2022

A Deep Learning Approach for Spatiotemporal-Data-Driven Traffic State Estimation

Amr Hatem Ragaa Abdelraouf
University of Central Florida

A DEEP LEARNING APPROACH FOR SPATIOTEMPORAL-DATA-DRIVEN TRAFFIC STATE ESTIMATION

by

AMR ABDELRAOUF
B.Sc., German University in Cairo, Egypt, 2015
M.Sc., Technical University of Munich, Germany, 2018

A dissertation submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy
in the Department of Civil, Environmental and Construction Engineering
in the College of Engineering and Computer Science
at the University of Central Florida
Orlando, Florida

Spring Term
2022

Major Professor: Mohamed Abdel-Aty

ABSTRACT

The past decade witnessed rapid developments in traffic data sensing technologies in the form of roadside detector hardware, vehicle on-board units, and pedestrian wearable devices. The growing magnitude and complexity of the available traffic data has fueled the demand for data-driven models that can handle large scale inputs. In the recent past, deep-learning-powered algorithms have become the state-of-the-art for various data-driven applications. In this research, three applications of deep learning algorithms for traffic state estimation were investigated. Firstly, network-wide traffic parameters estimation was explored. An attention-based multi-encoder-decoder (Att-MED) neural network architecture was proposed and trained to predict freeway traffic speed up to 60 minutes ahead. Att-MED was designed to encode multiple traffic input sequences: short-term, daily, and weekly cyclic behavior. The proposed network produced an average prediction accuracy of 97.5%, which was superior to the compared baseline models. In addition to improving the output performance, the model's attention weights enhanced the model interpretability. This research additionally explored the utility of low-penetration connected probe-vehicle data for network-wide traffic parameters estimation and prediction on freeways. A novel sequence-to-sequence recurrent graph networks (Seq2Se2 GCN-LSTM) was designed. It was then trained to estimate and predict traffic volume and speed for a 60-minute future time horizon. The proposed methodology generated volume and speed predictions with an average accuracy of 90.5% and 96.6%, respectively, outperforming the investigated baseline models. The proposed method demonstrated robustness against perturbations caused by the probe vehicle fleet's low penetration rate. Secondly, the application of deep learning for road weather detection using roadside CCTVs were investigated. A Vision Transformer (ViT) was trained for simultaneous rain and road surface condition classification. Next, a Spatial Self-Attention (SSA) network was

designed to consume the individual detection results, interpret the spatial context, and modify the collective detection output accordingly. The sequential module improved the accuracy of the stand-alone Vision Transformer as measured by the F1-score, raising the total accuracy for both tasks to 96.71% and 98.07%, respectively. Thirdly, a real-time video-based traffic incident detection algorithm was developed to enhance the utilization of the existing roadside CCTV network. The methodology automatically identified the main road regions in video scenes and investigated static vehicles around those areas. The developed algorithm was evaluated using a dataset of roadside videos. The incidents were detected with 85.71% sensitivity and 11.10% false alarm rate with an average delay of 27.53 seconds. In general, the research proposed in this dissertation maximizes the utility of pre-existing traffic infrastructure and emerging probe traffic data. It additionally demonstrated deep learning algorithms' capability of modeling complex spatiotemporal traffic data. This research illustrates that advances in the deep learning field continue to have a high applicability potential in the traffic state estimation domain.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1: INTRODUCTION

1.1 Overview

The ubiquitous deployment of traffic sensing devices, such as loop detectors, microwave sensors, and roadside traffic surveillance cameras results in tremendous amounts of real-time data streams and accumulates a colossal backlog of historical traffic data. Additionally, the increasing adoption of probe devices such as GPS-enable On-Board Units (OBUs) and smartphones led to an upsurge in probe traffic datasets. Furthermore, advances in edge and cloud computing have led to stronger connectivity of traffic sensors and by extension an increased availability of highly granular spatiotemporal traffic data. The understanding, processing, and wielding of real-time and historical traffic data has a profound impact on the progress of traffic state estimation research (Zhang et al. 2011; Zhu et al. 2018) and the development of intelligent transportation system applications. As a result, big data analytics has become an imperative research focus in the traffic state estimation domain (Veres and Moussa 2019).

Classical traffic-data-driven transportation technology research relied on analytical methods, statistical models, and machine learning approaches to model and understand various types of traffic data (Vlahogianni, Karlaftis, and Golias 2014). However, the growing size and complexity of traffic data led to an increase in demand for methodologies that are capable of ingesting, understanding, and interpreting larger amounts of traffic data while accounting for its intrinsic spatiotemporal nature. In the past decade, researchers have successfully employed deep neural networks for a variety of data-driven applications such as pattern recognition, sequence modeling, object detection, and image classification (Schmidhuber 2015). Inspired by their success in other domains, transportation technology researchers have adopted deep neural network

algorithms for numerous applications.

Various neural network models were crafted to handle different types of data. Several neural network subclasses have proved useful in traffic-related research. For instance, recurrent neural networks (RNNs) were designed to handle sequential data (Lipton, Berkowitz, and Elkan 2015) and were therefore well-suited to model the temporal dimension of traffic data in different problems. Convolutional Neural Networks (CNNs) were proposed to handle computer vision task such as object detection and images segmentation (Li et al. 2021; Khan et al. 2020). Hence, they were extensively employed in vision-related traffic applications to handle image and video-based tasks. CNNs were also widely employed to model the spatial characteristics in traffic data by utilizing the convolution and pooling operations. Graph Neural Networks (GNNs) are capable of modeling graph-based spatial topologies (Wu et al. 2019; Shi and Yeung 2018) and were thus utilized in traffic research to model the spatial dimensions in larger traffic networks. As the field of neural networks continues to advance, it introduces more nuanced models and sophisticated architectures that solve various data modeling problems. As demonstrated in recent years, these methodologies continue to have high impact on the traffic state estimation research domain.

In this dissertation, three traffic state estimation research objectives were studied. The proposed methodologies leveraged different types of spatiotemporal big traffic data and explored the utilization of different deep learning algorithms. The first research objective was network-wide traffic parameters estimation and prediction. To implement this objective, both static microwave traffic data and probe-vehicle data were utilized and studied as input data sources. The second research objective was vision-based road weather detection, where the target was to employ traffic surveillance cameras for the recognition of rainy weather conditions and road surface precipitation states. Finally, the third research objective was video-based traffic incident detection. In this

objective, an automatic video-based incident detection algorithm was developed and designed to run in real-time speed.

1.2 Research Objectives

1.2.1 Network-Wide Traffic Parameters Estimation and Prediction

Traffic parameters estimation and prediction are fundamental and long studied tasks in the field of intelligent transportation systems. They're essential tasks for planning applications such as operation planning, safety studies, and traffic scheduling. They're additionally vital for real-time traffic applications such as trip planning, dynamic navigation, and incident detection. Despite their long history in the literature, traffic parameters remain difficult to estimate and predict. The challenge stems from the complex non-linear relationships exhibited between traffic parameters in both the spatial and temporal dimensions. To address this issue, the following research tasks were identified:

1. Developing a neural network architecture which is capable of modeling multiple input traffic sequences in order to capture the different periodic characteristics of traffic. The network should use the multi-sequence input to accurately predict network-wide freeway traffic speed for up to 60 minutes ahead.
2. Focusing on the interpretability of the model output to understand the temporal dependencies between traffic parameters and verify the importance of multi-sequence input traffic data.
3. Exploring the applicability of probe-vehicle data for traffic speed and volume real-time estimation and up to 60-minute ahead prediction on freeways.

Two research studies were conducted in order to address and solve the research gaps in

network-wide traffic parameters estimation and prediction literature. The first research effort utilized attention-based encoder-decoder neural networks for network-wide traffic speed prediction. The research undertaken to achieve this task was presented at the 100th TRB Annual Meeting held in Washington DC 2021 and published in IEEE transaction on Intelligent Transportation Systems (Abdelraouf, Abdel-Aty, and Yuan 2021). This research is comprehensively described in Chapter 3. The second research effort employed connected probe-vehicle data for network-wide traffic parameters estimation and prediction using a sequence-to-sequence recurrent graph convolutional neural network. The research executed to address this objective was detailed in Chapter 4.

1.2.2 Vision-Based Road Weather Detection

Rainy weather conditions and the consequent wet road surface have an unfavorable effect on visibility, vehicle maneuverability, roadway infrastructure, and driver behavior and thus results in undesired consequences for traffic operation and safety. In order to alleviate the negative ramifications of inclement road weather on traffic, it must be constantly and accurately observed. To achieve this research objective, the following tasks were identified:

1. Compiling an image dataset which can be employed to train a neural network algorithm for rain and road surface condition classification.
2. Designing and evaluating a neural network architecture that is capable of classifying rain and road surface conditions using images captured from roadside cameras. The network should be able to utilize the spatial distribution of neighboring cameras to enhance its overall classification accuracy.

This research objective was realized by utilizing roadside traffic cameras for spatial-context-aware rain and road surface condition detection on freeways using Vision Transformers.

The findings of the research conducted to achieve this objective were presented at the 101th TRB Annual Meeting in held in Washington DC 2022 and published in IEEE Transactions on Intelligent Transportation Systems (Abdelraouf, Abdel-Aty, and Wu 2022). The research was thoroughly described in Chapter 5.

1.2.3 Video-Based Traffic Incident Detection

Expeditious traffic incident identification is critical for reducing potential road user fatalities, injuries, and property damage. The comprehensive network of roadside CCTV cameras on US roadways offers constant coverage of the road network and therefore can be a useful tool to monitor traffic incidents. However, due to the extensive number of cameras, it is unfeasible for human operators to constantly monitor all roadway segments simultaneously. The following research tasks were identified to undertake this research objective:

1. Developing a video-based algorithm for automatic incident identification using CCTV cameras.
2. Evaluating the proposed algorithm's detection accuracy and detection delay.
3. Assessing the algorithm's applicability in real-time by measuring the processing throughput.

This research objective was realized through the development of a real-time traffic incident identification algorithm using roadside CCTV cameras. The research was presented at the 101st TRB Annual Meeting held in Washington DC 2022. Chapter 6 delineates the findings of the research conducted to achieve this objective.

1.3 Dissertation Organization

The rest of the dissertation is organized as follows: Chapter 2 summarizes the literature

review and was categorized according to the three research objectives: 1) network-wide traffic parameters estimation and prediction, 2) vision-based road weather detection, and 3) video-based traffic incident identification. Chapter 3 discusses freeway traffic speed prediction by utilizing static microwave traffic sensors and employing a novel attention-based multi-encoder-decoder neural network architecture. Chapter 4 describes the research conducted for traffic speed and volume estimation and prediction using connected probe-vehicle data and applying a sequence-to-sequence recurrent graph convolution network algorithm. Chapter 5 explores the utilization of vision transformers for spatial-context-aware rain and road surface condition detection on freeways. In Chapter 6, the research pertaining to real-time video-based traffic incident identification using roadside CCTV cameras was detailed. Finally, Chapter 7 summarizes the dissertation's key findings and contributions. It additionally concludes and describes the implication of the conducted research.

CHAPTER 2: LITERATURE REVIEW

2.1 Network-Wide Traffic Parameters Estimation and Prediction

2.1.1 Traffic Data Sources

Traffic flow estimation and prediction tasks have been conducted using a variety of static sensors such as loop detectors (Kwon, Varaiya, and Skabardonis 2003; Pan, Demiryurek, and Shahabi 2012; Wilkie, Sewall, and Lin 2013), microwave detectors (Abdelraouf, Abdel-Aty, and Yuan 2021; Ma, Tao, et al. 2015; Jin et al. 2018), and traffic cameras (Mahmoud et al. 2021a, 2021b; Zhan, Li, and Ukkusuri 2015). For instance, Pan et al. (Pan, Demiryurek, and Shahabi 2012) used speed, volume, and occupancy data collected from loop detectors deployed in Los Angeles County in order to predict traffic speed. Mahmoud et al. (Mahmoud et al. 2021a) utilized traffic counts collected from intersection-facing CCTV cameras in order to estimate real-time cycle-level traffic movements at signalized intersections based on upstream and downstream traffic data. Mahmoud et al. utilized the same data for predicting cycle-level traffic movements at signalized intersections using machine learning models (Mahmoud et al. 2021b). Despite their ability to provide accurate traffic flow estimations, static detectors are limited by their spatial distribution. Furthermore, they can be costly to set up, deploy, and maintain.

Probe vehicle datasets pose a different set of challenges since they only represent samples of the vehicles on the roadway. Dataset attributes like penetration rate and sampling rate and fleet type affect the granularity, coverage, and driving behavior of the probe vehicle data. Hence, the data cannot be used to directly measure traffic parameters, particularly vehicle volume data. Most of the previous probe-vehicle-based research was focused on travel time estimation (Zheng and Van Zuylen 2013; Efentakis et al. 2013; Pfoser, Tryfona, and Voisard 2006; Wang, Zheng, and

Xue 2014) and travel time forecasting (Zhang et al. 2017; Zhan, Ukkusuri, and Yang 2016; Li et al. 2017; Derrow-Pinion et al. 2021). Derrow-Pinion et al. (Derrow-Pinion et al. 2021) described how Google Maps leverages its collected travel time data to accurately predict trip Estimated Time of Arrival (ETA). Efentakis et al. (Efentakis et al. 2013) utilized probe vehicle data from a fleet of 2000 – 5000 vehicles sampled at 60 – 180 seconds in order to estimate the real-time segment travel times and derive the shortest path for each vehicle in the fleet.

Furthermore, some previous research efforts relied on special types of vehicle fleets such as taxis (Li et al. 2017; Zhan et al. 2013; Wang, Zheng, and Xue 2014; Zhao, Song, et al. 2019) to collect probe vehicle data for traffic modeling. The vehicles in the dataset exhibit driving behavior patterns that may be specific to the dataset but not generalizable for all vehicles. Li et al. (Li et al. 2017) used the trajectories of 15 taxi vehicles deployed in Shenzhen, China. Using a GPS sampling interval of 10 seconds, they predicted travel time on urban arterials. The authors reported that one of the main challenges of the taxi dataset was the inability to differentiate whether the travel time delay was caused by congestion or by the taxi stopping temporarily to pick up or drop off passengers.

2.1.2 Traffic Modeling Methods

2.1.2.1 Parametric Models

In the past decades, parametric models have been used to predict traffic speed. Several works have implemented Autoregressive Integrated Moving Average Model (ARIMA) for time series prediction (Ahmed and Cook 1979; Kumar and Vanajakshi 2015; Chandra and Al-Deek 2009). Additionally, multiple variations of ARIMA were implemented. For example, ARIMA with explanatory variables (ARIMAX) (Williams 2001), seasonal ARIMA (SARIMA) (Williams and

Hoel 2003), space-time ARIMA (Kamarianakis and Prastacos 2003), and Kohonen-ARIMA (KARIMA) (Van Der Voort, Dougherty, and Watson 1996). However, ARIMA models are not the best suited to capture the nonlinear spatiotemporal relationships of traffic speed since the model is built on linear regression concepts. Therefore, the implementations underutilize the learning data.

2.1.2.2 Non-Parametric Models

Conventional machine learning techniques have also been employed to tackle the speed prediction problem. For instance, linear conditional Gaussian Bayesian Networks (Zhu, Peng, et al. 2016), Support Vector Regression (SVR) (Castro-Neto et al. 2009; Hong et al. 2011; Asif et al. 2013), k-Nearest Neighbors (k-NNs) (Chang et al. 2012; Davis and Nihan 1991) and ANNs (Ye, Szeto, and Wong 2012; Van Lint, Hoogendoorn, and van Zuylen 2005; Ma, Yu, et al. 2015b; Tang et al. 2017). Castro-Neto et al. (Castro-Neto et al. 2009) proposed an SVR model to predict short-term traffic flow in typical and atypical traffic conditions and found that the SVR model performed better compared to ANNs. Davis et al. (Davis and Nihan 1991) used k-NNs to forecast freeway traffic and found that they produced comparable results to time-series linear regression models like ARIMA. Ye et al. (Ye, Szeto, and Wong 2012) used GPS data to extract speed and acceleration data on a road segment during irregular intervals. The data was then fed to an ANN that was used to forecast short-term traffic speed (60 seconds ahead). Deeper neural networks have also been applied to the traffic speed prediction problem. Lv et al. (Lv et al. 2014) proposed a Stacked Autoencoder (SAE) architecture to model the spatial and temporal traffic flow patterns. Their results outperformed shallow Backpropagation Neural Networks (BP-NNs) and SVR.

