Incident congestion propagation prediction using incident reports

Mathias N. Tygesen
mnity@dtu.dk
Technical University of Denmark
Kgs. Lyngby, Denmark

Mayank Sharan
mayanksharan96@gmail.com
University of California San Diego
La Jolla, CA, USA

Francisco C. Pereira
Technical University of Denmark
Kgs. Lyngby, Denmark

Filipe Rodrigues
Technical University of Denmark
Kgs. Lyngby, Denmark

Rose Yu
University of California San Diego
La Jolla, CA, USA

ABSTRACT

Accurate predictions of how congestion propagates are essential for mitigating its effects on traffic and the urban environment. However, the vast majority of state-of-the-art traffic prediction models focus on regular traffic scenarios and struggle to adapt to the conditions following incidents. This is particularly problematic since the irregular periods after incidents are arguably when traffic predictions are most critical. Current traffic models struggle with non-recurring congestion for two reasons: they lack inputs alerting them an incident has happened, and traffic data containing incident information is scarce. We create two new such datasets: one by simulating incidents and their congestion in an open-source microscopic simulator and another by fusing real-world traffic flow data with incident reports. We then propose a framework that integrates incident reports into deep learning models for congestion propagation prediction. Our framework utilizes the recent traffic flow data and fuses it with information from incident reports. We perform a detailed empirical comparison between recurrent and graph-based models utilizing incident reports against baselines. Our study demonstrates that our framework significantly outperforms state-of-the-art graph-based models that do not account for incident reports.

CCS CONCEPTS

• **Information systems → Spatial-temporal systems**; • **Computing methodologies → Neural networks**.

KEYWORDS

Congestion Propagation Prediction, Graph Neural Networks

ACM Reference Format:
Mathias N. Tygesen, Mayank Sharan, Francisco C. Pereira, Filipe Rodrigues, and Rose Yu. 2023. Incident congestion propagation prediction using incident reports. In *1st ACM SIGSPATIAL International Workshop on Sustainable Mobility (SuMob '23), November 13, 2023, Hamburg, Germany.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3615899.3627934

1 INTRODUCTION

Accurately predicting how traffic evolves is vital if we want to make sure that our cities are sustainable, unpolluted, and pleasant for travelers. These predictions are especially important during congestions as these have a huge impact on the travel experience and the urban environment. In fact, a daily 10 km car commute through the London city center would yearly amount to 139 hours of lost time, 282 kgs of CO_2 emitted, and 200 GBP spent on fuel, just from congestion alone [24]. As cities around the world continue to grow in size and population density [18], the frequency and severity of congestions will only increase. While personal vehicles are responsible for a majority of traffic [25], other uses of the road network such as public transport and freight are also affected. As such mitigating the impact of congestion benefits not only car owners but all travelers. Many drivers now rely on route planners for accurate travel time predictions and congestion avoidance, while traffic managers rely on the predictions to make sure that traffic runs smoothly. This is made possible by the significant improvements that modern deep learning methods have made in predicting traffic flow during normal operations [12, 23]. These models excel at capturing the spatial and recurring temporal patterns that govern traffic flow. However, up to 50% of traffic congestion comes from non-recurring incidents [17, c13-p28], and these congestions have a huge impact on the traffic state. Since these impacts change the traffic state suddenly, it is crucial that intelligent transport systems react fast. However, although traffic flow prediction is a very active research field with recent research focusing on, e.g., utilizing auxiliary data [13, 19], recent advances in deep learning [4**?**], or newly available trajectory data [8], none of these studies considers how incidents affect the traffic or how the models perform during non-recurring congestion.

Since incidents happen for a multitude of reasons, it is unreasonable to expect models to be able to predict the time and place of incidents. Hence, for a machine learning model to predict the effects of non-recurrent events like incidents, it first has to infer that an incident has happened by looking at traffic flow data. This leads to an unavoidable delay before the model can update its predictions. One way to help models better deal with incidents is by giving them more information through incident reports. In California, *Sigalerts* has been used since the 1950s [2], and more recently, *Waze* has used smartphones to bring incident reports to their users [1]. Given a sufficiently high user penetration rate and as V2X communication technologies evolve, we can assume that we can get an incident report before the next prediction is needed. A key reason for the lack of research using incident reports is that a large-scale dataset of incident reports combined with traffic flow data does not exist. We remedy this by presenting two new datasets. First, we turn to the research on microscopic simulation of traffic flow. Microscopic traffic simulation has been a central topic in traffic research for a long time [14]. It can approximate the aggregate traffic flow features that a real-world induction loop traffic volume sensor gathers by simulating how individual vehicles drive on a road network. Nevertheless, limited research has been focused on simulating incidents on traffic networks. Therefore, we extend the well-known open-source simulation tool SUMO with incident simulation capabilities, thus allowing us to create a unique large-scale dataset of incidents and subsequent traffic congestion that can foster traffic incident research. Secondly, we fuse open-source traffic flow data from PEMS[1] with the recent dataset of incident reports from [16][2] to create a real-world dataset of incidents and their associated effects on congestion. The immediate impact of an incident can be quantified by analyzing its Spatio-temporal Impact Region (SIR) [3, 21]. By predicting the SIR immediately after an incident, traffic planners are better able to mitigate the negative effects of the congestion, and route planners can select routes to avoid the congestion.

Apart from the datasets, we also propose a novel deep learning framework for predicting the SIR after incidents. Our framework utilizes the recent traffic flow and takes real-time incident reports into account when predicting the spatio-temporal impact on traffic congestion. We explore different recurrent and graph neural network architectures in the framework. By comparing different models on the abovementioned datasets, we empirically demonstrate that adding incident reports and using more spatially-aware models can dramatically improve congestion propagation predictions. All code and data are made available online[3].

In summary, our contributions are:

- We formulate the problem of traffic congestion propagation prediction as a joint classification and regression task.
- We create a unique large-scale dataset for traffic incident research by extending the SUMO microscopic simulator to simulate incidents.
- We present a real-world dataset by combining PEMS traffic flow data with the LSTW dataset of incident reports.
- We propose a deep learning framework using recurrent and graph neural network-based spatio-temporal models with incident information for congestion propagation predictions.
- We empirically demonstrate the significant positive effect of adding incident information for congestion propagation prediction, and we investigate the impact of different neural architecture design choices in the quality of the predictions.

2 RELATED WORK

Deep learning is currently the state-of-the-art method for traffic prediction during free-flow traffic or recurring congestion. As stated in the survey paper [9], many different deep learning methods have been used, with recent research focusing on applying graph neural networks (GNN) to capture spatial dependencies. In [12], the authors present the GNN-based Diffusion Convolutional Recurrent Neural Network (DCRNN) model that uses bi-directional random walks to capture the spatial dependencies in the graph. The authors also use modified Gated Recurrent Units (GRU) and an encoder-decoder structure to capture the temporal dependencies. The ASTGCN model presented in [6] instead uses attention layers to model the spatio-temporal dependencies. In [30], the authors also use attention layers, but they do so in the frequency domain. Another recent thread of research has been on inferring an adjacency matrix for the graph neural network instead of simply relying on the road network as an adjacency matrix. For example, the Graph WaveNet proposed in [28] generates an adjacency matrix using node embeddings that can be learned from data end-to-end while [29] learns a collection of higher-order spatial and temporal graphs to model them using higher-order GNNs. In [26], the authors use a Neural Relational Inference (NRI) [10] structure to infer a distribution of adjacency matrices from data during training. The sampled adjacency matrix is then used in a Message-Passing Neural Network with GRU cells to predict the traffic flow.

However, none of these papers focuses on how well the model predicts non-recurring congestion following traffic incidents. In this paper, we focus precisely on predicting the Spatio-temporal Impacted Region (SIR) of incidents. Inference of SIR from observed data is also an active research area. However, most research on impact regions is often limited to small datasets with only a few incident observations and on a small road network. Hence, the models are usually limited to statistical models like in [3], where the SIR is inferred by comparing traffic speed to the historical average speeds, or in [15] where they use k-means clustering. The authors [27] use a Graph Neural Network and have a larger dataset, but they predict whether an incident leads to *any* congestion instead of predicting the full SIR.

