cs.LG].

[11] Mohammed A Fadhel et al. 2024. Comprehensive systematic review of information fusion methods in smart cities and urban environments. *Information Fusion*, 102317.

[12] Zhonghua Gao, Zhenjie Chen, Yongxue Liu, and Kang Huang. 2007. Study on the complex network characteristics of urban road system based on gis. In *Geoinformatics 2007: Geospatial Information Technology and Applications*. Vol. 6754. International Society for Optics and Photonics, 67540N.

[13] Xu Geng, Yaguang Li, Leye Wang, Lingyu Zhang, Qiang Yang, Jieping Ye, and Yan Liu. 2019. Spatiotemporal multi-graph convolution network for ride-hailing demand forecasting. In *Proceedings of the AAAI conference on artificial intelligence* number 01. Vol. 33, 3656–3663.

[14] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. 2017. Neural message passing for quantum chemistry. In *International conference on machine learning*. PMLR, 1263–1272.

[15] Shengnan Guo, Youfang Lin, Ning Feng, Chao Song, and Huaiyu Wan. 2019. Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. In *Proceedings of the AAAI conference on artificial intelligence* number 01. Vol. 33, 922–929.

[16] William L Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 1025–1035.

[17] Mikael Henaff, Joan Bruna, and Yann LeCun. 2015. Deep convolutional networks on graph-structured data. *arXiv preprint arXiv:1506.05163*.

[18] Baixiang Huang, Bryan Hooi, and Kai Shu. 2023. Tap: a comprehensive data repository for traffic accident prediction in road networks. In *Proceedings of the 31st ACM International Conference on Advances in Geographic Information Systems*, 1–4.

[19] Ziheng Huang, Weihan Zhang, Dujuan Wang, and Yunqiang Yin. 2022. A gan framework-based dynamic multi-graph convolutional network for origin–destination-based ride-hailing demand prediction. *Information Sciences*, 601, 129–146.

[20] Weiwei Jiang and Jiayun Luo. 2022. Graph neural network for traffic forecasting: a survey. *Expert Systems with Applications*, 207, 117921.

[21] Weipeng Jing, Wenjun Zhang, Linhui Li, Donglin Di, Guangsheng Chen, and Jian Wang. 2022. Agnet: an attention-based graph network for point cloud classification and segmentation. *Remote Sensing*, 14, 4, 1036.

[22] Jintao Ke, Xiaoran Qin, Hai Yang, Zhengfei Zheng, Zheng Zhu, and Jieping Ye. 2021. Predicting origin-destination ride-sourcing demand with a spatio-temporal encoder-decoder residual multi-graph convolutional network. *Transportation Research Part C: Emerging Technologies*, 122, 102858.

[23] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

[24] Christopher Klinkhamer, Elisabeth Krueger, Xianyuan Zhan, Frank Blumensaat, Satish Ukkusuri, and P Suresh C Rao. 2017. Functionally fractal urban networks: geospatial co-location and homogeneity of infrastructure. *arXiv preprint arXiv:1712.03883*.

[25] Guanyao Li, Xiaofeng Wang, Gunarto Sindoro Njoo, Shuhan Zhong, S-H Gary Chan, Chih-Chieh Hung, and Wen-Chih Peng. 2022. A data-driven spatial-temporal graph neural network for docked bike prediction. In *2022 IEEE 38th International Conference on Data Engineering (ICDE)*. IEEE, 713–726.

[26] Guohao Li, Chenxin Xiong, Ali Thabet, and Bernard Ghanem. 2020. Deepergcn: all you need to train deeper gcns. (2020). arXiv: 2006.07739 [cs.LG].

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

[28] Lei Lin, Zhengbing He, and Srinivas Peeta. 2018. Predicting station-level hourly demand in a large-scale bike-sharing network: a graph convolutional neural network approach. *Transportation Research Part C: Emerging Technologies*, 97, 258–276.

[29] Lingbo Liu, Jingwen Chen, Hefeng Wu, Jiajie Zhen, Guanbin Li, and Liang Lin. 2020. Physical-virtual collaboration modeling for intra-and inter-station metro ridership prediction. *IEEE Transactions on Intelligent Transportation Systems*, 23, 4, 3377–3391.

[30] Xu Liu, Yuxuan Liang, Chao Huang, Yu Zheng, Bryan Hooi, and Roger Zimmermann. 2022. When do contrastive learning signals help spatio-temporal graph forecasting? In *Proceedings of the 30th International Conference on Advances in Geographic Information Systems*, 1–12.

[31] Microsoft Bing. 2021. Bing map traffic api. (2021). Retrieved Dec. 31, 2021 from hhttps://www.bingmapsportal.com.

[32] Moosavi Sobhan. 2022. Us accidents. (2022). https://www.kaggle.com/datasets/sobhanmoosavi/us-accidents.

[33] Alameen Najjar, Shun'ichi Kaneko, and Yoshikazu Miyanaga. 2017. Combining satellite imagery and open data to map road safety. In *Thirty-First AAAI Conference on Artificial Intelligence*.

[34] National Highway Traffic Safety Administration. 2023. Nhtsa report. (2023). https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/813514.

[35] Jutaek Oh, Simon P Washington, and Doohee Nam. 2006. Accident prediction model for railway-highway interfaces. *Accident Analysis & Prevention*, 38, 2, 346–356.

[36] Yulong Pei and Lin Hou. 2024. Safety assessment and risk management of urban arterial traffic flow based on artificial driving and intelligent network connection: an overview. *Archives of Computational Methods in Engineering*, 1–19.

[37] Bhagwant Persaud and Leszek Dzbik. 1992. Accident prediction models for freeways. *Transportation Research Record*, 1401, 55–60.