2.1.2.3 Deep Neural Network Models

Recurrent Neural Networks (RNNs) are a class of deep learning algorithms that were designed to model sequential input and can possibly be utilized to construct sequential outputs. In the recent past, RNNs have been widely used to forecast traffic speed (Fu, Zhang, and Li 2016; Ma, Tao, et al. 2015; Tian and Pan 2015; Polson and Sokolov 2017; Zhao et al. 2017; Ma, Yu, et al. 2015a; Cui et al. 2018), especially popular variations like Long Short Term Memory (LSTM) (Hochreiter and Schmidhuber 1997) and Gated Recurrent Units (GRUs) (Chung et al. 2014). These models gained popularity due to their ability to memorize long-term sequential dependencies. Ma et al. (Ma, Tao, et al. 2015) was one of the first to use LSTMs for short-term traffic speed prediction (up to 8 minutes). The LSTM outperformed ARIMA, SVM and non-recurrent variations of neural networks. Zhao et al. (Zhao et al. 2017) modeled the temporal component using LSTMs and the special component using a correlation matrix between detector locations. Fu et al. (Fu, Zhang, and Li 2016) compared the performance of LSTMs and GRUs and reached slightly better results using GRUs. Cui et al. (Cui et al. 2018) used stacked bidirectional and unidirectional LSTMs (SBU-LSTMs) to perform network-wide speed prediction. Bidirectional recurrent units can capture temporal dependencies in both chronological and reverse chronological order.

Convolutional Neural Networks (CNNs) have been exceptionally successful at executing computer vision and image processing tasks in the past decade (Krizhevsky, Sutskever, and Hinton 2012). They have since been leveraged to capture the spatial dependencies across the road network in speed forecasting tasks (Wu and Tan 2016; Liu et al. 2017; Ke et al. 2020; Wang et al. 2016; Cao, Li, and Chan 2020; Li et al. 2018; Mahmoud et al. 2021a, 2021b). For instance, a multi-channel CNN was utilized by Ke et al. (Ke et al. 2020) to model lane-wise traffic speed and volume

data. The proposed method was designed to capture the spatial relationship between adjacent traffic lanes. Cao et al. (Cao, Li, and Chan 2020) extracted the traffic spatial features from the road network using CNNs and then used the derived features as input for an LSTM, which in turn captured the temporal correlation. Liu et al. (Liu et al. 2017) combined convolutions and LSTMs as a single Conv-LSTM neural network architecture (first proposed by Xingjian et al. (Xingjian et al. 2015)) that is capable of both learning spatial and temporal dependencies.

Many research efforts relied on graph convolution networks (GCN) for spatial dependencies extraction (Wu et al. 2019; Yu, Yin, and Zhu 2017; Yu, Lee, and Sohn 2020; Zheng et al. 2020; Zhao, Song, et al. 2019). Graph convolutions rely on extracting spatial features by convolving the directed graph representation of the road network. In this representation, detectors are represented by nodes and adjacent detectors are connected by edges. Spatio-Temporal Graph Convolutional Network (STGCN) was proposed by Yu et al. (Yu, Yin, and Zhu 2017) to extract both spatial and temporal traffic features and subsequently using those features to predict traffic flow by applying graph convolution. Zhao et al. (Zhao, Song, et al. 2019) combined GCN for spatial modeling and GRU for temporal modeling and developed a Temporal Graph Convolution Network (T-GCN). T-GCN was trained and tested for traffic speed prediction using two separate datasets: a probe-vehicle type data, and a loop detector-based data. Guo et al. (Guo et al. 2019) proposed an Attention-based Spatio-Temporal Graph convolution network (ASTGCN). The authors used traffic flow, occupancy, and speed data collected from infrastructure-based sensors to predict traffic flow.

Attention mechanisms were first introduced in the natural language processing field and gained popularity as a result of their ability to determine and preserve sequence-wide dependencies (Bahdanau, Cho, and Bengio 2014; Vaswani et al. 2017). Previous works employed attention

mechanism for traffic prediction (Zheng et al. 2020; Xu et al. 2020; Yu, Lee, and Sohn 2020). Zheng et al. (Zheng et al. 2020) developed a Graph Multi-Attention Network (GMAN) for traffic parameter prediction. Their proposed network employed attention in both the spatial and temporal domains. Their network was trained using traffic detector data for 60-minute ahead speed and volume prediction. Xu et al. (Xu et al. 2020) developed a spatial-Temporal transformer network for traffic flow forecasting. The authors modeled spatial dependencies in the network graph using self-attention.

2.1.3 Summary

Traffic parameters estimation and prediction is a long and deeply studied topic due to its high applicability in a wide range of intelligent transportation systems domains. In the early years, researchers focused on analyzing traffic parameters using classical statistical approaches. The technological development of traffic data collection devices led to an explosion in the availability of traffic data. With larger datasets to model, research efforts looked towards non-parametric approaches that relied less on underlying model assumptions and more towards data fitting. In the recent past, technical developments in computation capabilities led to the development of deep neural networks models with model parameters in the order of millions. As a result, numerous variations of deep neural networks have been employed for traffic parameter estimation and prediction.

Despite the advances in deep-learning-based traffic parameters estimation and prediction, there are still some research gaps. Most proposed deep learning models depended on short-term input sequences for prediction and failed to consider the daily and weekly periodic behavior of traffic. Another problem posed by deep-learning-based approaches is the lack of interpretability. Furthermore, researchers focused on employing traffic data collected from static infrastructure-

based sensors, while the emerging probe-vehicle traffic data was underutilized for traffic parameters estimation and prediction. This dissertation addresses the identified literature shortcomings in the first research objective covered in Chapters 3 and 4.

2.2 Vision-Based Road Weather and Detection

2.2.1 Rain Detection

Various research efforts have attempted to detect rain from images or video feeds. Many of these efforts depend on handcrafted features extracted using different image processing methods. Bossu et al. (Bossu, Hautière, and Tarel 2011) proposed a method to detect rain by analyzing high intensity streaks in the input video stream. Mixture of Gaussians (MoG) was employed to segment moving objects in the foreground from the video feed background. Rain streaks were detected using the difference of pixel intensities between foreground objects and the background model. Photometrical and size-based rules were used to filter out non-streak-shaped foreground objects. Next, the orientations of the extracted streaks were used to construct a histogram of streak orientations (HOS). The shape of the output histogram was approximated using a uniform Gaussian distribution. Finally, the decision of whether or not rain existed in an image was taken by applying a goodness of fit test on the HOS of past images. Allamano et al. (Allamano, Croci, and Laio 2015) used rigorous mathematical analysis to calculate rainfall in a video stream. They detected candidate rain drop clusters by analyzing high intensity pixels in short-term consecutive frames. Next, they took advantage of their calibrated camera and estimated droplet size, velocity and position. The droplet parameters were extracted in each frame. Finally, the frame rate information was added to estimate rainfall in millimeters per hour. Zhang et al. (Zhang et al. 2016) extracted sky and shadow pixels, as well as rain streaks and snowflake clusters from each

image. Additionally, they determined global image features like contrast and saturation. Next, the authors used the extracted parameters to train a Multi Kernel Learning (MKL)-based classifier to label an image as sunny, rainy, snowy, or hazy. Lee et al. (Lee et al. 2016) measured the hue, saturation, and value (HSV) amounts of rainy and non-rainy video segments. A temporal-HSV feature list was generated for each video segment. Similarly, edge-detection was used to count the number of edge pixel in every frame. For every video segment, a temporal-edge-count features list was generated. To determine the weather condition in a test video segment, the temporal-HSV was extracted and compared with the patterns of rainy and non-rainy videos. A similarity score was calculated between the test video and each class.

Due to the superior performance exhibited by convolutional neural networks (CNNs) in various computer vision tasks (Simonyan and Zisserman 2014; Krizhevsky, Sutskever, and Hinton 2012), they have been recently utilized for image and video-based rain detection. Zhu et al. (Zhu, Zhuo, et al. 2016) used a GoogleNet CNN architecture for weather detection. The model weights were initialized using the pre-trained weights on the ILSVRC dataset. The authors collected and labeled a weather dataset. It was used to fine-tune the CNN weights. The model was trained to label images using one of 4 weather labels: runny, rainstorm, blizzard, and fog. Ozcan et al. (Ozcan et al. 2019) collected and labeled roadway CCTV images from the state of Iowa and used image augmentation techniques to enrich their dataset. The authors used the dataset to train a VGG16 CNN to label road images as clear, rainy or snowy. Haurum et al. (Haurum, Bahnsen, and Moeslund 2019) used a tipping-bucket rain gauge and a Laser disdrometer to measure rainfall next to surveillance cameras. They generated a video dataset containing video footage and matching rainfall measurements in millimeters. However, they modeled their problem as a binary classification problem and attempted to differentiate between “rain” and “no rain”. They trained a

3D convolutional neural network to detect rainfall from 8 consecutive video frames. Their method was compared with the one proposed in Bossu et al. (Bossu, Hautière, and Tarel 2011) and produced superior detection accuracy. Zen et al. (Zen et al. 2019) combined deep image features with extracted rain streaks to estimate rainfall from an image. They first collected images from cameras near weather sensors and used the rain gauge measurement in the weather sensors as ground truth. Rain streaks were extracted from each image using DID-MDN algorithm. The authors proposed a CNN architecture that uses the raw image features and the extracted streaks as input. The model was then trained end-to-end to estimate rainfall values. Sirirattanapol et al. (Sirirattanapol et al. 2019) trained a multi-class CNN using traffic images to detect rainy conditions and road surface conditions. The authors implemented a ResNet CNN architecture and used it to obtain multiple labels for each image. Sun et al. (Sun et al. 2020) utilized pre-trained CNNs to implement a 5-class weather detection system. Their proposed system detected sunny, overcast, rainy, snowy, and foggy conditions. They collected and manually labeled an image dataset from highway traffic surveillance cameras. The labeled dataset was used to train and test a 5-class deep convolutional neural network model which classifies input images into one of the selected weather classes.

2.2.2 Road Surface Condition Detection

While some research efforts targeted the combined objective of rain and road state detection (Sirirattanapol et al. 2019), others focus exclusively on detecting the road surface condition. Omer et al. (Omer and Fu 2010) extracted images from on-board camera videos and classified the images into 3 categories: dry, snow tracks, and snow. The authors extracted 2 sets of features to test for snow and snow tracks. The first feature group was designed to capture snow density. To achieve that, authors extracted a 32 bin RGB color histogram from each image. The

second feature set was constructed to capture edge information on the road. Hence, the images were transformed to grayscale, then gaussian smoothing and gradient edge masks were applied. Finally, an SVM classifier was trained on the extracted features to label the images using one of the 3 target categories. Sun et al. (Sun and Jia 2013) proposed a method to label road surface state as dry, light snow, or heavy snow based on a video stream input. The authors first remove the moving objects in the image and obtain the image background. Next, color and texture features were extracted for each grayscale background image. These features included the Angular Second Moment (ASM), which described the image texture smoothness, and Inverse Difference Moment (IDM), which described the image homogeneity. The pixel contrast and entropy were also computed. The features were used to train a Bayesian Network classifier for 3-class classification. Amthor et al. (Amthor, Hartmann, and Denzler 2015) constructed a reflection map from car-dash-mounted cameras to classify the road surface state as dry, wet, or snowy. The authors sampled pavement pixels by identifying a static trapezoid at the bottom of the image and transforming the extracted shape into a square image using a homography matrix. Next, a reflection map was obtained from the image using a Local Binary Patterns (LBPs) texture descriptor. For classification, the authors proposed an Extremely Randomized Trees. The authors reported superior classification results compared to (Omer and Fu 2010) and (Sun et al. 2020). Qian et al. (Qian, Almazan, and Elder 2016) attempted to differentiate between dry, wet and snowy images from on-board vehicle dash cameras. As a preprocessing step, they manually annotated the region of interest (ROI) in each image in their dataset and created an ROI heatmap. Next, they extracted a histogram of textons and a histogram of luminance from the image ROIs. The extracted histograms were used as features to model a supervised learning problem. To solve this problem, the authors compared different machine learning models like Nearest Neighbors, SVM, and

Decision Trees. Ultimately, they were able to produce the highest classification accuracy using a Naïve Bayes Boosting model.

Recently, convolutional neural networks have grown in popularity as a methodology for deep feature extraction and road surface condition classification from images. Roychowdhury et al. (Roychowdhury et al. 2018) developed a road surface classification algorithm as a first step to their target objective, which was road friction estimation. To detect the road surface condition, the hue, saturation, and value color channels (HSV) were extracted and appended to the red, green, and blue color channels (RGB). Furthermore, the authors annotated a segmentation mask for each image to mark the drivable parts of the image on the road. Histogram of gradient (HOG) features were extracted from the segmented drivable road surface. Two models were implemented to classify the road condition in images into one of 4 classes: dry, wet, slush, or ice. Firstly, the HSV and RGB color channels were used to train a CNN. Secondly, the HOG features were used to train a feed-forward neural network. The two models were compared, and the authors reported that the CNN achieved superior classification accuracy. Pan et al. (Pan et al. 2018) implemented a CNN to detect snowy road surface condition from dash-mounted cameras. Their problem was formulated in 3 different ways: a 2-class problem (no snow/snow), a 3-class problem (no snow/ light snow/ heavy snow), and a 5-class problem (no snow + 4 levels of snow). They used a VGG16 architecture with pre-trained weights. The authors expanded on their work in (Pan et al. 2019). They collected another image dataset consisting of images from traffic surveillance cameras mounted on highways. Additionally, they focused their problem definition on 4-class road surface condition classification. The authors tested their VGG16 model against 3 other deep convolutional neural network architecture, namely ResNet50, InceptionV3, and Xception. The ResNet50 model produced the highest classification accuracy on both the traffic surveillance and the dash-mounted

datasets. Grabowski et al. (Grabowski and Czyżewski 2020) detected road surface condition based on 3 categories: dry, wet, snowy. They collected and manually labeled images from traffic surveillance cameras. The authors implemented, trained and compared the results of Resnet, Densenet and VGG19. All the models utilized pre-trained weights. It was found that Densenet produced the highest testing accuracy on the collected dataset.

2.2.3 Summary

Different methodologies were adopted to detect rain in images or video feeds. Many approaches depended on the luminosity of rain pixels. These approaches searched for rain-streak-shaped blobs or high intensity droplet clusters in images by applying filters on images or segmenting the foreground in videos. The image HSL color channels and edge detection filters were also used as features to differentiate between rainy and non-rainy conditions. Deep neural networks were also employed for rain detection. Multiple CNN and 3D-CNN models were used to extract rain detection features from images and videos respectively. Road surface condition was determined in various ways. Some approaches assessed the reflectivity or calculated the luminance of the pavement pixels and used the reflectivity map to determine whether the road surface is wet. Other methods relied on color channel and edge-detection analysis. Deep features were extracted using CNNs and utilized for road surface condition detection. Furthermore, some authors combined color channel analysis with deep convolution layers to detect and classify different road states.

Vision-based rain and road surface condition detection methodologies in the literature targeted stand-alone image classification. As part of the second research objective of this dissertation, the spatial relationship between different input sources has been factored into the proposed methodology and used to improve classification results. To the best of our knowledge,

this is the first effort that explores the geospatial relationship between input images for rain and road surface condition detection.

2.3 Video-Based Traffic Incident Detection

2.3.1 Anomaly Detection in Videos

Video-based incident detection can be characterized as a spatiotemporal anomaly detection problem. Numerous methods have been proposed in the literature to solve the problem of video-based anomaly detection. Most efforts attempted to model normal behavior using a set of features, and then classified data points as anomalous when their features deviated from the regular pattern. Some researches leveraged timeseries statistical models like Hidden Markov Models (Kratz and Nishino 2009; Hospedales, Gong, and Xiang 2009) and sparse reconstruction (Luo, Liu, and Gao 2017; Cong, Yuan, and Liu 2011). In the recent past, deep learning techniques have been more commonly employed to detect anomalies. Hasan et al. (Hasan et al. 2016) used convolutional feed-forward autoencoders for anomaly detection. They used a combination of hand-crafted-feature autoencoders and learned-feature autoencoders. Their autoencoders' reconstruction costs were used to determine whether the input video sequence contains anomalies. Sultani et al. (Sultani, Chen, and Shah 2018) proposed a weakly supervised approach to classify normal and abnormal videos. They divided their input videos into segments and used Multiple Instance Learning (MIL) to assign an anomaly score to each segment. They used 3D convolutional neural networks (Tran et al. 2015) to extract spatiotemporal features from each video segment and transformed those features to an anomaly score. Furthermore, they released the UCF-Crimes dataset (Sultani, Chen, and Shah 2018) which contains 13 categories of videos containing abnormal events like assault, burglary, robbery, and vandalism. Zhong et al. (Zhong et al. 2019) used graph convolutional

networks (GCN) to strengthen the labels in the weakly labeled UCF-Crimes dataset and then solved the problem as a supervised learning anomaly detection problem.