Due to the lack of large-scale datasets, researchers have used microscopic simulation to create larger synthetic data sets. In [20], the authors use a microscopic simulator of a highway corridor to create a dataset of free-flow and incident scenarios for adaptation to incident scenarios. In [22], the authors also use a microscopic simulator to simulate non-recurring incidents and measure the average impact on agents' travel time. The authors of [5] also simulate incidents and use the simulated flow and incident data to analyze how including incident information improves the traffic flow predictions in sequence-to-sequence models. They analyze how the congestion spreads to the incident links immediate neighbors but do not predict the full SIR. Similar to these papers, we use microscopic simulation to generate a data set of traffic flows following incidents. However, to the authors' knowledge, this is the first paper to use the simulated data to predict the SIR. Furthermore, we predict the full SIR immediately following the incident using deep learning models, whereas previous works either focus on improving short-term traffic flow predictions or the aggregate impact on the entire traffic system. Lastly, unlike previous works, we compare the same methods on both simulated and real-world data, thus allowing for novel insights on the discrepancies between those data sources.

[1]https://pems.dot.ca.gov/
[2]https://smoosavi.org/datasets/lstw
[3]https://github.com/MathiasNT/CongestionPropagationPrediction

3 METHODOLOGY

3.1 Traffic Congestion Prediction.

The task of traffic incident propagation prediction is to predict how the congestion following a non-recurring incident propagates through the network. We refer to the area that was impacted during the full run of congestion as the Spatio-temporal Impact Region (SIR) [3]. Following the formulation in [21], the problem is finding a function $\mathcal{F}$ that takes in historical traffic data $\boldsymbol{X}_{\text{traffic}}$ from the last q timesteps, incident information data $\boldsymbol{I}$ and network information data $\boldsymbol{N}_{\text{info}}$, and outputs the predicted propagation of the congestion $\boldsymbol{Y}$. We define the propagation of the congestion $\boldsymbol{Y}$ as an $E \times 4$ matrix, where E is the number of traffic sensors:

$$Y = [\boldsymbol{L}_{\text{class}}, \boldsymbol{y}_{\text{start}}, \boldsymbol{y}_{\text{end}}, \boldsymbol{y}_{\Delta\text{v}}], \tag{1}$$

where $\boldsymbol{L}_{\text{class}} \in \{0,1\}^E$ is a Boolean indicating whether or not the link is affected by the incident at any point, $\boldsymbol{y}_{\text{start}} \in \mathbb{R}^E$ and $\boldsymbol{y}_{\text{end}} \in \mathbb{R}^E$ are the start and end time of the effect for each link and $\boldsymbol{y}_{\Delta\text{v}} \in \mathbb{R}^E$ is the change of traffic speed for each link. The SIR is visualized in Figure 1 for a road network with three sensors, where Sensor 1 and Sensor 2 are congested. The incident propagation prediction problem is hence finding $\mathcal{F}$ such that

$$[\boldsymbol{X}_{\text{traffic}}{}^{(t_i - q\,:\,t_i)}, \boldsymbol{I}, \boldsymbol{N}_{\text{info}}] \xrightarrow{\mathcal{F}} Y, \tag{2}$$

where $\boldsymbol{X}_{\text{traffic}} \in \mathbb{R}^{E \times T \times (F_{\text{traffic}} + F_{\text{time}})}$, $\boldsymbol{I} \in \mathbb{R}^{F_{\text{info}}}$, $\boldsymbol{N}_{\text{info}} \in \boldsymbol{R}^{E \times F_{\text{net}}}$. T is the number of timesteps before the incident. F_{traffic} is the traffic flow features from the sensors, such as flow, average occupancy, and average speed. F_{time} is temporal features such as time of day, F_{info} is the incident report features such as location and severity, and F_{net} is the network features, specifying information about the road network if available. Likewise, an uninformed model without incident information can be formulated as follows:

$$[\boldsymbol{X}_{\text{traffic}}{}^{(t_i - q\,:\,t_i)}] \xrightarrow{\mathcal{F}} Y. \tag{3}$$

The prediction problem in (2) is challenging because the model both has to be able to infer the current state of the traffic network from $\boldsymbol{X}_{\text{traffic}}$ and fuse it with the incident information in $\boldsymbol{I}$ and network information in $\boldsymbol{N}_{\text{info}}$. Since traffic has complex spatial and temporal dependencies, this requires a model that can model both. Furthermore, the congestion following an incident only affects a localized upstream area of the incident. Hence, the model has to be able to take the structure of the road network into account to be able to predict this area precisely.

3.2 SIR prediction using Graph Recurrent Neural Networks

Due to the complex spatio-temporal dependencies in traffic data and the importance of utilizing the road structure in the predictions, we propose the model framework in Figure 2. The framework uses a spatio-temporal GNN that acts on the sensor locations as nodes and embeds the historical traffic flow data $\boldsymbol{X}_{\text{traffic}}$ to node embeddings $\boldsymbol{H}_X \in \mathbb{R}^{E \times F_{l_1}}$ with F_{l_1} features. The incident report is padded with zeros for all non-incident sensors and merged with the network information using a Multi-Layer Perceptron (MLP) to create a node embedding $\boldsymbol{H}_I \in \mathbb{R}^{E \times F_{l_2}}$. The node embeddings $\boldsymbol{H}_I$ and $\boldsymbol{H}_X$ are then fused using an MLP. The fused node embedding is passed through a GNN to propagate the information through the network.

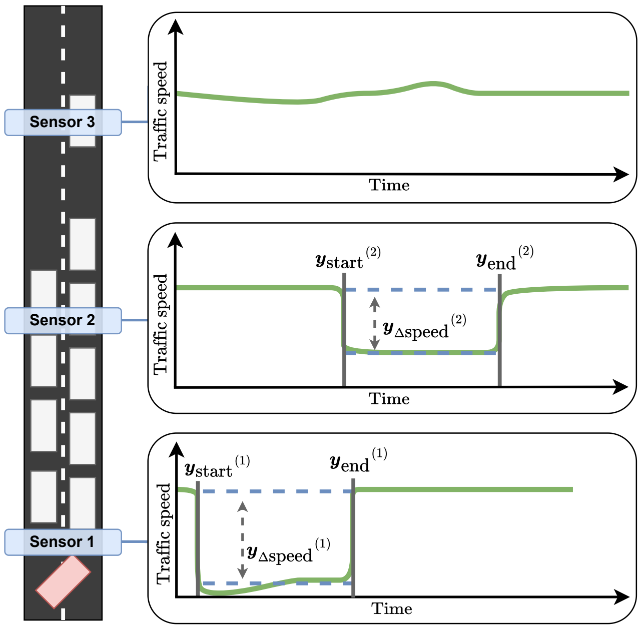


Figure 1: Visualization of the spatio-temporal impact region. The left-hand side shows a road with three sensors and a crashed vehicle at the bottom. The right-hand side shows the time series of measured traffic speeds by the sensors. Sensor 1 and sensor 2 are congested, while sensor 3 is not. For the congested sensors the congestion start time, end time and change in speed is depicted in the diagram.

The resulting embedding $\boldsymbol{H}_s \in \mathbb{R}^{E \times F_{l_3}}$ is then passed to four fully connected layers to create the predictions $\boldsymbol{L}_{\text{class}}, \boldsymbol{y}_{\text{start}}, \boldsymbol{y}_{\text{end}}$ and $\boldsymbol{y}_{\Delta\text{v}}$. The choice of GNN layers, shown in green in Figure 2, is flexible. Here we use DCRNN [12] and NRI [10] layers. For purely recurrent models, the same framework can be utilized on a per-node basis, and for uninformed models, the incident embedding can be removed.