[38] Simon J.D. Prince. 2023. *Understanding Deep Learning*. The MIT Press. http://udlbook.com.

[39] Saeed Rahmani, Asiye Baghbani, Nizar Bouguila, and Zachary Patterson. 2023. Graph neural networks for intelligent transportation systems: a survey. *IEEE Transactions on Intelligent Transportation Systems*.

[40] Abolfazl Razi, Xiwen Chen, Huayu Li, Hao Wang, Brendan Russo, Yan Chen, and Hongbin Yu. 2023. Deep learning serves traffic safety analysis: a forward-looking review. *IET Intelligent Transport Systems*, 17, 1, 22–71.

[41] Ahmad Sarlak, Abolfazl Razi, Xiwen Chen, and Rahul Amin. 2023. Diversity maximized scheduling in roadside units for traffic monitoring applications. In *2023 IEEE 48th Conference on Local Computer Networks (LCN)*. IEEE, 1–4.

[42] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. 2008. The graph neural network model. *IEEE transactions on neural networks*, 20, 1, 61–80.

[43] Weijing Shi and Raj Rajkumar. 2020. Point-gnn: graph neural network for 3d object detection in a point cloud. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 1711–1719.

[44] Yunsheng Shi, Zhengjie Huang, Shikun Feng, Hui Zhong, Wenjin Wang, and Yu Sun. 2021. Masked label prediction: unified message passing model for semi-supervised classification. (2021). arXiv: 2009.03509 [cs.LG].

[45] Yuyol Shin and Yoonjin Yoon. 2020. Incorporating dynamicity of transportation network with multi-weight traffic graph convolutional network for traffic forecasting. *IEEE Transactions on Intelligent Transportation Systems*, 23, 3, 2082–2092.

[46] Luqi Tang, Fuwu Yan, Bin Zou, Wenbo Li, Chen Lv, and Kewei Wang. 2023. Trajectory prediction for autonomous driving based on multiscale spatial-temporal graph. *IET Intelligent Transport Systems*, 17, 2, 386–399.

[47] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *International Conference on Learning Representations*. https://openreview.net/forum?id=rJXMpikCZ.

[48] Duncan J Watts and Steven H Strogatz. 1998. Collective dynamics of 'small-world'networks. *nature*, 393, 6684, 440–442.

[49] Boris Weisfeiler and Andrei Leman. 1968. The reduction of a graph to canonical form and the algebra which appears therein. *NTI, Series*, 2, 9, 12–16.

[50] Jianjun Wu, Ziyou Gao, Huijun Sun, and Haijun Huang. 2004. Urban transit system as a scale-free network. *Modern Physics Letters B*, 18, 19n20, 1043–1049.

[51] Tian Xie and Jeffrey C. Grossman. 2018. Crystal graph convolutional neural networks for an accurate and interpretable prediction of material properties.

Phys. Rev. Lett., 120, (Apr. 2018), 145301, 14, (Apr. 2018). DOI: 10.1103/PhysRevLett.120.145301.

[52] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2018. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*.

[53] Jiexia Ye, Juanjuan Zhao, Kejiang Ye, and Chengzhong Xu. 2020. How to build a graph-based deep learning architecture in traffic domain: a survey. *IEEE Transactions on Intelligent Transportation Systems*, 23, 5, 3904–3924.

[54] Xueyan Yin, Genze Wu, Jinze Wei, Yanming Shen, Heng Qi, and Baocai Yin. 2021. Deep learning on traffic prediction: methods, analysis, and future directions. *IEEE Transactions on Intelligent Transportation Systems*, 23, 6, 4927–4943.

[55] Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. 2020. Graph contrastive learning with augmentations. *Advances in neural information processing systems*, 33, 5812–5823.

[56] Le Yu, Bowen Du, Xiao Hu, Leilei Sun, Liangzhe Han, and Weifeng Lv. 2021. Deep spatio-temporal graph convolutional network for traffic accident prediction. *Neurocomputing*, 423, 135–147.

[57] Bohang Zhang, Jingchu Gai, Yiheng Du, Qiwei Ye, Di He, and Liwei Wang. 2024. Beyond weisfeiler-lehman: a quantitative framework for GNN expressiveness. In *The Twelfth International Conference on Learning Representations*. https://openreview.net/forum?id=HSKaGOi7Ar.

[58] Xinan Zhang, Yung-An Hsieh, Pingzhou Yu, Zhongyu Yang, and Yichang James Tsai. 2023. Multiclass transportation safety hardware asset detection and segmentation based on mask-rcnn with roi attention and ioma-merging. *Journal of Computing in Civil Engineering*, 37, 5, 04023024.

[59] Ling Zhao, Yujiao Song, Chao Zhang, Yu Liu, Pu Wang, Tao Lin, Min Deng, and Haifeng Li. 2019. T-gcn: a temporal graph convolutional network for traffic prediction. *IEEE transactions on intelligent transportation systems*, 21, 9, 3848–3858.

[60] Lai Zheng, Tarek Sayed, and Fred Mannering. 2021. Modeling traffic conflicts for use in road safety analysis: a review of analytic methods and future directions. *Analytic methods in accident research*, 29, 100142.

[61] Zhengyang Zhou, Yang Wang, Xike Xie, Lianliang Chen, and Hengchang Liu. 2020. Riskoracle: a minute-level citywide traffic accident forecasting framework. In *Proceedings of the AAAI Conference on Artificial Intelligence* number 01. Vol. 34, 1258–1265.

[62] Zhengyang Zhou, Yang Wang, Xike Xie, Lianliang Chen, and Chaochao Zhu. 2020. Foresee urban sparse traffic accidents: a spatiotemporal multi-granularity perspective. *IEEE Transactions on Knowledge and Data Engineering*, 34, 8, 3786–3799.