2.3.2 Video-Based Traffic Incident Detection

The general definition of anomaly is subjective and has high inter-class variance. Thus, generic anomaly detectors lack the nuance required to focus on a specific problem such as traffic incident detection (Li et al. 2020). Various approaches have been proposed to tackle the more fine-grained problem of video-based traffic incident detection. Anomaly detection in traffic videos involves defining normal vehicle flow behavior and conversely detecting irregular vehicle trajectories. The proposed approaches can be classified into 3 classes: 1) pixel-based approach, which relies on pixel-level feature extraction to detect anomalies in traffic cameras, 2) object-based approach, which involve detection and tracking of objects of interest on the road scene, and 3) hybrid approach, which leverage both pixel-level and object-level features for detecting incidents in traffic videos (Kumaran, Dogra, and Roy 2019).

2.3.2.1 Pixel-Based Approach

Pixel-based approaches have been widely used to detect incidents in traffic videos. The majority of pixel-based approaches relied on optical flow algorithms such as Lucas-Kanade (Lucas and Kanade 1981) and Farneback (Farnebäck 2003) optical flow. Optical flow estimates the pattern of apparent motion between one video frame and the next. It relies on pixel intensity values to compute a pixel-wise vector field of motion between the first frame and the second frame. The result is a pair of magnitude and orientation values for each pixel. However, optical flow methods are better suited for detecting rapid or intense changes, which generate considerable perturbations in the extracted motion field. Therefore, most pixel-based efforts that rely on optical flow have

limited their scope to traffic *accident* detection.

Sadeky et al. (Sadeky et al. 2010) used Lucas-Kanade optical flow to compute a motion field for each frame in the input video. The output vectors were grouped by angle into a feature space dubbed the histogram of flow gradients (HFG). The values in the HFG were used as input features to a logistic regression algorithm which labeled the features as accident or not. Yun et al. (Yun et al. 2014) also used optical flow to extract a motion interaction field (MIF) in the traffic scene. The MIP modeled the interaction of moving blobs in the motion field. The computed interactions were then used to calculate a temporal abnormality score and a traffic accident was flagged if the score surpassed a pre-set threshold. Ullah et al (Ullah et al. 2015) extracted the Farneback dense optical flow motion field. They used the Enthalpy Model to compute traffic motion in the scene. Smoothed Particles Hydrodynamics was applied on the calculated traffic motion to detect accidents. Li et al. (Li, Liu, and Huang 2016) partition their input traffic videos into spatiotemporal blocks. Next, SIFT features (Ng and Henikoff 2003) were computed for each block and then transformed into a category number of blocks (CNB) feature space. The extracted CNB features were used to train a Gaussian model to recognize traffic anomalies. Topic modeling was used by Ahmadi et al (Ahmadi, Tabandeh, and Gholampour 2016) to model normal traffic flow based on Lucas-Kanade optical flow vectors. The extracted vectors were indexed in a document of words to represent normal velocities in each range of vector orientation. Abnormal behavior was detected in an input frame when the extracted words were different from the normal traffic flow document model. Chen et al (Chen, Yu, and Li 2016) used optical flow to calculate a Scale Invariant Feature Transform (SIFT)-like histogram of features. They used bag of features (BOF) to encode their histogram into a latent feature space. A supervised extreme learning machine was then trained on this feature space and subsequently used to detect accidents. Yuan et

al. (Yuan, Wang, and Wang 2016) applied optical flow to their traffic video stream and used it to build 2 dictionaries that represent normal motion flow: one for magnitudes and the other for orientations. They used a Bayesian integration model to construct an anomaly map between the input stream and the normal traffic flow model. Farneback optical flow was also used by Maaloul et al. (Maaloul et al. 2017) to detect accidents in traffic videos. They filtered out noisy motion flow vectors caused by wind or camera shaking. The remaining vectors were used to model normal flow behavior based on vector orientations. Finally, dynamic thresholding of the optical flow vectors was used to detect accidents in traffic videos. Vu et al. (Vu and Pham 2017) used a convolutional neural network to detect lane-wise incidents based on manually marked zones in the traffic camera scene. SIFT features were extracted from traffic video frames by Xia et al. (Xia, Hu, and Wang 2018). A fisher kernel was used to extract trajectories from the extracted SIFT features. Afterwards, a sparse topic model was used to build a dictionary of words that represent normal trajectories. An anomaly was flagged in the evaluation phase if anomalous words were detected in the extracted features. Veni et al. (Veni, Anand, and Santosh 2020) calculated the dispersion of vectors in the optical flow motion field and used thresholds to detect traffic accidents. Finally, Kim et al. (Kim, Park, and Paik 2020) trained a 3D resnets model (He et al. 2016), which is capable of processing multiple frames simultaneously, to detect traffic accidents from the traffic video stream.

2.3.2.2 Object-Based Approach

Tracking and detection algorithms are the cornerstones of object-based traffic incident detection. Object detection is the long-standing computer vision problem of identifying, localizing and classifying objects of interest in an image (Zhao, Zheng, et al. 2019). In the past few years, there has been a significant improvement in the performance of object detection and classification

in images thanks to advances in deep learning algorithms, specifically, convolutional neural networks (CNNs) (Krizhevsky, Sutskever, and Hinton 2012; Simonyan and Zisserman 2014; He et al. 2016; Szegedy et al. 2015). These developments are illustrated in the results of the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) (Russakovsky et al. 2015).

2.3.2.2.1 Object Detection

Among the first algorithms to tackle CNN-based object detection was Girshick et al. (Girshick et al. 2014). Region-based Convolutional Neural Network (R-CNN) was introduced. R-CNN provides a 3-stage solution to object detection: 1) *region proposal generation*, where selective search (Uijlings et al. 2013) was used to generate about 2000 object bounding boxes. 2) *CNN-based feature extraction*, where the VGG16 network architecture (Simonyan and Zisserman 2014) was used to transform each object region proposal to a set of features. Lastly, 3) *classification and localization*, where each set of features were classified using an SVM classifier. Next, the bounding boxes were adjusted using bounding box regression and filtered using non-max suppression (NMS). To improve classification accuracy, supervised pre-training was conducted using the ILSVRC auxiliary dataset (Russakovsky et al. 2015). R-CNN was improved and later reintroduced as Fast R-CNN (Girshick 2015). In Fast R-CNN, the computation time and space efficiency of the network were improved through a multitask loss function, which evaluated object classification and bounding box regression simultaneously. To improve region proposal generation, Ren et al. (Ren et al. 2015) replaced selective search in stage 1 of R-CNN with a region proposal network (RPN). RPN is a feed-forward convolutional network (FCN) which takes an image as an input and generates a set of rectangular proposal regions. Mask R-CNN was introduced to provide higher-fidelity object detection (He et al. 2017). Given the Faster R-CNN network architecture, Mask R-CNN adds a branch to generate pixel-wise binary object segmentation masks

along with the generated bounding boxes.

The R-CNN family of networks separated the bounding box proposal stage from the classification and bounding box adjustment stage. In contrast, Dai et al. (Dai et al. 2016) proposed Region-Based Fully Convolutional Network (R-FCN), a fully convolutional network based on ResNet-101 (He et al. 2016) which combines both stages. They demonstrated state-of-the-art results on the Microsoft COCO dataset (Lin et al. 2014). Lin et al. (Lin et al. 2017) utilized the underlying architecture of CNNs and used pooled layers to generate a Feature Pyramid Network (FPN). The down-sampled CNN layers in the forward pass were combined with the corresponding up-sampled layers in the backwards pass to improve scale invariance in object detection in a cost-effective manner. Single Shot Multibox Detector (SSD) was introduced by Liu et al. (Liu et al. 2016). SSD augmented the VGG-16 architecture with several layers at the end. These layers were responsible for predicting the offsets and aspect ratios of default bounding boxes.

The You Only Look Once (YOLO) object detection algorithm was proposed by Redmon et al. (Redmon et al. 2016) for real-time object detection. YOLO divides each input image into an S*S grid and calculates B bounding box detections per grid. For each bounding box, an object detection confidence score, center coordinates x and y, width, and height are calculated. Moreover, class probabilities are calculated for each target class C. YOLOv2 (Redmon and Farhadi 2017) was introduced as an improvement over the original YOLO algorithm. It adopted batch normalization layers after each convolutional layer. Furthermore, predefined bounding boxes, called anchor boxes, were utilized. Another improvement over YOLO was introduced in YOLOv3 (Redmon and Farhadi 2018). YOLOv3 provided more detailed object classification by adopting multi-class labels. Furthermore, it provided better performance with smaller objects in an image by utilizing FPN-like feature pyramids. Yet another improvement over YOLO was proposed by

Bochkovskiy et al. (Bochkovskiy, Wang, and Liao 2020) and dubbed YOLOv4. YOLOv4 utilized techniques such as Weighted Residual Connections (WRC), Cross Stage Partial connections (CSP), and Cross Mini-Batch Normalization (CmBN). YOLOv4 achieved state-of-the-art detection speed and accuracy on the Microsoft COCO dataset (Lin et al. 2014).

2.3.2.2.2 Object Tracking

Tracking-by-detection is essential for object-based traffic incident detection. Multi-object Tracking (MOT) is often used. Recently, Bewly et al. (Bewley et al. 2016) proposed the simple online real-time tracking (SORT) algorithm. SORT depended on tracking bounding boxes generated from object detection algorithms. It assumed constant velocity across frames and used Kalman Filters (Kalman 1960) to compute each detected object's state. To associate bounding boxes across consecutive frames, the Hungarian algorithm (Kuhn 1955) was utilized to match bounding boxes IoUs. Wojke et al. (Wojke, Bewley, and Paulus 2017) introduced deepSORT, which follows the same procedure as (Bewley et al. 2016) but instead of matching by bounding box, object association is computed using deep feature matching.

2.3.2.2.3 Object-Based Incident Detection

Hui et al. (Hui et al. 2014) proposed a method based on Gaussian Mixture Model to detect vehicles in the video frames and applied the Mean Shift algorithm to track the detected vehicles and extract their position, acceleration and direction. The extracted trajectories are used to calculate an accident score. An accident is detected if the score surpasses a pre-set threshold. Yu et al. (Yu et al. 2018) proposed a traffic danger recognition model based on predicted camera trajectories. They used Mask R-CNN and deepSORT to detect vehicles and track their trajectories. They used camera calibration techniques to transform the input 2D image to 3D coordinates and hence obtain an estimation of vehicle positions and speed on the road. Vehicles were assumed to

follow the projected trajectory based on speed and orientation. A danger heat map was produced based on vehicle proximities, given that the vehicles follows their projected paths. Chakraborty et al. (Chakraborty, Sharma, and Hegde 2018) used YOLOv3 and SORT to detect vehicles on the road and extract their trajectories. They use trajectory subsampling to transform the extracted variable length trajectories into a fixed-dimensional feature vector then used Principal Component Analysis (PCA) to further reduce the dimensionality of their descriptors. They trained a Contrastive Pessimistic Likelihood Estimation (CPLE), a semi-supervised learning technique, and Linear Discriminant Analysis (LDA) as their objective function. The classifier was trained to classify normal versus incident trajectories. Ijjina et al. (Ijjina et al. 2019) utilized Mask R-CNN to detect vehicles and Centroid Tracking algorithm to determine their trajectories. They detected candidate traffic accidents by computing the overlap in vehicle bounding boxes. For each candidate traffic accident, the scaled vehicle accelerations were calculated by dividing the difference in pixels between the vehicle centroids across its trajectory by the detected vehicle height in pixels. The angles between vehicle trajectories and the angles of rotation of each vehicle around its central vertical axis were also calculated. An accident score was computed using a weighted combination of the features mentioned above. An accident was flagged when the score surpassed a preset threshold. Wang et al. (Wang et al. 2020) extended the detection capability of a YOLOv3 object detector by training it to recognized fallen pedestrians, fallen bicycles and rolled over vehicles. They also calculated the intersection over union (IoU) of each detected pedestrian, bicycle and vehicle to determine whether they are stationary. A decision tree was trained to detect traffic accidents by using the number of stationary road users, the amount of time they spent stationary, the number of fallen pedestrians, and the number of rolled over vehicles.

2.3.2.3 Hybrid Approach

Hybrid approaches combine features from pixel-based techniques and object-based methodology. Ren et al. (Ren et al. 2016) used background subtraction to separate vehicle pixel clusters from pavement pixels. The trajectories of the resulting clusters were tracked using a Kalman-Filter-based tracker. The trajectories were divided into spatiotemporal cells based a manually set length and time period. The grouped trajectories from each cell were used as a feature vector to train an SVM classifier. The classifier was trained to detect traffic incidents based on trajectories. Arceda et al. (Arceda and Riveros 2018) use YOLOv3 and correlation filter tracking to identify video segments that contain moving vehicles. They use violent optical flow (ViF) where the magnitude values are set to 1 if they are greater than a preset threshold and 0 otherwise. The resulting binarized magnitudes are separated into a histogram based on their direction. The histogram is then used to train an SVM classifier to classify normal traffic trajectories and crashes. Singh et al. (Singh and Mohan 2018) combined 2 methodologies to detect traffic accidents. Firstly, they divided their input video into spatiotemporal volumes and trained a stacked autoencoder using normal traffic videos. At testing time, they used the model reconstruction error to generate an abnormality score. Next, they use background subtraction and blob tracking to identify the trajectories of moving objects on the road. They calculate a collision score based on the number of intersecting trajectories which are followed by an interruption in trajectory flow. A one-class SVM was trained to detect outliers based on the intermediate representation of the autoencoder, the autoencoder reconstruction error, and the collision score. An accident is detected if the outlier score from the SVM exceeds a preset threshold.

Since 2018, NVIDIA has published a traffic anomaly detection challenge as one of the

tracks of its yearly AI City Challenge (Naphade et al. 2018; Naphade et al. 2019; Naphade et al. 2020; Naphade et al. 2021). To handle the variance between traffic incident videos, and to deal with the low-resolution of the video input, top-scoring research efforts across the different competition editions relied on the deduction that traffic anomalies lead to stationary vehicles in areas where traffic is expected to flow for prolonged periods of time (Zhao et al. 2021; Doshi and Yilmaz 2021; Aboah 2021; Li et al. 2020; Shine and CV 2020; Doshi and Yilmaz 2020; Biradar et al. 2019; Wang et al. 2019; Bai et al. 2019; Xu et al. 2018; Wei et al. 2018; Wu et al. 2021; Chen et al. 2021). Xu et al (Xu et al. 2018) operated under the assumption that every traffic anomaly leads to stationary vehicles on the road for a prolonged period of time. Additionally, stationary vehicles in an uninterrupted flow scene blended into the video background. Based on these assumptions, they devised a method to construct a video background based on moving window averaging across frames. Furthermore, they measured vehicle stopping time to eliminate normal stationary vehicles (like vehicles in a red light) from abnormality contention. In addition, anomalies were detected using vehicle trajectories. Mask R-CNN was used to get each vehicle segmentation mask. Vehicle trajectories were obtained using optical flow of pixels in each vehicle mask. The velocities of normal-flow vehicles were obtained in pixels/frame and velocities that were identified as outliers based on the inter-quartile range were flagged as anomalous. Xu et al. were the winners of the NVIDIA AI City Challenge 2018 anomaly detection track. Wang et al (Wang et al. 2019) used Gaussian Mixture Models to estimate background pixels and the foreground mask in the video stream. Consecutive foreground masks were accumulated to create a region of interest. If a vehicle was detected in the background pixels using the YOLOv3 detector, it was flagged as anomalous. Furthermore, a TrackletNet Tracker (TNT) was used to obtain the trajectories of stationary vehicles and obtain the starting time of the traffic anomaly event. Bai et

al. (Bai et al. 2019) utilized averaging-based background detection to identify stationary vehicles on the road. Furthermore, they detected all vehicles on the road using Faster R-CNN and overlayed the results of subsequent frames to obtain a heatmap of motion. The heatmap was used to identify the main road and eliminate the effect of stationary vehicles on auxiliary roads. Perspective normalization was utilized to enhance the detection of vehicles that were farther away from the camera. A Spatio-temporal pixel-tracking matrix was used to track stationary vehicles and signal a traffic anomaly. Bai et al. ranked first in the NVIDIA AI City Challenge 2019. Doshi et al. (Doshi and Yilmaz 2020) also leveraged averaging-based background modeling and overlayed detected objects to get the road heatmap. They used a YOLOv3 object detector and specified a low confidence score to detect small vehicles in the video frame. To eliminate false positives, the distance between each object centroid across frames was computed and only objects within a distance threshold were considered in the anomaly detection algorithm. Li et al (Li et al. 2020) proposed an anomaly detection algorithm using box-level tracking and pixel-level tracking. They first identify a main road mask using vehicle trajectories extracted by Faster R-CNN and deepSORT. A forward MOG2 background model was utilized to detect stationary vehicles on the road. Additionally, a finer backwards MOG2 background model was utilized to identify the starting time of the anomaly. Afterwards, Faster R-CNN was applied once more to identify the bounding boxes of stationary vehicles. The presence of a stationary vehicle was confirmed by matching the bounding box IoUs across frames. Li et al. ranked first in the NVIDIA AI City Challenge 2020. Aboah et al. (Aboah 2021) automatically detected road type (freeway vs. intersection), time of day (day vs. night), and weather conditions. Road type was detected by observing vehicle trajectory directions while time of day and weather were detected using the distribution of pixel intensities. Next, background scene estimation was performed by randomly

sampling video frames from each scene and extracting a road mask based on pixel intensities. Finally, they identified stationary vehicles in the background scene within the road mask and perform box level IoU tracking to identify traffic anomalies.