3.2.1 DCRNN. Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, A)$ with node set $\mathcal{V}$, edge set $\mathcal{E}$ and a directed weighted adjacency matrix $A \in \mathbb{R}^{E \times E}$, the diffusion convolution of a graph signal with P features $\boldsymbol{X} \in \mathbb{R}^{E \times P}$ can be defined for the pth feature as

$$\boldsymbol{X}_{:,p} \star_{\mathcal{G}} f_\theta = \sum_{k=0}^{K-1} \left(\theta_{k,1} (\boldsymbol{D}_O^{-1} \boldsymbol{A})^k + \theta_{k,2} (\boldsymbol{D}_I^{-1} \boldsymbol{A}^\top)^k \right) \boldsymbol{X}_{:,p}, \tag{4}$$

where $\boldsymbol{\theta} \in \mathbb{R}^{K \times 2}$ are the parameters for the filter f_θ, $\boldsymbol{D}_O$ is the out-degree matrix, $\boldsymbol{D}_I$ is the in-degree matrix and K is the amount of steps in the diffusion process. The diffusion convolution layer going from P dimensions to Q dimensions is then:

$$H_{:q} = \boldsymbol{a} \left(\sum_{p=1}^{P} \boldsymbol{X}_{:,p} \star_{\mathcal{G}} f_{\Theta_{q,p,:,:}} \right) \quad \text{for } q \in \{1, \dots Q\}, \tag{5}$$

where $\boldsymbol{\Theta} \in \mathbb{R}^{Q \times P \times K \times 2}$ is the collection of all convolution filters and $\boldsymbol{a}$ is an activation function.

In order to extend the DCRNN layer to capture temporal dynamics, the layer utilizes Gated Recurrent Units (GRU). The layer

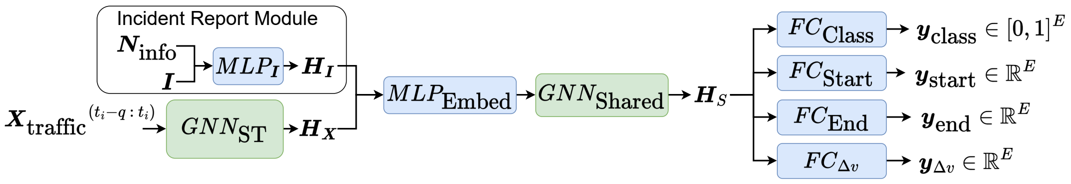


Figure 2: The proposed model for SIR prediction using GNNs. The GNN layers, depicted with green, can be interchanged with any GNN layer. For uninformed models, the Incident Report Module can be removed.

substitutes the matrix multiplications in the GRU with the diffusion convolution layer in Equation (5).

3.2.2 NRI. The NRI model uses an encoder to predict a latent graph structure and then a decoder to produce node embeddings. Both the encoder and the decoder use Message Passing Neural Network (MPNN) layers. The MPNN layer creates messages based on the node features, passes the message along the directed edges, and then updates the node embeddings based on the aggregated messages. The message going from node i to node j can be calculated as

$$\tilde{\boldsymbol{h}}_{(i,j)} = f_e([\boldsymbol{h}_i, \boldsymbol{h}_j, \boldsymbol{h}_{(i,j)}]. \tag{6}$$

The message can also be seen as an edge embedding of the directed edge from node i to j. $\boldsymbol{h}_i$ and $\boldsymbol{h}_j$ are the node embeddings for node i and j respectively and $\boldsymbol{h}_{(i,j)}$ is the previous edge embedding. The aggregation of messages and update of the node embedding is then

$$\tilde{\boldsymbol{h}}_j = f_v\left(\left[\sum_{i \in \mathcal{N}_j} \mathbf{h}_{(i,j)}, \mathbf{h}_j\right]\right), \tag{7}$$

where $\tilde{\boldsymbol{h}}_j$ is the updated node embedding. $\mathcal{N}_j$ denotes the set of nodes with an edge going to node j. f_e and f_v are small MLPs. The encoder takes the node features $\boldsymbol{X}$ and uses a fully connected graph to create edge embeddings for all possible directed edges:

$$\begin{aligned} \mathbf{h}_j^1 &= f_{\text{emb}}(\mathbf{x}_j) \\ \mathbf{h}_{(i,j)}^1 &= f_e^1\left(\left[\mathbf{h}_i^1, \mathbf{h}_j^1\right]\right) \\ \mathbf{h}_j^2 &= f_v^1\left(\sum_{i \in \mathcal{N}_j} \mathbf{h}_{(i,j)}^1\right) \\ \mathbf{h}_{(i,j)}^2 &= f_e^2\left(\left[\mathbf{h}_i^2, \mathbf{h}_j^2, \mathbf{h}_{(i,j)}^1\right]\right). \end{aligned} \tag{8}$$

We then define the edge posterior as $q_\phi(z_{ij}|\boldsymbol{X}) = \text{softmax}(\boldsymbol{h}_{(i,j)}^2)$. This posterior graph distribution is then sampled from using the Gumbel distribution:

$$\mathbf{z}_{ij} = \text{softmax}\left(\left(\mathbf{h}_{(i,j)}^2 + \mathbf{g}\right)/\tau\right), \tag{9}$$

where $\mathbf{g}$ is a vector of i.i.d samples from a Gumbel(0,1) distribution and τ is a parameter to control the smoothing of the samples. The resulting samples then denote an adjacency matrix $\boldsymbol{A}$ s.t. $\boldsymbol{A}_{ij} = z_{ij}$. The decoder can then create node embeddings using the sampled adjacency matrix. We include GRU cells in the decoder to capture temporal dynamics. The decoder for timestep t is then:

$$\begin{aligned} \tilde{\mathbf{h}}_{(i,j)}^t &= z_{ij,k}\tilde{f}_e\left(\left[\tilde{\mathbf{h}}_i^t, \tilde{\mathbf{h}}_j^t\right]\right) \\ \text{MSG}_j^t &= \sum_{i \neq j} \tilde{\mathbf{h}}_{(i,j)}^t \\ \tilde{\mathbf{h}}_j^{t+1} &= \text{GRU}\left(\left[\text{MSG}_j^t, \mathbf{x}_j^t\right], \tilde{\mathbf{h}}_j^t\right) \\ \boldsymbol{h}_j^{t+2} &= \mathbf{x}_j^t + f_{\text{out}}\left(\tilde{\mathbf{h}}_j^{t+1}\right). \end{aligned} \tag{10}$$

where $\tilde{\boldsymbol{h}}_i^t$ is the node embedding from the previous timestep and $\boldsymbol{x}_j^t$ is the current node features.

The encoder and decoder can then be trained jointly. For details in the training we refer to the original paper [10].

3.3 SIR inference from observed traffic flows.

We use an approach like in [3] to infer the SIR from the observed traffic flows. First, we process the data such that all sensors have the same number of lanes. In case of multiple lanes per traffic sensor, all unused lanes are masked and padding lanes are added. Then, a unique congestion threshold is calculated based on the average speed of non-incident traffic for each traffic sensor, considering distinct weekdays and hours of the day. Let $\tilde{s}_{i,d,h}$ be the average speed of all non-incident traffic at sensor i on weekday d during hour h and $\delta_{i,d,h}$ the corresponding standard deviation. The congestion threshold is then defined as

$$T_{i,d,h} = \tilde{s}_{i,d,h} - \alpha\delta_{i,d,h} \tag{11}$$

where α is a hyperparameter. Following an incident, a sensor at a timestep is defined as affected if its speed is below the relevant congestion threshold. Data from loop detectors can be prone to noise if not aggregated over an extended period. Therefore, all affected observations not part of a sequence of affected observations longer than η observations, where η is a hyperparameter, are filtered out. The hyperparameters α and η are optimized to identify affected sensors deviating significantly from their historical average speed while minimizing the deviation for unaffected sensors. Lastly, all downstream affected links are filtered out as the SIR considers the immediate congestion propagation directly following the incident. Then $\boldsymbol{L}_{class}$ is created by defining any road link with a sequence of affected observations on any lane as affected. $\boldsymbol{y}_{start}$ and $\boldsymbol{y}_{end}$ are created by finding the timestep of the first and last affected observation on the road links, and $\boldsymbol{y}_{\Delta v}$ by calculating the mean difference between the observed speed on the affected link and the historical average speed on the link between $\boldsymbol{y}_{start}$ and $\boldsymbol{y}_{end}$.

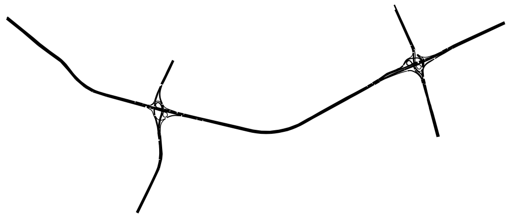


Figure 3: The road network in the Irish Motorway SUMO scenario.

4 DATASETS

4.1 Simulated Irish Motorway

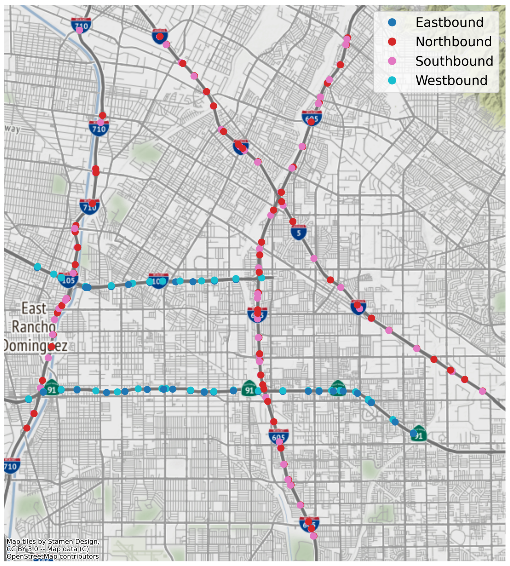
Currently, no large-scale datasets of incident reports and the corresponding traffic congestion exist. However, microscopic simulators like SUMO [14] have been used to simulate traffic flow on road networks. Unfortunately, SUMO does not have the capability to simulate incidents. Therefore, we extend the SUMO simulator to create incidents by blocking road lanes and simulate the subsequent congested traffic flows. Instead of creating a new simulation scenario, we use the already calibrated Irish Motorway scenario from [7]. As seen in Figure 3 the scenario contains a stretch of highway and two intersections with ramps on and off the highway. The simulated road links have between 1 and 6 lanes. We add simulated loop detectors on all lanes at the middle of each road link totaling 147 sensors, and aggregate the observations in 30-second intervals. Using the scenario, we run 10,000 simulations of the traffic flow with and without incidents. We use the previous 10 timesteps to create X_{traffic}. The traffic features F_{traffic} are the average occupancy of the sensor, the average speed, and the total flow in the 30-second interval for each lane. The temporal features consist of the trigonometric transformation of the time of day in seconds.

For incident reports, we use the index of the closest upstream sensor, the number of blocked lanes, how fast cars are able to drive past the incident, and the time until the incident is cleared. The network features are indicator variables denoting the number of lanes at each sensor location. We infer the SIR using the method explained in 3.3. However, instead of creating a confidence interval around the historical average of the observed speed, we instead create it based on the counterfactual simulation without the incident. This way, we utilize the simulation to get the best possible SIR. We select an α of 1.95 and an η of 10. After simulation and inference, we split the data into train, validation, and test sets using a 60/20/20 ratio. As the incidents are simulated independently, we do the split randomly. We retain some of the sensors for the test set to have a spatially out-of-distribution sub-set of the test set. This is done to test the models' ability to generalize to unseen incident locations.

4.2 LSTW PEMS (Real World)

To create a real-world data set of incidents and corresponding traffic flow, we combine traffic flow data and incident reports from Los Angeles, California. The traffic flow data is created by CalTrans and is available from the PEMS system [4]. We select five stretches of intersecting highways with 215 sensors in the northern part of LA and get the traffic data from the year 2017. The selected sensors can be seen in Figure 4. The observations are aggregated in five-minute intervals, and we use the last 12 timesteps for X_{traffic}. The traffic features F_{traffic} are the average occupancy of the sensor, the average speed, and the total flow in the five-minute interval. The temporal features are the day of the week and the time of day in seconds, resulting in four temporal features after trigonometric transformation.

[4]http://pems.dot.ca.gov/



● Eastbound
● Northbound
● Southbound
● Westbound
East Rancho Dominguez
Map tiles by Stamen Design, CC BY 3.0 -- Map data (C) OpenStreetMap contributors

Figure 4: The location of the selected sensors in the PEMS dataset. The colors indicate the direction of the road on which the sensors are placed.

The incident data is from [16]. We extract the incidents on the same five stretches from which we have traffic flow data. We then match each incident with the corresponding time slice of the traffic flow data and find the closes upstream sensor of the incident. We create incident reports containing the index of the closest sensor, a severity score between zero and four, a Traffic Message Channel [5] code with a more detailed description of the event and the reported duration of the block. With this we can then run inference of the SIR using the Frame Region method explained in 3.3 using an α of one and an η of 2. After this, we end up with 1024 data points of incidents with corresponding traffic flow and SIR. We again split the data into train, validation, and test sets using a 60/20/20 ratio. As the incidents are not overlapping, we do this split randomly.

5 EXPERIMENTS

In this section, we conduct a series of experiments analyzing the effect of incident information, the size of the dataset, as well as time-lagged traffic on congestion prediction.

[5]https://wiki.openstreetmap.org/wiki/TMC/Event_Code_List

Table 1: Classification performance of the different uninformed models on the full test set of the simulated data. The mean of four runs is reported along with one standard deviation.

Model	Accuracy	Precision	Recall	F1
MLP	0.640 ± 0.002	0.069 ± 0.000	0.705 ± 0.002	0.126 ± 0.000
LSTM	0.635 ± 0.018	0.069 ± 0.002	0.713 ± 0.011	0.126 ± 0.004
LSTM-A	0.605 ± 0.028	0.065 ± 0.003	0.729 ± 0.022	0.120 ± 0.005
TCN	0.642 ± 0.006	0.070 ± 0.001	0.708 ± 0.005	0.127 ± 0.001
DCRNN	**0.663 ± 0.004**	**0.077 ± 0.000**	**0.744 ± 0.005**	**0.140 ± 0.001**
NRI	0.640 ± 0.052	0.071 ± 0.005	0.722 ± 0.061	0.130 ± 0.007

5.1 Comparing informed and uninformed models on simulated and real-world data

First, we compare different *uninformed* and *informed* models on the simulated and real-word data sets. By *uninformed*, we mean that the models receive no information about the incident and base their predictions solely on the observed traffic flow before the incident, i.e., $\boldsymbol{X}_{\text{traffic}}$, and by *informed* we mean models that also receive the incident information I. The models follow the framework presented in Section 3.2, with the incident report module removed for the uninformed models.

We compare our framework with six neural network architectures that have been used for traffic flow prediction. First, we have four models that create an embedding for each sensor location independently:

- *MLP*: Uses a 3-layer MLP.
- *LSTM*: Uses the last hidden state from an LSTM layer.
- *LSTM-A*: Uses self-attention on the hidden states from an LSTM layer.
- *TCN*: Uses a Temporal Convolution Layer [11].

Then we have two different GNN-based models:

- *DCRNN*: Uses a DCRNN layer to create the embedding. The adjacency matrix used is the road network with the distance between sensors as weights [12].
- *NRI*: Uses an NRI model to create the embedding. As the encoder in the NRI uses a fully connected adjacency matrix, no road network is added to the model [10].

We train all models until convergence and calculate the mean performance over four runs using different random seeds.

5.1.1 Uninformed models. First, we look at how well uninformed models are able to predict the congestion labels $\boldsymbol{L}_{class}$. The accuracy, precision, recall, and F1 scores of the uninformed models' classifications can be seen in Table 1. From the table, we can see that all uninformed models perform poorly with low precision scores, with the highest being DCRNN with 0.077. The DCRNN model does have a better recall of 0.744 but only achieves an F1 score of 0.140. The uninformed models' bad classifications are unsurprising as these models lack information on which to base their predictions. They only observe the pre-incident traffic flows and guess the impact of the unknown incident. The graph-based DCRNN and NRI know or infer a structure of the road network and are therefore able to make more coherent guesses. Hence these models have a slight advantage in predicting the impact of the incident over the per-node models.