While numerous approaches were proposed in the literature for video-based traffic incident detection, none of best-performing methodologies are suitable for real-time traffic incident detection with low-resolution traffic cameras. Optical flow methods are best suited to detect high impact traffic incidents that would cause a notable disturbance in the video motion field such as crashes. Low impact incidents, such as stalled or abandoned vehicles can have an adverse effect on safety and operations and are therefore important to detect as well. Top-scoring teams that solved the NVIDIA AI City Challenge – Traffic Anomaly Detection track participants proved that their approaches could detect traffic incidents from roadside traffic CCTV cameras with high accuracy. Nonetheless, the challenge did not require participants to run their algorithms in real-time, and therefore no restriction on computation speed were placed. In fact, top-scoring teams applied a variety of post-processing techniques to optimize their detection performance such as global object matching across the input video (Li et al. 2020), post-processed video stabilization (Zhao et al. 2021), and cross-video frame sampling for background extraction (Aboah 2021).

2.3.3 Summary

Traffic incident detection can be characterized as a video-based anomaly detection problem. Abnormality detection in videos has long been studied in the literature. However, the definition of anomaly is too broad. Generic anomaly recognition algorithms lack the nuance required to solve the more fine-grained problem of traffic anomaly detection. Traffic anomaly detection was modeled by first determining normal traffic behavior using a set of features and then identifying instances that deviated from the modeled normalcy. Numerous methods have been

proposed to detect anomalies in traffic videos. Those methods can be broadly classified into pixel-based approaches, object-based approaches, and hybrid approaches. Pixel-based approaches depend on dense pixel-wise feature extractors such as optical flow algorithms. The extracted pixel-based features were often transformed using another feature generator like Histogram of Flow Gradients or Topic Models. Optical flow features are better suited for detecting severe changes in the motion flow on the road. Hence, they have been mainly utilized for traffic crash detection. As a result of the recent advancement in deep learning-based object detection, object-based approaches have increased in popularity. Object-based approaches depend on the detection and tracking of different road users. The extracted trajectories were often combined with methods to estimate speed, acceleration, and/or proximity between road users. Hybrid approaches utilized pixel-based features and object-based features. Pixel-based methods such as background subtraction, autoencoders, or optical flow, were used in tandem with object-based features such as heatmaps created from vehicle tracking and stationary vehicle detection. State-of-the-art video-based traffic anomaly detection algorithms were built using hybrid techniques. A summary of traffic incident methodologies is provided in

Table 2.1. Despite the recent advances in traffic incidents detection, traffic incident classification has not gained the same amount of attention. Some research efforts proposed incident classification approaches based on traffic counts or police records, but very few have attempted video-based traffic incident classification.

Table 2.1 Summary of video-based traffic incident detection literature

Reference	Data Source	Features	Classifier
Pixel-Based Approaches			
(Sadeky et al. 2010)	Highway + Intersection CCTVs	Lucas-Kanade optical flow + Histogram of Flow Gradients	Logistic Regression

Reference	Data Source	Features	Classifier
(Yun et al. 2014)	Intersection CCTVs	Optical Flow + Motion Interaction Field	Static Thresholds
(Ullah et al. 2015)	Highway + Intersection CCTVs	Farneback Optical Flow + Enthalpy model	Smoothed Particle Hydrodynamics
(Li, Liu, and Huang 2016)	Highway + Intersection CCTVs	SIFT + Category Number of Blocks	Gaussian Distribution Model
(Ahmadi, Tabandeh, and Gholampour 2016)	Intersection CCTVs	Lucas-Kanade Optical Flow + Topic Models	Topic Model Document Matching
(Chen, Yu, and Li 2016)	Highway + Intersection CCTVs	Optical Flow + Histogram of Flow Gradients	Extreme Learning Machine
(Yuan, Wang, and Wang 2016)	Vehicle On-board Camera	Optical flow + Topic Models	Topic Model Document Matching
(Maaloul et al. 2017)	Highway CCTVs	Farneback Optical Flow + Histogram of Flow Gradients	Dynamic Thresholds
(Vu and Pham 2017)	Highway CCTVs	Manual Lane Annotation + CNN	Softmax Layer
(Xia, Hu, and Wang 2018)	Intersection CCTVs	SIFT + Fisher Kernel + Topic Models	Topic Model Document Matching
(Veni, Anand, and Santosh 2020)	Intersection CCTVs	Optical Flow + Vector Dispersion	Static Thresholds
(Kim, Park, and Paik 2020)	Highway + Intersection CCTVs	3D-CNN	Softmax Layer
Object-Based Approaches			
(Hui et al. 2014)	Vehicle On-board Camera	GMM + Mean Shift	Static Thresholds
(Yu et al. 2018)	Highway CCTVs	Mask R-CNN + deepSORT + Camera Calibration	Static Thresholds
(Chakraborty, Sharma, and Hegde 2018)	Highway CCTVs	YOLOv3 + deepSORT + PCA	Contrastive Pessimistic Likelihood Estimation
(Ijjina et al. 2019)	Highway + Intersection CCTVs	Mask R-CNN + Centroid Tracking + scaled acceleration	Static Thresholds
(Wang et al. 2020)	Intersection CCTVs	YOLOv3 + manual rules	Decision Tree
Hybrid Approaches			
(Ren et al. 2016)	Highway + Intersection CCTVs	Background Subtraction + Cluster Tracking	SVM
(Xu et al. 2018)	Highway + Intersection CCTVs	Background Detection + Mask R-CNN + optical flow trajectories	Static Thresholds

Reference	Data Source	Features	Classifier
(Arceda and Riveros 2018)	Highway + Intersection CCTVs	YOLOv3 + Violent Optical Flow + Histogram of Flow Gradients	SVM
(Singh and Mohan 2018)	Highway + Intersection CCTVs	Autoencoders + Background Subtraction + Cluster Tracking	One-Class SVM
(Bai et al. 2019)	Highway + Intersection CCTVs	Background Detection + Motion Heatmap + Faster R-CNN	Stationary Vehicle Detection
(Wang et al. 2019)	Highway + Intersection CCTVs	Background Detection + Motion Heatmap + YOLOv3	Stationary Vehicle Detection
(Doshi and Yilmaz 2020)	Highway + Intersection CCTVs	Background Detection + Motion Heatmap + YOLOv3	Stationary Vehicle Detection
(Li et al. 2020)	Highway + Intersection CCTVs	Background Detection + Motion Heatmap + Faster R-CNN + deepSORT	Stationary Vehicle Detection
(Aboah 2021)	Highway + Intersection CCTVs	Background Detection + Road/Weather Type + YOLOv5	Decision Tree + Stationary Vehicle Detection

Numerous approaches were proposed in the literature for video-based traffic incident detection. However, none of best-performing methodologies are suitable for real-time traffic incident detection with low-resolution traffic cameras. Optical flow methods are best suited to detect high impact traffic incidents that would cause a notable disturbance in the video motion field such as crashes. Low impact incidents, such as stalled or abandoned vehicles can have an adverse effect on safety and operations and are therefore important to detect as well. Top-scoring teams that solved the NVIDIA AI City Challenge – Traffic Anomaly Detection track participants proved that their approaches could detect traffic incidents from roadside traffic CCTV cameras with high accuracy. Nonetheless, the challenge did not require participants to run their algorithms in real-time, and therefore no restriction on computation speed were placed. In fact, top-scoring teams applied a variety of post-processing techniques to optimize their detection performance such as global object matching across the input video (Li et al. 2020), post-processed video stabilization

(Zhao et al. 2021), and cross-video frame sampling for background extraction (Aboah 2021). In the third research objective of this dissertation, a video-based traffic incident identification algorithm was proposed with a focus on real-time applicability. At any given frame at inference time, the proposed method does not make any computations based on future frames. Moreover, the algorithm computation time was evaluated in order to prove the applicability of the method on real-time traffic video feeds. In addition, the real-time delay of the detection time of traffic incidents was assessed.

CHAPTER 3: UTILIZING ATTENTION-BASED MULTI-ENCODER-DECODER NEURAL NETWORKS FOR FREEWAY TRAFFIC SPEED PREDICTION

3.1 Introduction

Traffic speed prediction is an essential component of Intelligent Transportation Systems (ITS). It contributes towards crucial transportation applications like navigation guidance, traffic scheduling and traffic management (Zhang et al. 2011). Traffic speed prediction is a difficult problem since it is necessary to consider both the temporal dependency between traffic parameters and the spatial connection between traffic parameters at different parts of the road network. Furthermore, this spatiotemporal relationship is stochastic and highly non-linear in nature, making it difficult to model accurately (Park et al. 2011).

The ubiquity of traffic sensors on the road network has led to an explosion of traffic data, which has in turn fueled a growing body of research in data-driven modeling of the speed prediction problem (Vlahogianni, Karlaftis, and Golias 2014). These types of models can be classified into two types: classical statistical models and machine intelligence models (Van Lint and Van Hinsbergen 2012). Statistical models, specifically parametric models, are favored for their interpretability and relative simplicity. On the other hand, machine intelligence models are non-parametric models which make little to no assumptions regarding the input variables. Hence, they can model strong generalizations in pattern recognition. Due to the advancement in computation space and time efficiency, Artificial Neural Networks (ANNs), specifically deep learning techniques, have been widely applied in traffic speed prediction during the past few years. They have since demonstrated superior performance compared to classical statistical models and

machine learning algorithms.

Despite the recent achievements in deep learning-based traffic speed prediction, there are some major limitations in the literature. Firstly, most research efforts focus on a short-term input sequence, like data from the past minutes/hours, when in fact, traffic speed exhibits cyclic behavior across consecutive days (specifically weekdays). In addition, there's a periodic correlation between traffic patterns during the same day-of-week across successive weeks (Wu, Ho, and Lee 2004; Liu et al. 2017; Cao, Li, and Chan 2020). Failing to account for long-term sequences inhibits the deep learning model's ability to generalize traffic patterns well and can limit prediction accuracy. Furthermore, most methods in the literature focus on 5-15 minute ahead speed prediction, which provides a short window for reaction to predicted traffic turbulence, and by extension, a limited practical advantage to potential users.

Secondly, most research efforts treat their deep learning models as "black box" models and thus fail to understand the reasons behind their model's prediction processes and results (Tang et al. 2017; Yu, Lee, and Sohn 2020). The lack of interpretability leads to several problems. For instance, it might discourage decision makers from deploying the model due to the absence of output explicability. Furthermore, the scarce interpretability impedes the model development, debugging and dissection of where performance deficiencies may be coming from.

In this research, an attention-based multi-encoder-decoder model, dubbed Att-MED, is proposed to model and predict freeway traffic speed for up to a 60-minute horizon. The encoder component uses convolutions to model the spatial dependency, and Long-Term-Short-Memory (LSTM) to model the temporal dependency. Additionally, Att-MED can incorporate multiple encoders and is therefore able to extract traffic patterns from multiple input sequences with different periodicity. The decoder also uses an LSTM to model the output horizon sequentially.

Furthermore, attention mechanism is used as an intermediate component between the encoder and decoder to enhance the temporal context of the decoder. Moreover, the attention weights are visualized to help decipher the neural network's prediction process. The proposed model is trained and tested on traffic data from January 1st, 2017, to June 30th, 2017, extracted from State Road 408 in Orlando, Florida, USA. In summary, the contribution of this work is threefold:

1) The methodology demonstrates an attention-based multi-encoder-decoder model (Att-MED) that can encode multiple input traffic sequences. This allows the model to capture the different periodic characteristics of traffic flow, namely short-term, daily and weekly characteristics.
2) The Att-MED model was trained and tested for up to 60-minute-ahead network-wide speed prediction. The model performance was compared against baseline models. Moreover, the contribution of the attention layer towards prediction accuracy was quantified by implementing a version of the network without attention.
3) The interpretability of the proposed model was enhanced by leveraging attention mechanisms to extract and visualize temporal dependencies in speed prediction. The visualization provides a deeper understanding of how the network operates and allows for nuanced and meticulous conclusions to be drawn from the results.

3.2 Methodology

The network proposed in this research effort tackles the problem of speed prediction by utilizing 3 input sequences. First, a short-term sequence of length T ($\mathcal{X}_{t-T+1}, \dots, \mathcal{X}_{t-1}, \mathcal{X}_t$), which captures the traffic features from the previous timesteps. Next, a daily sequence of length D ($\mathcal{X}_{d-D}, \dots, \mathcal{X}_{d-2}, \mathcal{X}_{d-1}$), which lists the traffic features during the same time-of-day for the past D days. Finally, a weekly sequence of length W ($\mathcal{X}_{w-W}, \dots, \mathcal{X}_{w-2}, \mathcal{X}_{w-1}$), which captures the traffic

features during the same time-of-day in the same day-of-week for the past W weeks. Each element $\mathcal{X}$ in the input sequence contains the traffic features of all road segments in the freeway network. The model forecasts traffic speed for the next M timesteps ($\mathcal{Y}_{t+1}, \mathcal{Y}_{t+2}, \ldots, \mathcal{Y}_{t+M}$). Each element $\mathcal{Y}$ in the output sequence contains the network-wide traffic speed predictions (i.e., all locations in the freeway network). The proposed network consists of 3 main components: encoder, attention layer and decoder.

3.2.1 Encoder

The encoder combines 2 neural network algorithms: convolution is used to extract the spatial features between traffic detectors and LSTM is used to model the temporal dependency. The two modules are combined into one network architecture called Conv-LSTM. A Conv-LSTM accepts a 4-dimensional input of the following structure: (time sequence, features, width, length). Time sequence is the temporal input sequence dimension. The feature vector contains the traffic parameters pertaining to each specific location in the network at a given time in the sequence. Finally, the width and length dimensions capture the spatial characteristics of the freeway network. To model the last 2 dimensions, the traffic parameters at each timestep have been reshaped into a vector of size (width, length) where width = 1 and length = number of road segments in the freeway network.

The convolution module focuses on extracting the spatial features from the road network and hence computes the features from each time slice separately. The module accepts a 3D tensor in $\mathbb{R}^{F \times W \times \mathrm{L}}$ as input where the tensor represents features, width, and length. The temporal feature extraction, handled by the LSTM module, involves computing cell outputs $\mathcal{C}_1 \ldots \mathcal{C}_T$ and hidden states $\mathcal{H}_1 \ldots \mathcal{H}_T$ for each 3D tensor in the input sequence $\mathcal{X}_1 \ldots \mathcal{X}_T$. At each time step, the LSTM gates i_t, f_t and o_t decide how much information should be carried over from the previous cell

output $\mathcal{C}_{t-1}$ and hidden state $\mathcal{H}_{t-1}$ and how much information can be drawn from the current input $\mathcal{X}_t$. Weight matrices W and bias vectors b are the Conv-LSTM trainable parameters, the values of which are determined by the backpropagation algorithm (Hecht-Nielsen 1992). In summary, the Conv-LSTM is governed by the following set of equations, where $\sigma(\cdot)$ is the sigmoid function, $*$ denotes a convolution operation, and $\diamond$ computes the Hadamard product (also known as the element-wise product).

$$i_t = \sigma(W_{xi} * \mathcal{X}_t + W_{hi} * \mathcal{H}_{t-1} + W_{ci} \diamond \mathcal{C}_{t-1} + b_i) \quad (3.1)$$

$$f_t = \sigma(W_{xf} * \mathcal{X}_t + W_{hf} * \mathcal{H}_{t-1} + W_{cf} \diamond \mathcal{C}_{t-1} + b_f) \quad (3.2)$$

$$\mathcal{C}_t = f_t \diamond \mathcal{C}_{t-1} + i_t \diamond tanh(W_{xc} * \mathcal{X}_t + W_{hc} \diamond \mathcal{H}_{t-1} + b_c) \quad (3.3)$$

$$o_t = \sigma(W_{xo} * \mathcal{X}_t + W_{ho} * \mathcal{H}_{t-1} + W_{co} \diamond \mathcal{C}_t + b_o) \quad (3.4)$$

$$\mathcal{H}_t = o_t \diamond tanh(\mathcal{C}_t) \quad (3.5)$$

To decrease the dimensionality of the Convolutional-LSTM output at each time step, the output is flattened and then passed to several fully connected (FC) layers with ReLU activation. The encoder network produces a sequential output $\mathcal{E}_1 \dots \mathcal{E}_T$ as shown in the equations bellow. The full, unrolled encoder network structure is demonstrated in Figure 3.1.