Table 2: Classification performance of informed and network-informed models on the simulated dataset. We report the scores on the full test set along with the scores on the out-of-distribution subset. The mean of four runs is reported along with one the standard deviation.

	Model	Accuracy	Precision	Recall	F1
Full test set	HA	0.980	0.710	0.750	0.730
	Inf-LSTM	0.729 ± 0.001	0.098 ± 0.000	0.778 ± 0.002	0.175 ± 0.001
	Inf-LSTM-A	0.726 ± 0.003	0.097 ± 0.001	0.775 ± 0.002	0.173 ± 0.001
	Inf-DCRNN	0.968 ± 0.000	0.842 ± 0.001	0.149 ± 0.000	0.253 ± 0.000
	Inf-NRI	0.962 ± 0.005	0.512 ± 0.145	0.241 ± 0.063	0.314 ± 0.029
	NetInf-LSTM	0.978 ± 0.001	0.657 ± 0.021	0.866 ± 0.030	0.747 ± 0.003
	NetInf-LSTM-A	0.979 ± 0.001	0.667 ± 0.011	0.877 ± 0.017	0.758 ± 0.003
	NetInf-DCRNN	0.975 ± 0.000	0.610 ± 0.002	0.867 ± 0.008	0.716 ± 0.002
	NetInf-NRI	**0.981 ± 0.003**	**0.669 ± 0.041**	**0.966 ± 0.013**	**0.789 ± 0.025**
Out of distribution	HA	0.967	0.541	0.617	0.577
	Inf-LSTM	0.697 ± 0.002	0.090 ± 0.001	0.817 ± 0.004	0.163 ± 0.001
	Inf-LSTM-A	0.694 ± 0.003	0.089 ± 0.001	0.811 ± 0.003	0.160 ± 0.002
	Inf-DCRNN	0.969 ± 0.000	**0.840 ± 0.001**	0.155 ± 0.000	0.262 ± 0.001
	Inf-NRI	0.962 ± 0.006	0.508 ± 0.162	0.252 ± 0.059	0.321 ± 0.021
	NetInf-LSTM	0.979 ± 0.001	0.652 ± 0.025	0.906 ± 0.027	0.758 ± 0.008
	NetInf-LSTM-A	0.979 ± 0.001	0.651 ± 0.019	0.921 ± 0.016	0.763 ± 0.004
	NetInf-DCRNN	0.976 ± 0.000	0.616 ± 0.002	0.876 ± 0.011	0.723 ± 0.004
	NetInf-NRI	**0.980 ± 0.003**	0.646 ± 0.041	**0.970 ± 0.013**	**0.775 ± 0.026**

5.1.2 Informed models. Next, we extend the models also to take information from the incident reports as input. We do this in two ways. First, we add the incident information $\boldsymbol{I}$ and zero pad it for the non-incident sensors. Models with this additional input are specified as *Inf* models. Furthermore, we extend the Boolean feature to be a fraction that describes how far upstream the sensor is of the incident. We do this by adding an exponential decay upstream by

$$i_{\text{upstream}} = e^{-0.25\, l} \tag{12}$$

where l is the number of links the node is upstream from the incident. We add this feature to the zero-padded incident report of the *Inf* models. Models with the extended upstream feature are specified as *NetInf* models. We also create a Historical Average (HA) model based on the incident report. The HA model uses the incident report to find the most similar incident in the training set and copy its SIR as the prediction.

The classification results of the *Inf*, *NetInf*, and HA models on the full test set and the out-of-distribution subset can be seen in Table 2. As seen from the table, adding the incident report improves the classification of the impact area of the models. For the Inf-LSTM and the Inf-LSTM-A, the F1 scores go from 0.126 to 0.173 and 0.175, respectively. However, it is worth noting that they both still achieve a precision of less than 0.1. If we look at the graph-based Inf-DCRNN and Inf-NRI, we can see that the inclusion of the incident report has a more significant impact on the performance of these models, with the models achieving F1 scores of 0.253 and 0.314, respectively. The Inf-LSTM and Inf-LSTM-A do not improve as much as the graph-based models because even though the models receive information about where the incident happened, they have no knowledge of the structure of the road network. Therefore, the incident report only helps predict the impact of the incident sensor itself, not any road links to which the congestion can propagate. The Inf-DCRNN and Inf-NRI models have an adjacency with the road structure or an inferred equivalent. Hence they can use the incident report also to find the local upstream area that the congestion can spread to. This enables the models to achieve higher precision scores of 0.842 for Inf-DCRNN and 0.512 for NRI.

Looking at the NetInf models, we can see that adding the indicator variable further improves the predictions. The NetInf models achieve an F1 score over 0.71, with NetInf-NRI being the best with 0.789. The NetInf-LSTM and the NetInf-LSTM-A also achieve high F1 scores with 0.746 and 0.758, respectively, outperforming NetInf-DCRNN, which only reaches 0.716. Also note that the HA model achieves an F1 score of 0.730, beating the DCRNN model. It is clear that by adding the upstream indicator variable, i_{upstream}, all models get the structure of the upstream area of the incident, which dramatically improves the NetInf models' performance. Furthermore, the difference between the graph-based models and the models operating on each sensor separately is diminished. This shows that, for congestion propagation prediction, the inclusion of an upstream indicator is more important than an advanced graphical model. From the results of the out-of-distribution subset of the test set, we can see that the HA model has a big drip in performance, only achieving an F1 score of 0.577. However, all the Inf and NetInf models perform similarly to how they did on the full test set. This means that these models are able to generalize to incidents on road links with no previous incident history, which is paramount for an application in a real-world scenario with limited data.

The regression tasks of the temporal predictions and the effect on traffic speed can be seen in Table 3 for the full test set and Table 4 for the out-of-distribution subset. In the tables, we show the mean absolute error (MAE) and the masked mean absolute percentage error (MMAPE). We mask the mean absolute percentage error to the affected sensors to get more interpretable results. For the full test set, we see that the HA model consistently achieves a lower MMAPE than the other methods and is only beaten on the MAE of the start time prediction by NetInf-NRI. The HA model performs especially well in MMAPE for speed in the predicted area, with a score of only 0.08. It is worth noting that on the predicted impact area, the MMAPE becomes all of the correctly classified road links. So, in cases where the HA model correctly classifies something as affected, the historical change in speed that the HA model uses is also a good prediction for the change in speed. The NetInf models all achieve similar results, with the NetInf-NRI being slightly better for the start time and change in speed predictions but slightly worse in the end time prediction. Generally, we can see that the NetInf models achieve MMAPES between 0.399 and 0.498. For the out-of-distribution subset, we can see that the HA model again struggles to generalize while the NetInf models are much better. The NetInf-LSTM-A model is the best model on the true SIR, achieving an MMAPE of 0.433, 0.404, and 0.338 on the start time, end time, and change in speed, respectively. NetInf-LSTM-A also achieves the best MAEs on the predicted SIR, while the HA model still gets better MMAPE scores. However, again note that the MMAPE on the predicted SIR is the MAPE of the sensors where the models correctly classify congestion.

Next, we also compare the informed models on the real-world dataset. The classification results can be seen in Table 5. In line with the results of the simulated data, the Inf-LSTM-A and Inf-LSTM models are the worst-performing models, with F1 scores of 0.085 and 0.087, respectively. NetInf-DCRNN achieves the best F1 score with 0.227 beating the other NetInf models. However, NetInf-DCRNN achieves the lowest recall of the NetInf models with 0.601 but gets a significantly better precision with 0.140. The HA model achieves a precision of 0.153 but has a low recall of only 0.170. This shows that the findings from the simulated data carry over to the real world. Adding in the i_{upstream} vastly improves the models. However, here NetInf-DCRNN beats the other NetInf models. NetInf-DCRNN achieves the lowest recall of the NetInf models but gets significantly better precision. Similar to the out-of-distribution test on the simulated data, the HA model does not achieve the same performance as the NetInf models, showing how important generalizability is for a real-world application. The regression results on the real-world dataset can be seen in Table 6. As seen from the table, for the true SIR, the NetInf-LSTM-A model has the best start and end time predictions, while the NetInf-LSTM model has the best predictions of the change in speed. It is also worth noting that all NetInf models perform better than HA. For the predicted SIR, we can see that the NetInf-LSTM-A and NetInf-LSTM models have better MAE scores, while the NetInf-DCRNN and NetInf-NRI models have better MMAPE scores, except for the MMAPE of the change in speed where HA is again performing well.

Comparing the performance of the models on the simulated data versus the real-world data, we can see that the models are better on the simulated data. There could be several reasons for this. The spatial size of the real-world data set is much larger than the simulated data, has more intersections, and has long distances between sensors. Hence, the spatial dependencies of the real-world data are harder to detect from the data. Another point is that the incident reports in the simulation are more precise. They contain the number of blocked lanes, while the real-world incident reports only have the severity level and TMC codes, which do not directly say how much of the traffic is blocked by the incident.

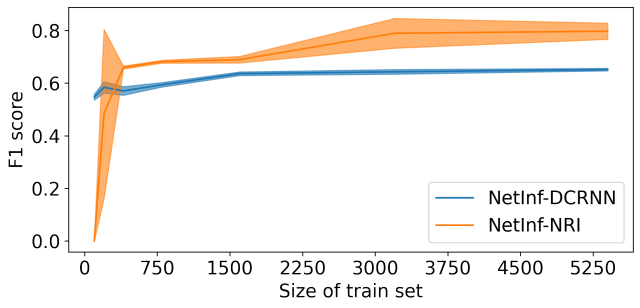


Figure 5: F1 scores of NetInf-DCRNN and NetInf-NRI when trained with varying training set sizes on the simulated data.

5.2 Dataset size's impact on performance

We investigate how the classification performance of NetInf-DCRNN and the NetInf-NRI vary depending on dataset size. We keep the test data set fixed and vary the size of the train data set. We then calculate the F1 scores of the models' classifications. The results

Table 3: MAE and MMAPE of the models' predictions of the start time, end time, and change in speed. The metrics are calculated both on the true affected area and the models' predicted affected area for the full test set of the simulated dataset. Note that the MMAPE on the predicted SIR corresponds to the MAPE on correctly classified congested sensors.

		Start time (sec)		**End time** (sec)		**Δ Speed** (km/h)	
Area	**Model**	MAE	MMAPE	MAE	MMAPE	MAE	MMAPE
True SIR	HA model	11.082	**0.446**	**28.851**	**0.435**	**6.700**	**0.310**
	NetInf-LSTM-A	11.875 ± 0.519	0.479 ± 0.025	31.847 ± 0.812	0.469 ± 0.017	8.184 ± 0.275	0.387 ± 0.014
	NetInf-DCRNN	12.403 ± 0.013	0.478 ± 0.004	31.874 ± 0.122	0.469 ± 0.005	9.076 ± 0.077	0.421 ± 0.003
	NetInf-LSTM	12.037 ± 0.816	0.489 ± 0.065	32.006 ± 0.800	0.470 ± 0.019	8.496 ± 0.324	0.399 ± 0.013
	NetInf-NRI	**10.991 ± 0.201**	0.463 ± 0.017	31.416 ± 1.152	0.498 ± 0.045	6.978 ± 0.439	0.357 ± 0.012
Predicted SIR	HA	11.484	**0.262**	**30.113**	**0.247**	**7.349**	**0.080**
	NetInf-LSTM-A	**10.247 ± 0.139**	0.406 ± 0.018	31.571 ± 0.701	0.395 ± 0.019	8.263 ± 0.180	0.302 ± 0.005
	NetInf-DCRNN	11.105 ± 0.053	0.398 ± 0.004	33.174 ± 0.236	0.387 ± 0.010	9.173 ± 0.042	0.333 ± 0.003
	NetInf-LSTM	10.411 ± 0.073	0.412 ± 0.057	31.749 ± 0.454	0.387 ± 0.015	8.338 ± 0.226	0.306 ± 0.010
	NetInf-NRI	11.288 ± 0.557	0.444 ± 0.019	33.611 ± 1.791	0.48 ± 0.047	8.973 ± 0.350	0.334 ± 0.005

Table 4: MAE and MMAPE of the models' predictions of the start time, end time, and change in speed. The metrics are calculated both on the true affected area and the models' predicted affected area for the out-of-distribution subset of the simulated datasets test set. Note that the MMAPE on the predicted SIR corresponds to the MAPE on correctly classified congested sensors.

		Start time (sec)		**End time** (sec)		**ΔSpeed** (km/h)	
Area	**Model**	MAE	MMAPE	MAE	MMAPE	MAE	MMAPE
True SIR	HA	12.517	0.581	33.419	0.555	9.7732	0.436
	NetInf-LSTM-A	**10.268 ± 0.531**	**0.433 ± 0.019**	**24.808 ± 0.889**	**0.404 ± 0.018**	**7.124 ± 0.294**	**0.338 ± 0.015**
	NetInf-DCRNN	11.671 ± 0.064	0.469 ± 0.004	28.326 ± 0.265	0.447 ± 0.007	8.736 ± 0.063	0.406 ± 0.003
	NetInf-LSTM	10.398 ± 0.723	0.442 ± 0.066	25.003 ± 1.115	0.405 ± 0.021	7.516 ± 0.373	0.352 ± 0.015
	NetInf-NRI	10.447 ± 0.212	0.464 ± 0.019	27.108 ± 1.307	0.480 ± 0.044	7.461 ± 0.437	0.389 ± 0.012
Predicted SIR	HA	13.719	**0.322**	36.148	**0.279**	11.098	**0.086**
	NetInf-LSTM-A	**10.635 ± 0.390**	0.385 ± 0.011	**29.715 ± 0.339**	0.353 ± 0.021	**8.375 ± 0.204**	0.281 ± 0.005
	NetInf-DCRNN	11.72 ± 0.124	0.394 ± 0.006	32.758 ± 0.412	0.369 ± 0.013	9.530 ± 0.037	0.322 ± 0.006
	NetInf-LSTM	10.692 ± 0.235	0.386 ± 0.057	29.795 ± 0.854	0.343 ± 0.015	8.432 ± 0.218	0.285 ± 0.008
	NetInf-NRI	11.655 ± 0.502	0.448 ± 0.019	32.261 ± 1.806	0.463 ± 0.045	9.662 ± 0.300	0.369 ± 0.008

Table 5: Classification performance of HA, Inf, and NetInf models on the real world dataset. The mean of four runs is reported along with one standard deviation.

Model	Accuracy	Precision	Recall	F1
HA	**0.949**	**0.153**	0.170	0.162
Inf-LSTM-A	0.704 ± 0.011	0.047 ± 0.001	0.474 ± 0.024	0.085 ± 0.002
Inf-LSTM	0.724 ± 0.034	0.048 ± 0.001	0.456 ± 0.065	0.087 ± 0.002
NetInf-LSTM	0.772 ± 0.004	0.113 ± 0.001	0.998 ± 0.004	0.203 ± 0.002
NetInf-LSTM-A	0.771 ± 0.001	0.112 ± 0.000	**0.999 ± 0.002**	0.202 ± 0.000
NetInf-DCRNN	0.881 ± 0.007	0.140 ± 0.003	0.601 ± 0.034	**0.227 ± 0.003**
NetInf-NRI	0.833 ± 0.007	0.117 ± 0.001	0.726 ± 0.024	0.202 ± 0.001

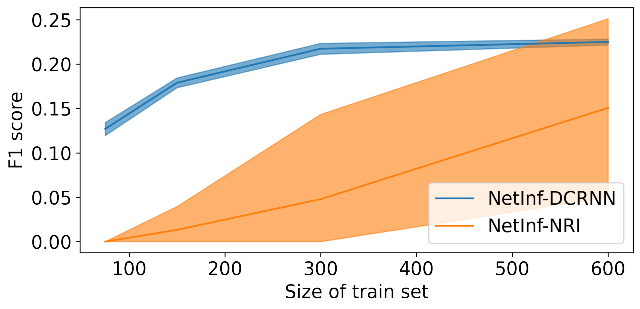


Figure 6: F1 scores of NetInf-DCRNN and NetInf-NRI when trained with varying training set sizes on the PEMS data.

on the simulated data set can be seen in Figure 5, and the results on the real-world data can be seen in Figure 6. As can be seen from both figures, NetInf-NRI is harder to train, leading to higher variance between runs than NetInf-DCRNN, and struggles to converge for lower training set sizes. Hence, for training set sizes lower than 400 observations, NetInf-NRI begins to fail to converge on the simulated data. It can also be seen in the figures that NetInf-NRI improves more with more data than NetInf-DCRNN, suggesting that for extensive datasets, NetInf-NRI could achieve better performance. However, this shows how selecting simpler models can be beneficial for real-world applications where data is scarce.