$$\mathrm{FC(1)} = ReLU(\mathrm{W}_{1t}\mathcal{H}_t + b_{1t}) \quad (3.6)$$

$$\mathrm{FC(N)} = ReLU(\mathrm{W}_{Nt}\mathrm{FC}(N-1) + b_{Nt}) \quad (3.7)$$

$$\mathcal{E}_t = FC(N) \quad (3.8)$$

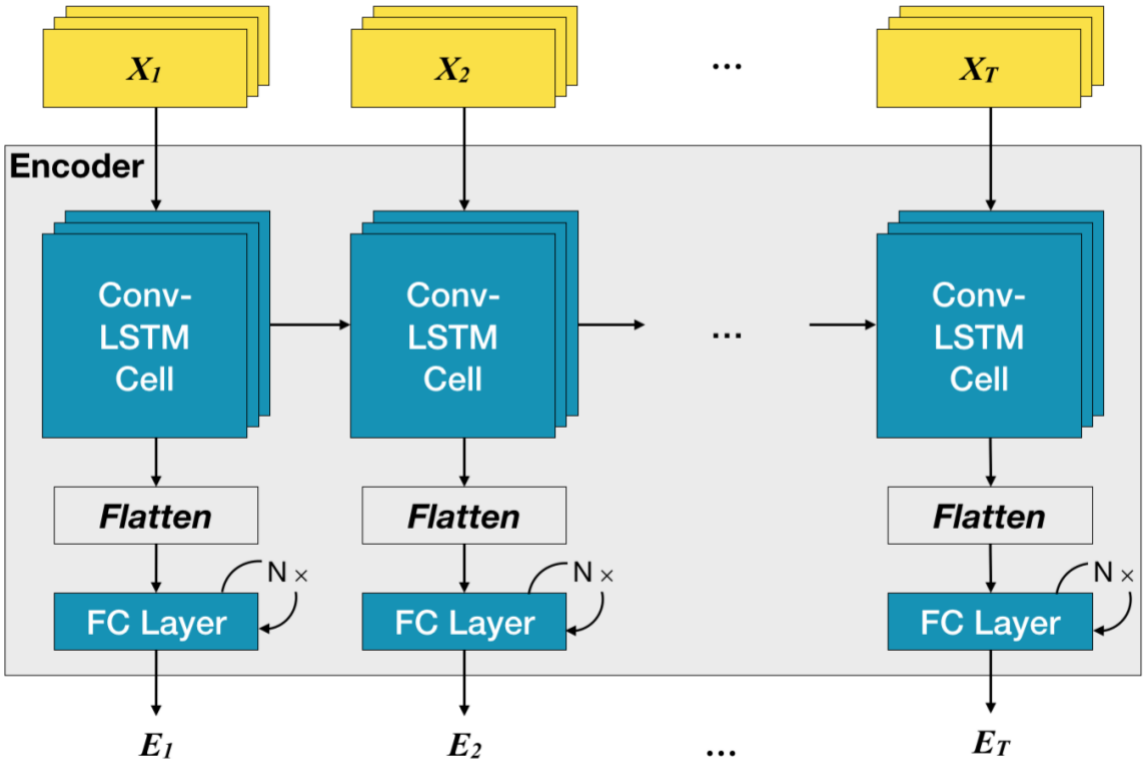


Figure 3.1 Encoder Neural Network Architecture

3.2.2 Attention Layer

The attention mechanism enhances the encoder-decoder network by allowing each step of the decoder to assign attention weights to the input sequence. The more relevant an encoder step in the sequence is, the higher the weight of the assigned attention. The proposed network utilizes the attention mechanism to compute the relevance of short-term, daily and weekly periodicity when predicting each output horizon in the output sequence.

To compute the decoder input at timestep t, the attention energy of the encoder output at timestep i, denoted as e_{ti}, is calculated as a function of the previous decoder hidden state $\mathcal{S}_{t-1}$ and the output of the encoder at timestep i, denoted as $\mathcal{E}_i$. The attention energies of all the encoder outputs are calculated and subsequently passed through a softmax function to create attention

weights α_{ti}. The softmax function ensures that the aggregate of the assigned attention weights sums to 1. Lastly, a single vector is calculated for the output attention at time t. It is computed as the dot product of the attention weights and the encoder outputs. The final attention layer output, called the context vector, is passed as input to the decoder layer. In essence, the attention weights are computed according to the equations below, where weight matrices W and bias vector b are the attention layer trainable parameters. The $\cdot$ operator calculates matrix multiplication and the $\exp(\cdot)$ function computes the exponent.

$$e_{ti} = W_{ti} \cdot \tanh(W_{st} \cdot \mathcal{S}_{t-1} + \mathrm{W}_{ei} \cdot \mathcal{E}_i) + b_{ti} \tag{3.9}$$

$$\alpha_{ti} = \frac{\exp(e_{ti})}{\sum_{k=1}^{K} \exp(e_{tk})} \tag{3.10}$$

$$CXT_t = \sum_{i=1}^{I} \alpha_{ti} \mathcal{E}_{ti} \tag{3.11}$$

3.2.3 Decoder

The decoder layer consists of an LSTM neural network which computes the output sequence $\mathcal{Y}_1 \ldots \mathcal{Y}_T$. The network uses the context vector CXT_t as input at each output timestep t. The LSTM computes its internal hidden state $\mathcal{S}_t$ and sends it to the attention unit of the next timestep. The i_t, f_t and o_t LSTM gates decide how much information to use from the context vector and how much to use from the previous decoder timesteps. The output of each decoder cell $\mathcal{Y}_t$ is a vector in $\mathbb{R}^N$ where N is the number of detectors on the road network. This enables the LSTM to predict network-wide traffic speed at each output timestep.

$$i_t = \sigma(W_{ix} \cdot CXT_t + W_{is} \cdot \mathcal{S}_{t-1} + W_{ic} \cdot c_{t-1} + b_i) \tag{3.12}$$

$$f_t = \sigma(W_{fx} \cdot CXT_t + W_{fs} \cdot \mathcal{S}_{t-1} + W_{fc} \cdot c_{t-1} + b_f) \tag{3.13}$$

$$c_t = f_t \diamond c_{t-1} + i_t \diamond tanh(W_{cx} \cdot CXT_t + W_{cs} \cdot \mathcal{S}_{t-1} + b_c) \quad (3.14)$$

$$o_t = \sigma(W_{ox} \cdot CXT_t + W_{os} \cdot \mathcal{S}_{t-1} + W_{oc} \cdot c_t + b_o) \quad (3.15)$$

$$\mathcal{S}_t = o_t \diamond tanh(c_t) \quad (3.16)$$

$$\mathcal{Y}_t = W_{ys} \cdot \mathcal{S}_t + b_y \quad (3.17)$$

To summarize, the decoder computes the output sequence according to the above set of equations. As mentioned above, $\sigma(\cdot)$ is the sigmoid function, $\cdot$ denotes matrix multiplication and $\diamond$ calculates the Hadamard product. Matrices W and vectors b are the decoder trainable parameters.

3.2.4 Att-MED Network

To model the multi-input-sequence to output-sequence problem, the Att-MED model constructs 3 encoders: short-term, daily and weekly. Each encoder extracts the spatiotemporal parameters of its assigned input sequence. Next, the encoder outputs are concatenated and subsequently used as an input to the attention layer. The attention layer weighs the importance of each value in each input sequence and computes a combined context vector, which is then passed as an input to the decoder network. Each decoder LSTM cell forwards its hidden state as an input to the next attention unit. Figure 3.2 illustrates how the network computes the network-wide traffic speed as an output sequence by encoding multiple input sequences.

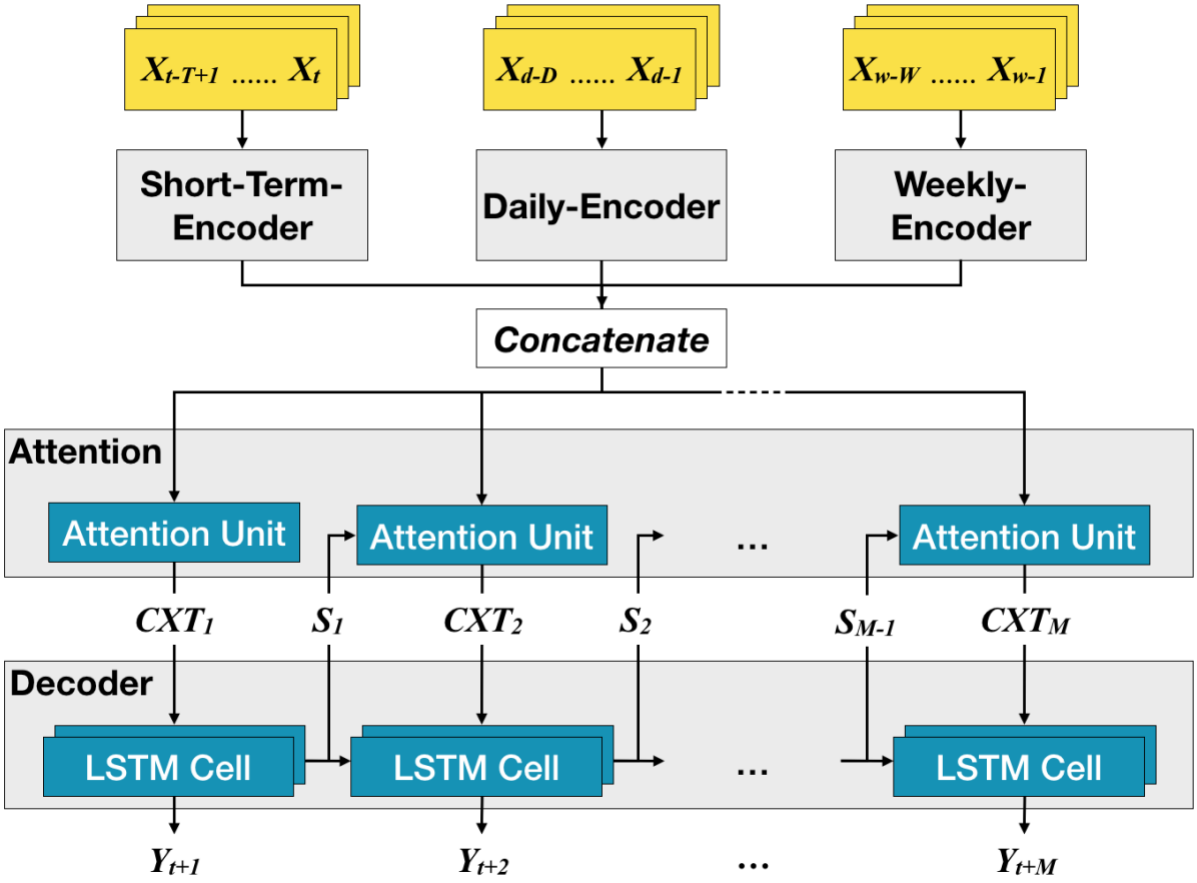


Figure 3.2 Att-MED network architecture

3.3 Experimentation

3.3.1 Data Preparation

The experiment studied traffic speed along State Road 408, an east-west freeway in Orlando, Florida, USA. Furthermore, the study focuses on the east-bound direction. There are 52 Microwave Vehicle Detection System (MVDS) sensors in the selected area, which are spaced 0.4 miles apart on average. The total length of the covered route is 20.8 miles. Fig. 3 depicts their distribution along the freeway. The microwave sensors record spot speed, total volume and average occupancy every 60 seconds. The MVDS historical data has been archived on the Regional

Integrated Transportation Information System (RITIS) and has been downloaded for this study from the RITIS website.

The time range of the historical data used for this study is January 1st, 2017, to June 30th, 2017. To take advantage of the daily traffic patterns, the study focuses on speed prediction during weekdays. Based on the Speed-Density-Flow traffic relationship, 2 of the detected traffic parameters can be used to compute the third (Hall, Allen, and Gunter 1986). Therefore, to diminish the potential irreducible error that may be caused by detector inaccuracies, only speed and volume were chosen as traffic features. Speed and volume data were aggregated to 5-minute intervals. The summary statistics of the per-segment aggregated features are presented in Table 3.1. The collected traffic dataset was split into a training set, a validation set, and a testing set into a ratio of 50%:25%:25%, respectively. The validation set was utilized in the deep learning models.

Table 3.1 Input variables summary statistics

	Min	**Max**	**Mean**	**Standard Deviation**
Speed (mph)	0.00	149.08	65.35	6.37
Volume	0.00	416.00	141.59	116.23

3.3.2 Performance Metrics

The experiment used three widely applied evaluation metrics to qualify the performance of each model. They are Mean Absolute Error (MAE), Root Mean Square Error (RMSE) and Mean Absolute Percentage Error (MAPE). In the following equations, $\hat{y}_i$ refers to a prediction made by the model while y_i refers to its corresponding ground-truth value. The total number of data points used for evaluation is denoted as n.

$$MAE = \frac{1}{n}\sum_{i=0}^{n}|y_i - \hat{y}_i| \tag{3.18}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=0}^{n}(y_i - \hat{y}_i)^2} \tag{3.19}$$

$$MAPE = \frac{100}{n}\sum_{i=0}^{n}\frac{|y_i - \hat{y}_i|}{y_i} \tag{3.20}$$

3.3.3 Baseline Models

The experiment target was to, at a given time t, predict the speed values of the next time increments up to 60 minutes. Since the processed data has been resampled as 5-minute intervals, the objective equated to predicting the speed values of the next 12 timesteps. Moreover, each MVDS detector represented a road segment in the experiment. The results of our model are compared against the following models:

- **ARIMA**: Auto-regressive integrated moving average.
- **SVR**: Support Vector Regression using a radial basis function (RBF) kernel.
- **ANN**: A deep, feed forward neural network. The hyperparameters are the number of layers and the number of units per layer.
- **Bi-LSTM**: An encoder-decoder Bidirectional LSTM network. The hyperparameters are the number of units in the encoder and decoder layers.
- **Conv-LSTM**: A sequence-to-sequence model which uses a convolutional LSTM network as an encoder and an LSTM as a decoder. The number of Conv-LSTM units, kernel size, as well as the number of LSTM units are the model hyperparameters.

As for the Att-MED model, the number of units in the Conv-LSTM encoder cells, LSTM

decoder cells as well as the number of encoder FC layers are considered hyperparameters. To include both traffic speed and volume in the input space, and to accommodate the shape of the freeway network displayed in Fig. 3, the features, width, and length input parameters were assigned sizes 2, 1, and 52, respectively. Furthermore, to enable network-wide speed detection, the size of the model's output at each prediction timestep was set to 52.

All neural network models were implemented using Tensorflow 2.2.1 (Abadi et al. 2016). The hyperparameters were tuned using the random search implementation of the Keras-Tuner package (O'Malley et al. 2019). The tuner used validation loss as a comparison metric to avoid optimizing the hyperparameters to overfit the training data. Furthermore, the Adam optimizer (Kingma and Ba 2014) was employed, and the loss function was set to MAE. The models ran for 250 epochs. Early stopping was utilized to prevent overfitting by monitoring the validation loss. It prevents overtraining by monitoring the trend of the validation loss. If the validation loss deteriorates for a certain number of epochs, the training process is halted even if the training loss continues to decrease.

3.3.4 Prediction Results

Tables Table 3.2 and Table 3.3 show the prediction results of the proposed model and baseline models on the testing dataset for the 52 road segments. Table 3.2 displays the short-term results (5min/10min/15min), while Table 3.3 focuses on longer term predictions (30min/45min/60min). The results indicate that the SVR model was not able to model the temporal dependency correctly. The ARIMA model was able to produce good short-term results, but the performance quickly deteriorated when predicting farther time horizons, indicating that it's unable to handle long-term temporal dependencies. The Att-MED, Bi- LSTM and Conv-LSTM models performed better than the basic ANN, highlighting the importance of capturing the temporal

dependency provided by the recurrent neural networks structure. Furthermore, the proposed model and the Conv-LSTM's convolution modules were able to model the spatial dependencies in the road network.