Table 6: MAE and MMAPE of the HA and NetInf models' predictions of the start time, end time, and change in speed on the real-world dataset. The metrics are calculated both on the true affected area and the models' predicted affected area. Note that the MMAPE on the predicted SIR corresponds to the MAPE on correctly classified congested sensors.

		Start time (sec)		**End time** (sec)		**ΔSpeed** (km/h)	
Area	**Model**	MAE	MMAPE	MAE	MMAPE	MAE	MMAPE
True SIR	HA	22.984	1.085	32.889	1.002	20.442	0.940
	NetInf-LSTM-A	**21.504 ± 0.172**	**0.837 ± 0.013**	**31.06 ± 0.775**	**0.807 ± 0.034**	20.062 ± 1.213	0.826 ± 0.068
	NetInf-DCRNN	22.247 ± 0.112	0.891 ± 0.006	32.423 ± 0.232	0.866 ± 0.01	20.374 ± 0.135	0.863 ± 0.009
	NetInf-LSTM	21.741 ± 0.154	0.841 ± 0.016	31.298 ± 0.856	0.816 ± 0.037	**19.812 (± 0.323)**	**0.801 (± 0.011)**
	NetInf-NRI	22.105 ± 0.016	0.872 ± 0.004	32.11 ± 0.343	0.844 ± 0.014	20.265 (± 0.067)	0.829 (± 0.005)
Predicted SIR	HA	21.163	1.514	30.948	1.010	22.037	**0.649**
	NetInf-LSTM-A	5.204 ± 0.264	0.836 ± 0.013	**7.304 ± 0.426**	0.807 ± 0.034	**4.698 ± 0.226**	0.826 ± 0.06)
	NetInf-DCRNN	5.855 ± 0.271	**0.817 ± 0.002**	8.668 ± 0.254	**0.778 ± 0.008**	6.102 ± 0.162	0.773 ± 0.002
	NetInf-LSTM	**4.905 ± 0.236**	0.841 ± 0.016	7.306 ± 0.323	0.815 ± 0.038	4.755 ± 0.124	0.800 ± 0.012
	NetInf-NRI	5.379 ± 0.179	0.82 ± 0.003	7.837 ± 0.328	0.785 ± 0.019	5.328 ± 0.038	0.764 ± 0.004

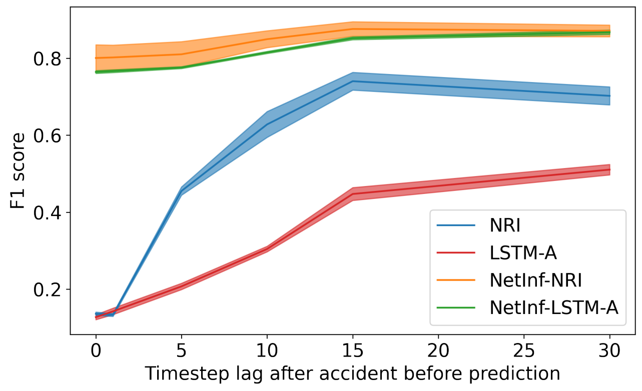


Figure 7: F1 scores for NRI, NetInf-NRI, LSTM-A, and NetInf-LSTM-A for varying numbers of timesteps after the incident before prediction on the full test set of the simulated dataset.

5.3 Impact of observing congestion before prediction

A reasonable argument against the importance of using incident reports in traffic predictions is that modern machine-learning methods are great at adapting to the current state of traffic. As such methods should be able to infer that an incident has occurred from the first congested observations and update the predictions accordingly. To test if this hypothesis is true, we trained models that used later observations to predict the impact of the incident. I.e., we slide the window of observed traffic flow data to later timesteps by varying S in

$$\boldsymbol{X}_{\text{traffic}}{}^{(t_i-q+S:t_i+S)} \tag{13}$$

We apply this method to both the uninformed NRI and NetInf-NRI models, as well as the uninformed LSTM-A and NetInf-LSTM-A models. The results can be seen in Fig. 7. As can be seen from the figure, the F1 score of the uninformed models is low for predictions immediately after the incident happens. As the models then observe data from after the incident happens, the F1 score improves. With the data from after the incident, the models can infer that an incident has occurred and update the predictions accordingly. It can also be seen that the NRI model improves more with the later observations than the LSTM-A model. This is probably because the NRI model infers a graph that the model can use to better predict how the congestion will spread spatially, while the LSTM-A model can only infer which edges the congestion will spread to after the congestion starts affecting an edge. Looking at the NetInf models, we can see that with the addition of the incident report, the immediate predictions are much better. The immediate predictions from the NetInf models are much better than those from uninformed models for any value of s. Hence, including incident reports in the models not only leads to better immediate predictions but also leads to better predictions overall.

6 CONCLUSION

In this paper, we have shown how including incident reports in models for congestion propagation prediction can vastly improve predictive performance. We proposed a novel framework utilizing deep neural networks that predicts the spatio-temporal impact region of incidents based on historical traffic flow and incident reports. We created two new datasets for congestion propagation prediction. First, we extended the SUMO microscopic traffic simulator to simulate incidents and simulated a large-scale dataset of incidents and their following congestion. Next, we took real-world incident reports and combined them with traffic flow data from PEMS. While the proposed simulated and real-world datasets are great first benchmarks, they have some limitations that should be considered. First, the simulated dataset is limited in the number of exogenous variables that it considers. For example, the weather is not implemented in the SUMO simulator, even though it is known to have a big impact on traffic. For the real-world dataset, another limitation is the resolution of the data. The data used comes from PEMS, one of the most comprehensive traffic sensor systems in the world. However, the distance between sensors and the 5-minute sampling rate still limits the precision with which congestion can be measured accurately. This problem also gets bigger when applying the proposed framework to areas with even lower spatial and temporal resolution. Analyzing how exogenous variables and spatio-temporal resolution affects the congestion predictions are both interesting directions for future work. The two datasets were used to do an empirical study of the congestion propagation predictions from uninformed, informed, and historical average models. The results showed that including incident reports in the framework leads to significantly more accurate predictions both on simulated and real-world data and how including an upstream indicator variable can be sufficient to get otherwise spatially unaware models to perform as well as models based on advanced graph neural networks. However, the results also showed that spatial awareness of the upstream structure is necessary. By training models that predict at various points in time after the incident, we demonstrated that the immediate predictions from models utilizing incident reports are superior to predictions from uninformed models, even when the uninformed models have observed the following congestion. Lastly, we analyzed the impact of dataset sizes on the prediction performance, thus highlighting the importance of the creation of large-scale open benchmark datasets, such as the ones that we developed and released, for traffic incident prediction research. Moreover, our analysis also showed that NRI requires more data than DCRNN to get good performance, thus indicating that relying on simpler models can be beneficial for real-world applications where incident data is scarce.

ACKNOWLEDGMENTS

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 875530.