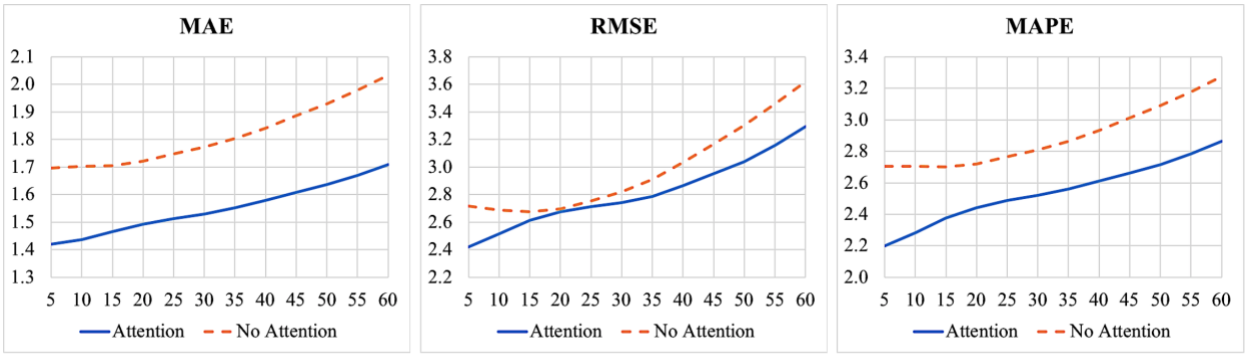
Table 3.2 Short-term speed prediction results

Model	5-minutes			10-minutes			15-minutes		
	MAE	**RMSE**	**MAPE**	**MAE**	**RMSE**	**MAPE**	**MAE**	**RMSE**	**MAPE**
ARIMA	1.580	2.946	2.721	1.716	3.419	3.030	1.953	3.814	3.315
SVR	2.610	5.369	4.075	2.609	5.262	4.077	2.613	5.269	4.081
ANN	1.634	2.927	2.794	1.847	3.420	3.153	1.871	3.635	3.220
Bi-LSTM	1.810	3.730	3.023	1.852	3.750	3.344	1.959	3.867	3.380
Conv-LSTM	1.605	2.979	2.656	1.699	3.181	2.835	1.865	3.398	2.998
Att-MED	**1.451**	**2.553**	**2.240**	**1.476**	**2.617**	**2.320**	**1.539**	**2.674**	**2.404**

Table 3.3 Long-term speed prediction results

Model	30-minutes			45-minutes			60-minutes		
	MAE	**RMSE**	**MAPE**	**MAE**	**RMSE**	**MAPE**	**MAE**	**RMSE**	**MAPE**
ARIMA	2.157	4.752	4.080	2.433	5.486	4.752	2.689	6.106	5.375
SVR	2.617	5.281	4.088	2.630	5.305	4.108	2.649	5.345	4.138
ANN	2.075	4.028	3.602	2.140	4.363	3.752	2.295	4.770	5.450
Bi-LSTM	2.044	4.210	3.567	2.154	4.472	3.751	2.254	4.684	3.949
Conv-LSTM	1.976	3.885	3.353	2.067	4.091	3.542	2.120	4.180	3.643
Att-MED	**1.582**	**2.801**	**2.548**	**1.671**	**3.034**	**2.703**	**1.777**	**3.376**	**2.899**

Att-MED generated superior results to all the baseline models for every prediction horizon. This outcome illustrates the significance of the daily and weekly traffic parameter periodicity as input in addition to the short-term periodicity. Furthermore, attention is needed to regulate the weight of each input sequence during prediction. Figure 3.3 depicts the results of the proposed model with and without the attention layer. The results indicate that the attention mechanism balanced between the input sequences to minimize the prediction error.



(a) (b) (c)

Figure 3.3 Prediction results for the proposed model with and without attention (a) MAE (b) RMSE (c) MAPE

3.3.5 Attention Visualization

Neural network models are often criticized for operating as "black boxes" and lacking interpretability. To increase model explainability, the attention weights assigned to prediction samples were extracted and visualized. Figure 3.4 (a), (b) and (c) map the attention weights of the Att-MED model when making a prediction of a morning peak sample (9:00 am), off-peak sample (1:00 pm) and nighttime sample (3:00 am), respectively. In order to avoid the potential training bias associated with the training and validation sets, the visualized attention weights were calculated for input data sampled from the testing set. The chosen values for T, D and W were 12, 5 and 2, respectively. Since attention weights pass through a softmax function, the aggregate value of each column in the plots is 1. It can be determined that the model relied on weekly periodicity while making predictions during peak hour. The model relied more on both daily and weekly cyclic characteristics to forecast the speed of off-peak samples. In contrast, the model shifted more focus toward short-term traffic features to make predictions during nighttime.

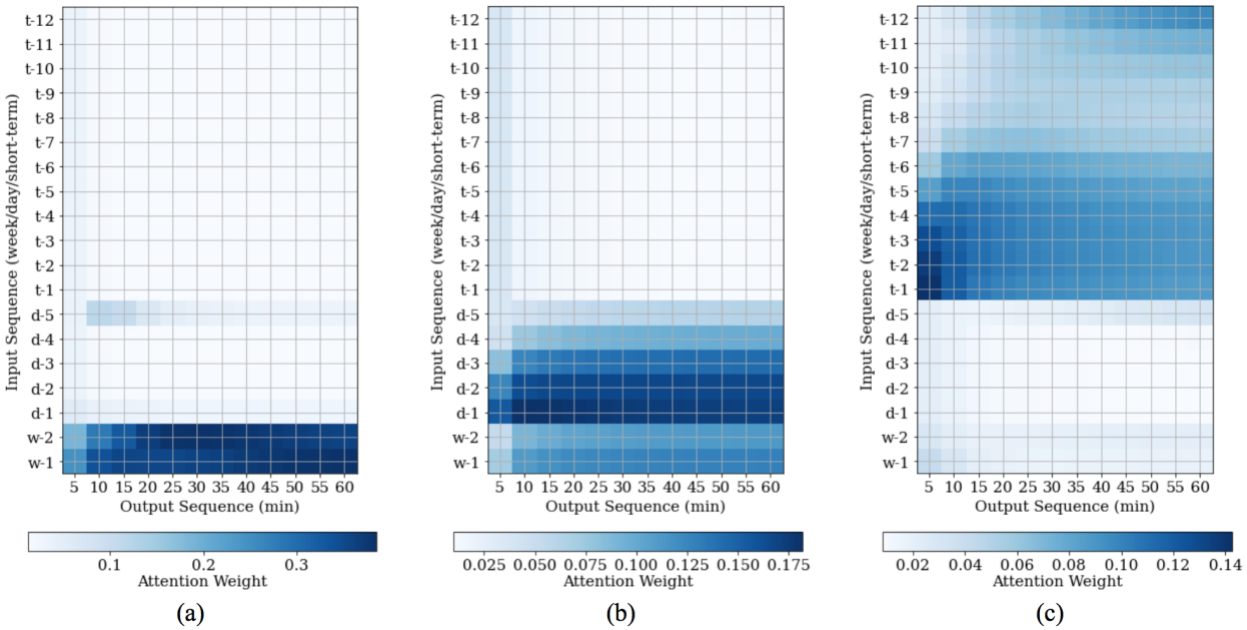


Figure 3.4 Attention weights of proposed model predictions during (a) morning peak hour sample (9 am) (b) off-peak hour sample (1 pm) (c) nighttime sample (3 am)

Figure 3.5 illustrates a sample of daily and weekly traffic speed patterns on State Road 408 milepost 09.6. Figure 3.5 (a) indicates that, during the morning peak, Friday and Thursday patterns differ slightly from the rest of the weekdays. On the other hand, weekly peak traffic rhythm seems to be closer as demonstrated by Figure 3.5 (b). This pattern validates the model's choice to focus on weekly traffic patterns when making predictions during peak hours. The same can be said for nighttime traffic, which exhibits high variability across both daily and weekly cycles. The model chose to focus the attention weights on short-term features due lack of strong cyclic traffic patterns during nighttime. As for off-peak hours, specifically between the morning and evening peak, Fig. 6 indicates that traffic speed exhibits a similarity across daily and weekly patterns. Fig. 5b demonstrates that this similarity was captured by the attention weights.

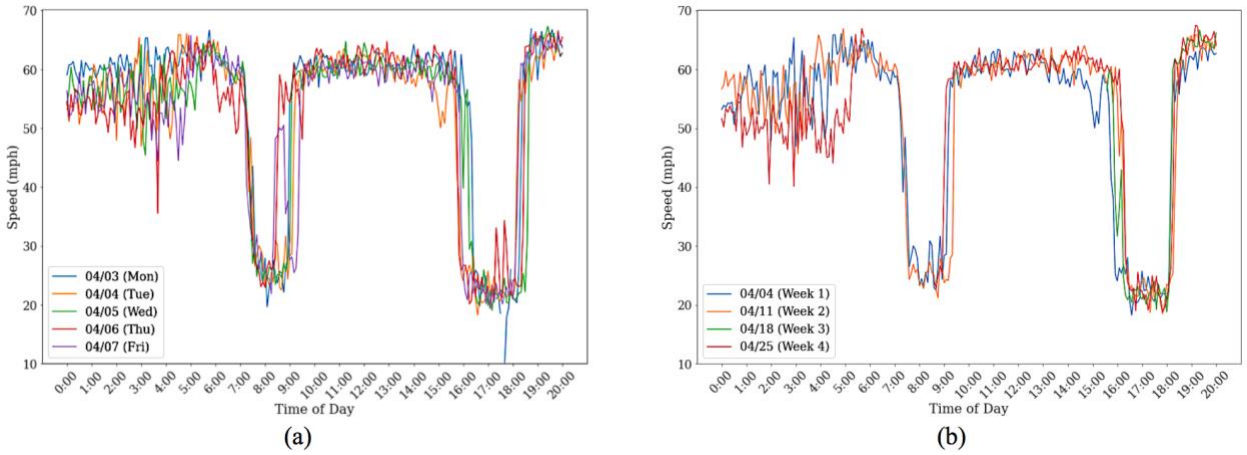


Figure 3.5 (a) Daily and (b) weekly trends of traffic speed at freeway milepost 09.6

Figure 3.6 (a), (b) and (c) map the attention weights throughout a sample day for 5-minute, 30-minute and 60-minute attention horizons, respectively. Like Figure 3.4, the data samples used for visualization were extracted from the testing set. The plots illustrate the model's choice to focus on weekly periodic characteristics during evening peak hours and a combination of daily and weekly cyclic features for off-peak hours. It can also be observed from Figure 3.6, as well as Figure 3.4 (a) and (b), that the model relied more on short-term features when forecasting speed for a shorter output horizon. As the model makes farther predictions, it relies more on daily and weekly features. Shorter horizons are inherently easier to predict, given that there is less variability between the prediction and short-term features. Furthermore, assigning more weight to short-term features allows the model to react to atypical conditions like crashes, especially for shorter-horizon speed forecasting which would be impacted the most by sudden abnormalities in traffic conditions. On the contrary, longer-term traffic forecasting is inherently more stochastic. Given that there's more variability between short-term traffic features and longer prediction horizons, the attention layer assigned a heavier weight to daily and weekly patterns.

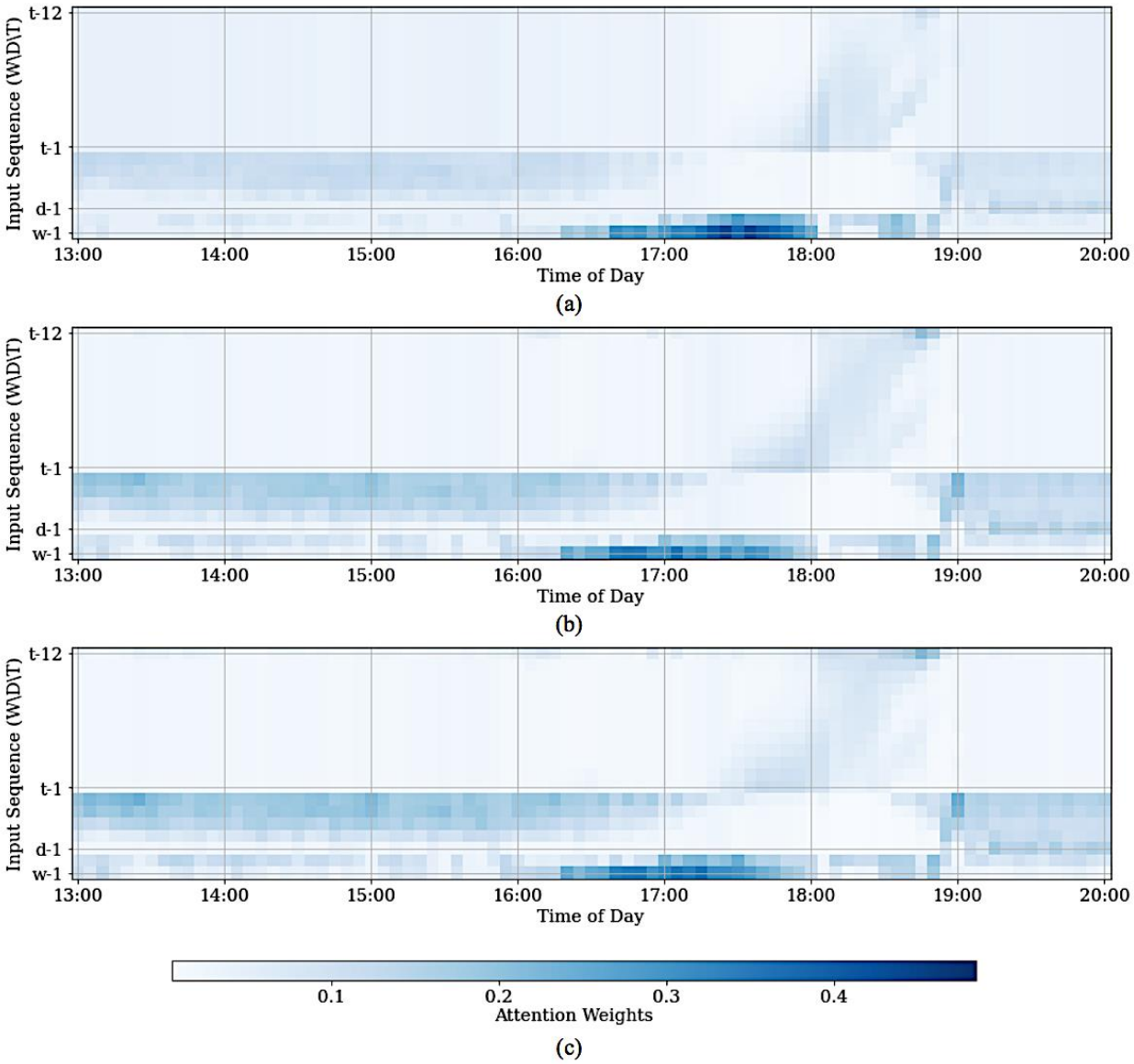


Figure 3.6 Attention weights of the prediction of a sample time segment for (a) 5-minute (b) 30-minute and (c) 60-minute output horizons

3.4 Conclusion

The architecture of the proposed Att-MED model is composed of multiple components. The encoder used convolutions and a recurrent neural network cells to capture the spatiotemporal relationship of the input sequence. The usage of multiple encoders helped capture the cyclic daily

and weekly traffic patterns and facilitated their use as additional input features. The attention layer enabled the use of multiple encoders, and by extension, multiple traffic sequence inputs. Attention balanced between the different encoder outputs and regulated their contribution towards prediction at each output horizon. Finally, the LSTM decoder modeled the temporal relationship of the output sequence. Training the attention-based multi-encoder decoder network end-to-end produced superior 60-minute ahead prediction accuracies compared to the baseline models. Furthermore, the attention layer enhanced the interpretability of the model outputs. The model prediction decisions were further studied and validated by comparing against the real traffic patterns. The attention weights confirmed the significance and contribution of the daily and weekly input sequences towards speed prediction. The attention weight maps quantified the value of periodic features, especially for longer-horizon prediction.

Accurate 60-minute ahead traffic speed prediction has many useful applications. For regular commuters and connected vehicles, it can be used for trip planning, routing and preemptive rerouting in case of predicted congestions. It can also be applied in traffic management, especially during peak hours. In practice, 60-minute ahead prediction gives traffic operators plenty of time to react to predicted traffic turbulence. Additionally, in contrast to traditional deep learning models, attention weight visualization enhances the interpretability of the proposed model, which allows operators to make nuanced, informed, and explainable decisions.

CHAPTER 4: SEQUENCE-TO-SEQUENCE RECURRENT GRAPH CONVOLUTIONAL NETWORKS FOR TRAFFIC ESTIMATION AND PREDICTION USING CONNECTED PROBE VEHICLE DATA

4.1 Introduction

Traffic parameters estimation, is indispensable for various fundamental transportation applications such as transportation planning, traffic incident detection, and conducting traffic operation and safety studies (Mahmoud et al. 2021a). Currently, most traffic data, such as traffic volume and speed, is collected using sensing hardware infrastructure such as inductive loop detectors, radar detectors, or roadside cameras. While these sensors can provide accurate measurements, they have several drawbacks. Firstly, the hardware sensors are expensive and time consuming to setup and deploy, especially on a large scale. Secondly, sensors are prone to hardware failure or network outage, which leads to data loss and high maintenance costs. Thirdly, the sensor spatial distributions are usually limited in order to minimize deployment cost. This problem entails that a single sensor downtime leads to missing traffic data for an entire road segment.

Recent advances in vehicular networking technology can disrupt the traditional traffic estimation methods. Newer vehicles are equipped with network-enabled on-board units (OBUs) which are capable of real-time vehicle-to-everything (V2X) communication with external agents. The V2X communication paradigm facilitates real-time sampling of per-vehicle location, speed, and heading data. When collected from a fleet of connected probe vehicles, the sampled data can provide enough information for infrastructure-free traffic flow estimation. Utilizing connected probe vehicle for traffic estimation can resolve many shortcomings of infrastructure-based data

collection methods. It can be used to derive traffic parameters without the dependence on pre-deployed infrastructure, meaning that new roadway segments and remote areas can be covered more quickly and inexpensively. Moreover, the decentralized nature of probe vehicle data sampling means that there's no single point of failure that can lead to missing data. A single road segment is covered by multiple vehicles and therefore sampled using various sensors.

In addition to traffic flow estimation, probe-vehicle data can be utilized for short-term traffic prediction. Traffic parameters prediction is essential for numerous data-driven Intelligent Transportation System (ITS) applications such as route optimization, dynamic navigation, and traffic scheduling (Zhang et al. 2011). Traditional traffic prediction methods depend on infrastructure-based data as input, and hence suffer from the same shortcomings as static infrastructure-based traffic estimation.

Traffic network modeling has long been studied in the literature. However, most studies utilized infrastructure-based data for traffic estimation and prediction (Kwon, Varaiya, and Skabardonis 2003; Pan, Demiryurek, and Shahabi 2012; Wilkie, Sewall, and Lin 2013; Abdelraouf, Abdel-Aty, and Yuan 2021; Ma, Tao, et al. 2015; Jin et al. 2018; Mahmoud et al. 2021b). Probe vehicle data presents a different set of challenges when employed for traffic modeling. Probe vehicle data only captures the traffic parameters from a sample of the vehicles on the road. It is easier for probe vehicle data to estimate traffic speed, however it cannot be directly used to infer traffic volume. Therefore, in previous research efforts, most authors focused on utilizing probe vehicle data for travel time estimation and prediction (Zheng and Van Zuylen 2013; Efentakis et al. 2013; Pfoser, Tryfona, and Voisard 2006; Wang, Zheng, and Xue 2014; Zhang et al. 2017; Zhan, Ukkusuri, and Yang 2016; Li et al. 2017; Derrow-Pinion et al. 2021). Nevertheless, traffic volume is a vital traffic parameter. It is an imperative measurement that can be further used

to derive other traffic parameters, namely traffic flow and traffic density.

Traffic estimation and prediction are complex problems due to the stochastic nonlinear relationships between traffic parameters in both spatial and temporal dimensions. Many methodologies have been utilized such as parametric statistical modeling (Ahmed and Cook 1979; Kumar and Vanajakshi 2015; Chandra and Al-Deek 2009) and machine learning-based methods (Castro-Neto et al. 2009; Ye, Szeto, and Wong 2012; Van Lint, Hoogendoorn, and van Zuylen 2005). However, due to advancements in computation time and efficiency, and due to the availability of large traffic datasets, deep neural networks-based methods have been widely applied in recent years. Deep learning methods have demonstrated superior traffic model compared to traditional machine learning and statistical modeling methods (Zheng et al. 2020; Zhao, Song, et al. 2019; Wu et al. 2019).

In this research effort, a sequence-to-sequence neural-network-based methodology for traffic estimation and prediction was proposed. The neural network consists of an encoder-decoder architecture which combines Graph Convolution Networks (GCN) for spatial modeling and Long-Short-Term-Memory networks (LSTM) for capturing temporal dependencies. The model performs traffic estimation and up to 60-minute ahead traffic prediction in the form of traffic speed and traffic volume. Moreover, the methodology generates traffic volume and speed estimations and predictions for all modeled road segment locations concurrently. The methodology utilizes connected probe vehicle data in the form of Wejo vehicle movement data (Wejo 2021a). Wejo provides probe vehicle movement data which was curated from multiple commercial cars Original Equipment Manufacturers (OEM) (Wejo 2021b). The proposed methodology was validated using Microwave Vehicle Detection System (MVDS) traffic sensors deployed at 182 locations distributed across 4 different freeways in Orlando, Florida, for a period of 14 days. Subsequently,

the results were compared against baseline models. In summary, the contributions of the proposed research are threefold:

1) A deep neural network methodology was used to perform road-network-wide infrastructure-free traffic volume and speed estimation and up to 60-minutes ahead prediction. The outputs were validated using traffic data collected from roadside microwave sensors as the ground truth.
2) Seq2seq GCN-LSTM, a sequence-to-sequence deep neural network architecture was developed and proposed. The method employs Graph Convolution Networks and Long Short-Term Memory to model traffic parameters in the spatial and temporal dimensions, respectively.
3) The penetration rate of the probe vehicle data was manually varied in order to measure its effect on the proposed model's traffic estimation and prediction capabilities.

4.2 Data Description

In this research effort, connected probe vehicle data from the Wejo dataset was used to estimate and predict network-wide traffic speed and volume. Roadside sensor-based data from Microwave Vehicle Detection System (MVDS) detectors was additionally collected and regarded as the ground truth. The distribution of the study area basemap is illustrated in Figure 4.1. The data was collected from 182 segments distributed across 4 expressways in Orlando, Florida for 14 days: SR-408, SR-417, SR-528, and Florida Turnpike. The total length of the roadways covered by the study was 112.4 miles. The road segmentation was defined by the availability of MVDS data. Each location marked in Figure 4.1 signifies the location of an MVDS detector. As illustrated, each of the expressways intersects with two others, indicating a strong spatial correlation between the traffic parameters across the roadway segments. The proposed model was trained using probe

vehicle data features as input variables and the MVDS data as the output. The resulting trained model was able to make infrastructure-free network-wide traffic speed and volume estimations and prediction purely based on probe vehicle data.

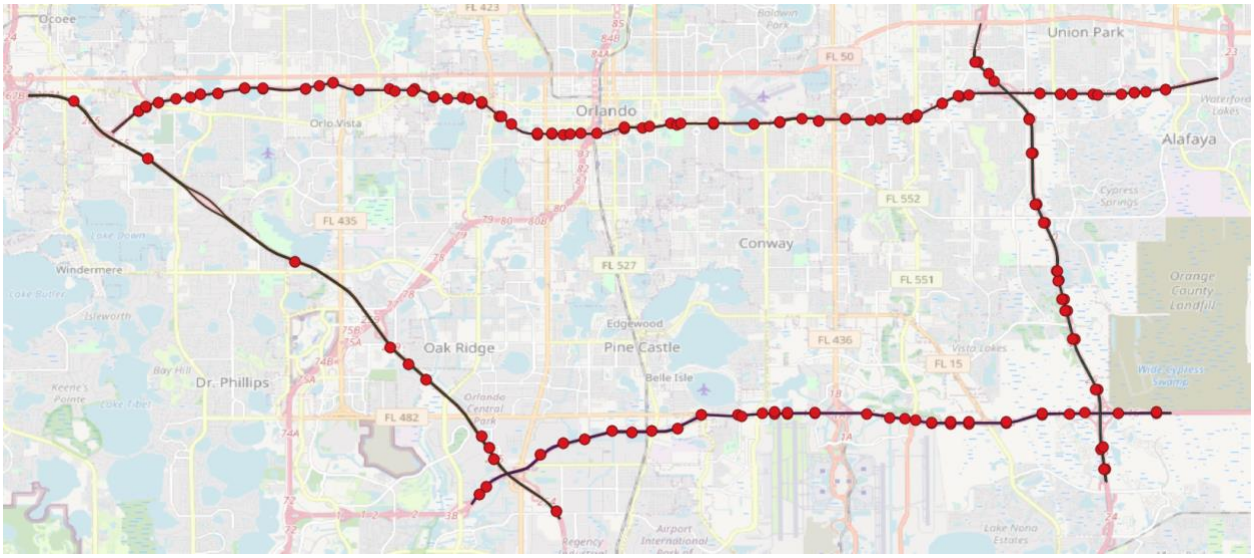


Figure 4.1 Study area map and sensor locations

4.2.1 Raw Data

4.2.1.1 Connected Probe Vehicle Data

The Wejo connected probe vehicle datapoints located on the study area expressways were collected. The Wejo fleet consists of commercial vehicles, meaning that the vehicles in the dataset represent a varied sample of the vehicles on the road. Furthermore, vehicle-to-cloud (V2C) communication was used to send the vehicles' information from the fleet directly to the Wejo servers. The dataset consisted of approximately 61.3 million GPS points generated from 580,864 unique trips. Each datapoint contained location information (latitude, longitude), timestamp, speed, heading, and journey ID. The sampling interval of the probe vehicle data was 3 seconds.

4.2.1.2 Microwave Sensor Data

The MVDS data was collected from the 182 detectors shown in Figure 4.1. The data was downloaded from the Regional Integrated Transportation Information System (RITIS). The microwave sensors detected spot vehicle speed. Consequently, the raw microwave sensor data consisted of average speed and total volume aggregated by 1 minute for each detector.

4.2.2 Data Processing

Firstly, the probe vehicle points were filtered by GPS location. Points that were not on the study area expressways were filtered out. Next, each MVDS detector was matched to the set of waypoints that lied within a 0.075-mile radius. The matching threshold was chosen based on the sampling rate. Since the sampling interval was 3 seconds, any vehicle driving under 180 miles per hour was captured and matched using at least 1 waypoint from the trip. Afterwards, to ensure that each detector was matched with waypoints traveling in the same direction, the points were filtered by heading according to the alignment of their corresponding detector. Finally, both the probe vehicle and microwave datasets were aggregated to 5-minute intervals.

4.2.3 Data Analysis

The 5-minute aggregated data from each MVDS detector was compared to the corresponding aggregated matched probe vehicle waypoints. The total number of trips was compared to the total sum of volume to compute the penetration rate of the Wejo connected vehicle probe data. Figure 4.2 depicts the distribution of penetration rates for the 182 detector locations. The average penetration rate was 3.422% with a standard deviation of 0.399%. The penetration rate values per location ranged between 2.167% and 4.957%.

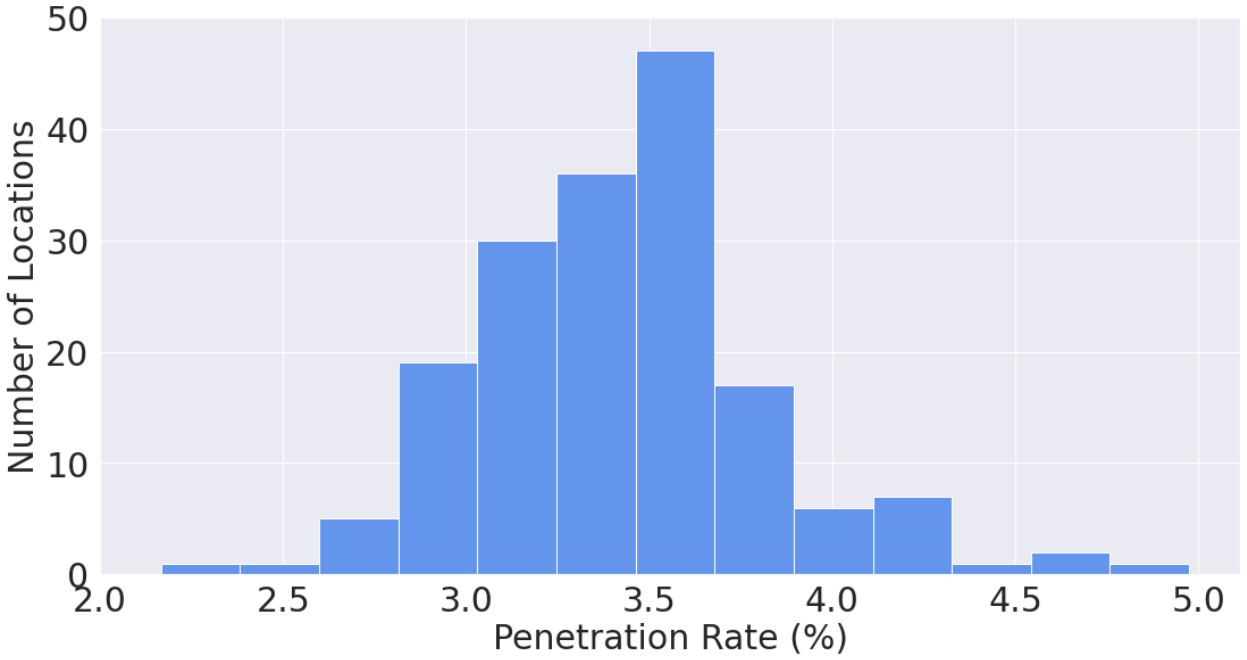


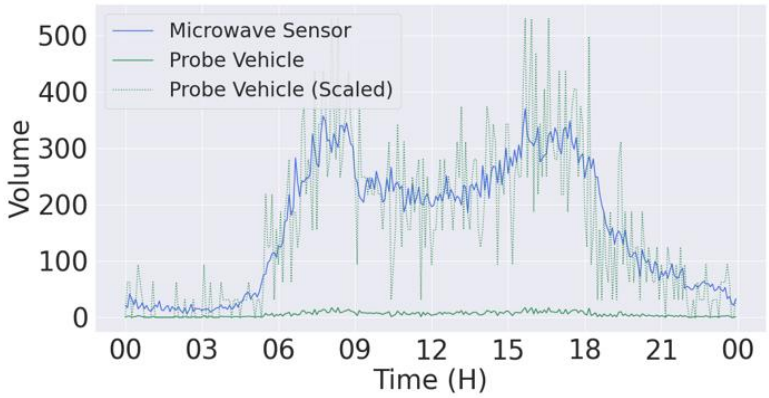
Figure 4.2 Distribution of the Wejo connected probe vehicle data penetration rates per location

Table 4.1 demonstrates the feature-wise summary statistics for both the connected probe vehicle data and the microwave sensor data. The average 5-minute volume of the probe vehicle data was 4.622 vehicles compared to 136.722 vehicles reported by the microwave sensors. The ratio of the two means is 3.380% which is consistent with the computed penetration rate. It can be noticed that the coefficient of variation is higher for the probe vehicle volume compared to the microwave sensor volume. The same phenomenon can be noticed for the 5-minute average speed feature. While the means of the two datasets are very close, the standard deviation and coefficient of variation for the probe vehicle average speed is higher. This discrepancy stems from the low penetration rate of the probe data. It is the reason why the probe data cannot be directly used to accurately estimate or predict traffic parameters.

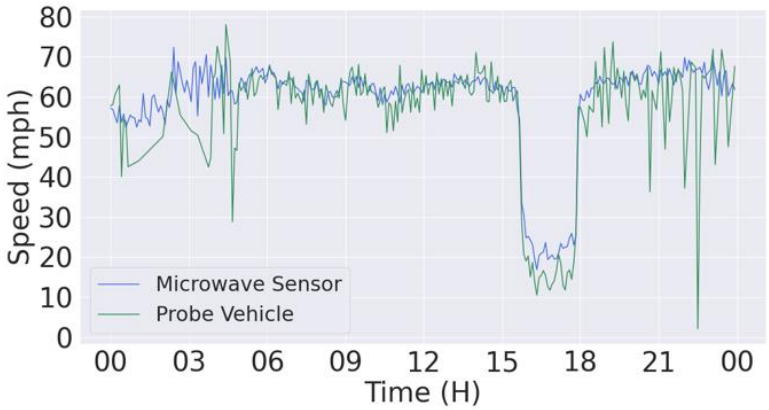
Table 4.1 Datasets' feature-wise summary statistics

	Mean	Std. Deviation	Coeff. of Variation	Maximum	Minimum
Connected Probe Vehicle Data					
Sum Volume	4.622	2.361	0.511	29.000	0.000
Avg Speed	68.134	7.210	0.120	118.103	0.000
Microwave Sensor Data					
Sum Volume	136.722	47.468	0.347	686.000	0.000
Avg Speed	68.405	6.864	0.100	117.462	0.000

The graphs in Figure 4.3 were plotted to further investigate the difference between the dataset characteristics. Figure 4.3 (a) and (b) compare the volume and speed plots of the microwave sensor data versus the probe vehicle data for the roadway segment SR-408 milepost 9.2 eastbound. Additionally, Figure 4.3 (a) depicts the scaled probe vehicle data volume, which is the volume data scaled by the penetration rate for the graphed location. It can be concluded that the Wejo probe vehicle data can capture the temporal trend of traffic. However, due to the low penetration rate, there's a high variance between the reported adjusted volume and the microwave sensor volume data. A similar situation can be observed in Figure 4.3 (b) for the average speed. The reported probe vehicle speed captures the temporal speed trend with some variance caused by the low penetration rate. However, Figure 4.3 (b) indicates that the probe vehicle data can report irregular speed readings such as demonstrated between 21H and 00H. The eccentric speed reading could be coming from an equipped vehicle that was performing an atypical action, such as stopping on the shoulder. Due to the low number of probe vehicles, highly deviated datapoints could have a considerable effect on the aggregated average speed data.