REFERENCES

[1] Mostafa Amin-Naseri, Pranamesh Chakraborty, Anuj Sharma, Stephen B Gilbert, and Mingyi Hong. 2018. Evaluating the Reliability, Coverage, and Added Value of Crowdsourced Traffic Incident Reports from Waze. *Transportation Research Record* 2672 (2018), 34 – 43.

[2] State of California CalTrans. 2023. *CalTrans FAQ.* https://dot.ca.gov/programs/public-affairs/faqs

[3] Danni Cao, Jianjun Wu, Xianlei Dong, Huijun Sun, Xiaobo Qu, and Zhenzhen Yang. 2021. Quantification of the impact of traffic incidents on speed reduction: A causal inference based approach. *Accident Analysis & Prevention* 157 (2021), 106163. https://doi.org/10.1016/j.aap.2021.106163

[4] Zheng Fang, Qingqing Long, Guojie Song, and Kunqing Xie. 2021. Spatial-Temporal Graph ODE Networks for Traffic Flow Forecasting. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining* (Virtual Event, Singapore) *(KDD '21)*. Association for Computing Machinery, New York, NY, USA, 364–373. https://doi.org/10.1145/3447548.3467430

[5] Shota Fukuda, Hideaki Uchida, Hideki Fujii, and Tomonori Yamada. 2020. Short-term prediction of traffic flow under incident conditions using graph convolutional recurrent neural network and traffic simulation. *IET Intelligent Transport Systems* 14, 8 (2020), 936–946. https://doi.org/10.1049/iet-its.2019.0778 arXiv:https://ietresearch.onlinelibrary.wiley.com/doi/pdf/10.1049/iet-its.2019.0778

[6] S. Guo, Youfang Lin, Ning Feng, Chao Song, and Huaiyu Wan. 2019. Attention Based Spatial-Temporal Graph Convolutional Networks for Traffic Flow Forecasting. In *AAAI Conference on Artificial Intelligence.*

[7] Maxime Guériau and Ivana Dusparic. 2020. Quantifying the impact of connected and autonomous vehicles on traffic efficiency and safety in mixed traffic. In *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC).* 1–8. https://doi.org/10.1109/ITSC45102.2020.9294174

[8] Bo Hui, Da Yan, Haiquan Chen, and Wei-Shinn Ku. 2021. TrajNet: A Trajectory-Based Deep Learning Model for Traffic Prediction. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining* (Virtual Event, Singapore) *(KDD '21)*. Association for Computing Machinery, New York, NY, USA, 716–724. https://doi.org/10.1145/3447548.3467236

[9] Weiwei Jiang and Jiayun Luo. 2021. Graph Neural Network for Traffic Forecasting: A Survey. *Expert Syst. Appl.* 207 (2021), 117921.

[10] Thomas N. Kipf, Ethan Fetaya, Kuan-Chieh Wang, Max Welling, and Richard S. Zemel. 2018. Neural Relational Inference for Interacting Systems. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018 (Proceedings of Machine Learning Research, Vol. 80)*. PMLR, 2693–2702. http://proceedings.mlr.press/v80/kipf18a.html

[11] Colin Lea, René Vidal, Austin Reiter, and Gregory D. Hager. 2016. Temporal Convolutional Networks: A Unified Approach to Action Segmentation. (2016), 47–54.

[12] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. 2018. Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting. In *International Conference on Learning Representations (ICLR '18).*

[13] Binbing Liao, Jingqing Zhang, Chao Wu, Douglas McIlwraith, Tong Chen, Shengwen Yang, Yike Guo, and Fei Wu. 2018. Deep Sequence Learning with Auxiliary Information for Traffic Prediction. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (London, United Kingdom) *(KDD '18)*. Association for Computing Machinery, New York, NY, USA, 537–546. https://doi.org/10.1145/3219819.3219895

[14] Pablo Alvarez Lopez, Michael Behrisch, Laura Bieker-Walz, Jakob Erdmann, Yun-Pang Flötteröd, Robert Hilbrich, Leonhard Lücken, Johannes Rummel, Peter Wagner, and Evamarie Wießner. 2018. Microscopic Traffic Simulation using SUMO, In The 21st IEEE International Conference on Intelligent Transportation Systems. *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, 2575–2582. https://elib.dlr.de/127994/

[15] Sohrab Mamdoohi and Elise Miller-Hooks. 2022. Identifying the Impact Area of a Traffic Event Through k-Means Clustering. *Journal of Big Data Analytics in Transportation* (2022).

[16] Sobhan Moosavi, Mohammad Hossein Samavatian, Arnab Nandi, Srinivasan Parthasarathy, and Rajiv Ramnath. 2019. Short and Long-Term Pattern Discovery Over Large-Scale Geo-Spatiotemporal Data. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '19)*. Association for Computing Machinery, 2905–2913. https://doi.org/10.1145/3292500.3330755

[17] Transportation Research Board National Research Council. 2000. *Highway capacity manual.* TRB.

[18] United Nations. 2018. The World's cities in 2018. *Department of Economic and Social Affairs, Population Division, World Urbanization Prospects* (2018), 1–34.

[19] Zheyi Pan, Yuxuan Liang, Weifeng Wang, Yong Yu, Yu Zheng, and Junbo Zhang. 2019. Urban Traffic Prediction from Spatio-Temporal Data Using Deep Meta Learning. *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (2019).

[20] Inon Peled, Raghuveer Kamalakar, Carlos Lima Azevedo, and Francisco C. Pereira. 2020. QTIP: Quick simulation-based adaptation of traffic model per incident parameters. *Journal of Simulation* 16, 2 (may 2020), 111–131. https://doi.org/10.1080/17477778.2020.1756702

[21] Cyrus Shahabi and Bei(Penny) Pan. 2017. *Accident Impact Prediction.* Springer International Publishing, Cham, 39–48. https://doi.org/10.1007/978-3-319-17885-1_1568

[22] David Smith, Soufiene Djahel, and John Murphy. 2014. A SUMO based evaluation of road incidents' impact on traffic congestion level in smart cities. In *39th Annual IEEE Conference on Local Computer Networks Workshops.* 702–710. https://doi.org/10.1109/LCNW.2014.6927724

[23] Cong Tang, Jingru Sun, Yichuang Sun, Mu Peng, and Nianfei Gan. 2020. A General Traffic Flow Prediction Approach Based on Spatial-Temporal Graph Attention. *IEEE Access* 8 (2020), 153731–153741.

[24] TomTom. 2023. London Traffic Report: Tomtom traffic index. https://www.tomtom.com/traffic-index/london-traffic/

[25] Department for Transport. 2023. Road traffic estimates in Great Britain: 2022. https://www.gov.uk/government/statistics/road-traffic-estimates-in-great-britain-2022/road-traffic-estimates-in-great-britain-2022-traffic-in-great-britain-by-vehicle-type

[26] Mathias Niemann Tygesen, Francisco Camara Pereira, and Filipe Rodrigues. 2023. Unboxing the graph: Towards interpretable graph neural networks for transport prediction through neural relational inference. *Transportation Research Part C: Emerging Technologies* 146 (2023), 103946. https://doi.org/10.1016/j.trc.2022.103946

[27] Xi Wang, Yibo Chai, Hui Li, Wenbin Wang, and Weishan Sun. 2021. Graph Convolutional Network-based Model for Incident-related Congestion Prediction: A Case Study of Shanghai Expressways. *ACM Trans. Manag. Inf. Syst.* 12 (2021), 21:1–21:22.

[28] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, and Chengqi Zhang. 2019. Graph WaveNet for Deep Spatial-Temporal Graph Modeling. In *International Joint Conference on Artificial Intelligence.*

[29] Kaixin Yuan, Jing Liu, and Jian Lou. 2022. Higher-Order Masked Graph Neural Networks for Traffic Flow Prediction. *2022 IEEE International Conference on Data Mining (ICDM)* (2022), 1305–1310.

[30] Xian Zhou, Yanyan Shen, and Linpeng Huang. 2020. FreqST: Exploiting Frequency Information in Spatiotemporal Modeling for Traffic Prediction. *2020 IEEE International Conference on Data Mining (ICDM)* (2020), 1442–1447.

Received 2 February 2023