(a)



(b)

Figure 4.3 Probe vehicle data versus microwave vehicle data plots for 5-minute aggregated (a) total volume and (b) average speed

4.3 Methodology

The proposed methodology was designed to model the past short-term traffic parameters extracted from the probe vehicle data in order to simultaneously estimate the ongoing traffic and predict the upcoming short-term traffic parameters. Traffic parameters exhibit stochastic nonlinear dependencies in both the spatial and temporal dimensions. To accurately capture both dependencies, an encoder-decoder deep neural network architecture was developed. Figure 4.4

illustrates the overall model architecture. The encoder module was used to model the input past traffic parameters, while the decoder was used to compute the output estimations and future prediction. Graph Convolution Networks were employed to model the spatial dependencies and Long Short-Term Memory networks were utilized to capture the temporal relations. The model was named Sequence-to-Sequence Graph Convolution Network-Long Sort-Term Memory (Seq2seq GCN-LSTM).

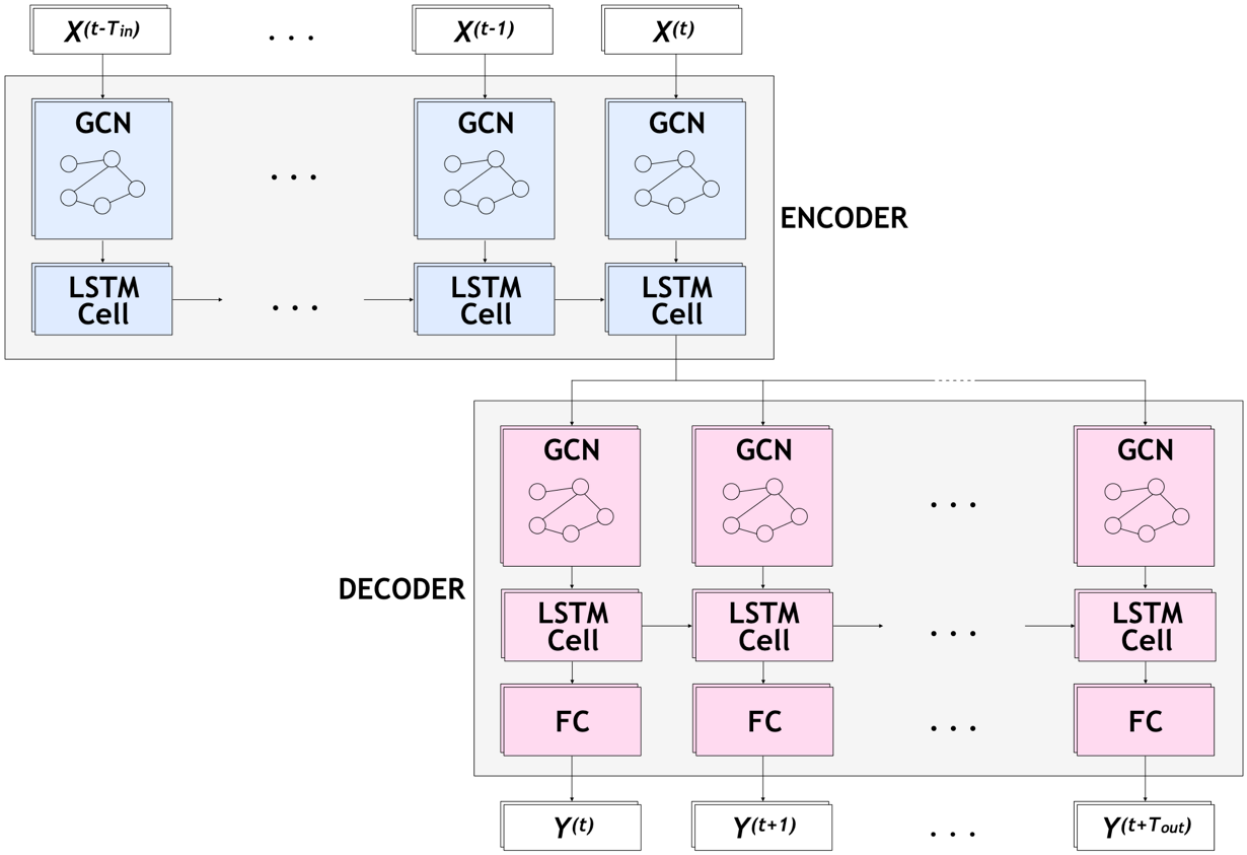


Figure 4.4 Overall architecture of the proposed Seq2seq GCN-LTM methodology

4.3.1 Graph Convolution Networks

Graph neural networks were designed as an alternative to convolution neural networks that can effectively model graph-based structures (Kipf and Welling 2016). The GCN takes 2 inputs:

an adjacency matrix, and a set of features. The adjacency matrix $A \in \mathbb{R}^{N \times N}$, where N is the number of vertices, represents the graph structure. The set of features $X \in \mathbb{R}^{N \times D}$, where D is the number of features, represents the input feature vector per vertex in the graph.

In the proposed methodology, the GCN was used to model the spatial relationship between connected segments in the road network topology. The road network was defined as an undirected weighted graph $G = (V, E)$, where V is the set of vertices of length N defined by the MVDS detector locations and E is the set of edges which correspond to the road segments that connect pairs of vertices. The weights of each graph edge were designated as the distance between neighboring vertices. Subsequently, for each edge in E that connects vertices i and j, the adjacency matrix element $A_{i,j}$ was set to the edge weight. Since the graph was undirected, the output adjacency matrix was symmetric ($A_{j,i} = A_{i,j}$). All other entries in the matrix were set to zero.

A multi-layer GCN network was employed where L^{GCN} is the number of layers. The GCN model uses the input adjacency matrix to propagate the feature information from neighboring vertices. Additionally, to utilize each vertex's own features, a self-connection was established by adding the identity matrix to the adjacency matrix $\tilde{A} = A + I$. Furthermore, the diagonal vector $\widetilde{D}_{i,i} = \sum_j^N A_{i,j}$ was used to normalize the adjacency matrix weights. The output of each layer was calculated as follows:

$$H^{(l)} = RELU\left(\widetilde{D}^{-\frac{1}{2}}\tilde{A}\widetilde{D}^{-\frac{1}{2}}H^{(l-1)}W^{(l-1)}\right) \qquad l = 1 \dots L^{GCN} \tag{4.1}$$

where $RELU$ is the rectified linear unit activation function, and $W^{(l)}$ is the trainable weight matrix of GCN layer l. The input to the first layer $H^{(0)}$ was simply the feature vector X. Finally, the output of the GCN network corresponded to the result of the last layer activation as shown in

equation 2.

$$GCN(X, A) = H^{(L^{GCN})} \tag{4.2}$$

4.3.2 Long Short-Term Memory

Long Short-Term Memory (Hochreiter and Schmidhuber 1997) is a recurrent neural network architecture that was designed to model sequential input. In the proposed methodology, the LSTM network was utilized to model the temporal relationship between traffic parameters at subsequent timesteps.

The LSTM network utilizes a sequence of connected cells, one cell per element in the input sequence. Each cell computes its internal state c_t and hidden state h_t state based on 3 functions: the input gate (i_t), forget gate (f_t), and output gate (o_t). The input and forget gates decide how much information should be drawn from the current timestep features x_t and how much information should be utilized from the previous cell hidden state h_{t-1}, respectively. The output gate is used to calculate the current cell hidden state h_t, and thus decides how much information gets propagated to the next cell. Equations 3 – 8 specify the calculations made by each LSTM cell at timestep t. The computations are repeated for each element in the modeled sequence $t = 0 \dots T$. The weight matrices w and bias vectors b are the network trainable parameters. σ and $*$ denote the sigmoid activation function and element-wise multiplication, respectively.

$$i_t = \sigma(w_i[h_{t-1}, x_t] + b_i) \tag{4.3}$$

$$f_t = \sigma(w_f[h_{t-1}, x_t] + b_f) \tag{4.4}$$

$$o_t = \sigma(w_o[h_{t-1}, x_t] + b_o) \tag{4.5}$$

$$\tilde{c}_t = tanh(w_c[h_{t-1}, x_t] + b_c) \tag{4.6}$$

$$c_t = f_t * c_{t-1} + i_t * \tilde{c}_t \tag{4.7}$$

$$h_t = o_t * tanh(c_t) \quad (4.8)$$

4.3.3 Encoder

The encoder network was responsible for extracting the spatiotemporal information from the input and encoding the features to an intermediate representation. The input was a temporal sequence comprised of the historical 5-minute aggregated volumes and average speeds extracted from the probe vehicle data. The GCN and LSTM networks were employed to model the spatial and temporal dependencies, respectively. As illustrated in Figure 4.4, the encoder received a normalized input $X \in \mathbb{R}^{T_{in} \times N \times D}$, where T_{in} was the length of the input time sequence. For each input timestep t, graph convolution was applied to the feature vector $X^{(t)} \in \mathbb{R}^{N \times D}$. Afterwards, the outputs of each convolution operation $GCN(X^{(t-T_{in})}), \dots, GCN(X^{(t)})$ were transmitted as input to the encoder LSTM network. The LSTM cells encoded the historical temporal features up to the current timestep t.

4.3.4 Decoder

The decoder network was used to decipher the intermediate representation vector computed by the encoder. The encoder output was used as input to the decoder GCN network. The decoder GCN extracted the spatial relationship between output estimations and predictions across neighboring road segments. Next, the decoder LSTM cells were used to model the temporal relationship between the output timesteps. Finally, a multi-output sigmoid activated fully connected (FC) layer was employed to generate an output vector $Y^{(t)} \in \mathbb{R}^{N \times D}$ for each output timestamp t. The output vector was used to extract the 5-minute aggregated total volume and average speed at each vertex in the road network graph concurrently. The FC layer was preceded by a dropout layer to act as a regularization technique.

The decoder produced the output vector $Y \in \mathbb{R}^{T_{out} \times N \times D}$, where T_{out} was the length of the

output sequence. The first element in the output sequence $Y^{(t)}$ was used to estimate the ongoing traffic parameters. The remaining elements $Y^{(t+1)}, \ldots, Y^{(t+T_{out})}$ were denormalized and used for traffic parameter prediction. During model training, the output datapoints Y were specified from the aggregated microwave sensor volume and average speed data.

4.3.5 Loss Function

Mean absolute error (MAE) was used as the loss function for the proposed methodology. MAE was chosen to fit the multi-output regression task performed by the network. Equation 9 indicates the computation carried out by the loss function where Y and $\hat{Y}$ represent the ground truth and model prediction vectors, respectively. The loss function measures the error at each output timestep for all location and for each target output feature. To train the proposed model end-to-end, the Adam optimizer (Kingma and Ba 2014) was employed to reduce the loss value through the backpropagation algorithm.

$$\mathcal{L}(Y, \hat{Y}) = \frac{1}{T_{out} * N * D} \sum_{t}^{T_{out}} \sum_{n}^{N} \sum_{d}^{D} \left| Y_{t,n,d} - \hat{Y}_{t,n,d} \right| \tag{4.9}$$

4.4 Experimentation

4.4.1 Setup

The probe vehicle data and the corresponding microwave sensor data were used to train and validate the proposed neural network architecture for traffic volume and speed estimation and prediction. The model utilized probe vehicle data from the past hour to estimate the current traffic state and predict the traffic parameters up to 60 minutes ahead in 5-minute increments ($T_{in} = T_{out} = 13$). To improve performance, the proposed model's hyperparameters were optimized. The hyperparameter search space included the number of layers for the encoder GCN, encoder LSTM,

decoder LSTM, decoder GCN. Furthermore, each layer's corresponding size was tuned. Moreover, the decoder's dropout ratio was tuned. During model training, the batch size and optimizer's learning rate were additionally tuned. Table 4.2 summarizes the variables in the hyperparameter optimization search space and their corresponding ranges.

The available data was split into 75% for model training and 25% for model testing. During model training, the learning rate was reduced on plateau by a factor of 0.5. Early stopping was implemented to avoid model overtraining.

The proposed model was implemented using Tensorflow (Abadi et al. 2016) and Keras (Chollet 2015) deep learning libraries. Moreover, KerasTuner (O'Malley et al. 2019) random search tuner was used to find the optimal hyperparameter values and subsequently optimize model performance.

Table 4.2 Seq2seq GCN-LSTM hyperparameter search space

Hyperparameter	**Range**	**Step**
GCN layers	[1, 5]	1
LSTM layers	[1, 5]	1
GCN layer size	[4, 32]	4
LSTM layer size	[128, 512]	128
Dropout ratio	[0.0, 0.5]	0.1
Batch size	[32, 128]	32
Learning rate	$[10^{-4}, 10^{-2}]$	0.5 (log scale)

4.4.2 Baseline Models

The proposed methodology was compared to three baseline models. The outputs for each target horizon were computed separately. The description of each baseline model is as follows:

- **SVR**: Support Vector Regression with a radial basis function (RBF) kernel. The SVR model was trained using all road segment features from all historical timesteps as input features.

- **LSTM**: A Long Short-Term Memory model which was trained using all road segment features per historical timestep as input features.
- **GCN**: A Graph Convolution Network model which was trained using all historical timestep features per road segment as input features. The same adjacency matrix that was used for the proposed model was utilized.

4.4.3 Evaluation Metrics

Three common regression performance metrics were used to evaluate estimation and prediction errors of the proposed and baseline models: Mean Absolute Percentage Error (MAE), Root Mean Square Error (RMSE), and Mean Absolute Percentage Error (MAPE). MAE measures the average error while RMSE uses a squared term to emphasize the penalty on larger error margins. Equations 10, 11, and 12 describe the computations carried out by the evaluation metrics functions. Y_i and $\hat{Y}_i$ refer to the ground truth and prediction vectors for one datapoint, respectively. n signifies the total number of datapoints used for evaluation.

$$MAE = \frac{1}{n}\sum_{i=0}^{n}\left|Y_i - \hat{Y}_i\right| \tag{4.10}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=0}^{n}(Y_i - \hat{Y}_i)^2} \tag{4.11}$$

$$MAPE = \frac{100}{n}\sum_{i=0}^{n}\left|\frac{Y_i - \hat{Y}_i}{Y_i}\right| \tag{4.12}$$

4.4.4 Estimation and Prediction Results

Table 4.3 demonstrates the 5-minute aggregated total volume and average speed results on the test dataset. The first two rows in the table describe the traffic parameter estimation results,

while the rest of the table lists the traffic parameter prediction results. In general, traffic volume was more difficult to estimate/predict that traffic speed. This is a result of the low probe vehicle penetration rate which resulted in a higher variance traffic volume trend compared to the captured speed as demonstrated in Figures 3 (a) and 3 (b). In addition, since the study area consisted of an expressway road network, many of the road segments did not have clear morning and afternoon peak drops in speed. On the other hand, traffic volume temporal fluctuation patterns occurred on every road segment.

The proposed Seq2Seq GCN-LSTM model yielded the best estimation and prediction results compared to the baseline models across the board. The neural network-based methods, particularly the LSTM and proposed methodology, were far superior to the SVR model output. The stand-alone LSTM model produced better results than the stand-alone GCN model. This outcome signifies that the temporal dependency of the input sequence had a more significant effect than the spatial dependency in the study area. Nevertheless, the Seq2Seq GCN-LSTM model produced better results compared to both the stand-alone GCN and LSTM models. The results demonstrate the synergetic effect of the GCN and LSTM modules in the proposed methodology. It categorically highlights the importance of modeling both the spatial and temporal dependencies of traffic volume and speed.

The proposed Seq2Seq GCN-LSTM generated the best longer-term prediction results. The results produced by the baseline models exhibit more deterioration as the prediction horizon increased. This decline is more notable for the models that have no capability of capturing temporal dependencies, namely the SVR and GCN models. The results emphasize the importance of modeling the temporal interdependency between the elements of the predicted sequence. The proposed model could capture this relationship by utilizing the decoder LSTM network.

Table 4.3 Volume and speed estimation and prediction results

Horizon	Metric	Sum Volume				Avg Speed			
		SVR	LSTM	GCN	Seq2seq GCN-LSTM	SVR	LSTM	GCN	Seq2seq GCN-LSTM
Real-Time Estimation									
0 min	**MAE**	23.756	18.150	24.015	**15.924**	4.374	1.768	1.783	**1.601**
	RMSE	29.330	25.276	32.343	**22.500**	5.121	2.776	2.741	**2.618**
	MAPE	13.191	10.727	14.939	**9.839**	7.176	3.552	3.595	**3.408**
Short-Term Prediction									
5 min	**MAE**	24.350	18.142	25.078	**16.029**	4.393	1.770	1.769	**1.608**
	RMSE	29.990	25.354	34.262	**22.673**	5.140	2.773	2.763	**2.616**
	MAPE	13.634	10.689	15.626	**9.849**	7.205	3.558	3.583	**3.410**
10 min	**MAE**	25.031	18.216	26.499	**16.480**	4.416	1.779	1.817	**1.616**
	RMSE	30.747	25.407	36.061	**23.233**	5.162	2.784	2.793	**2.617**
	MAPE	14.148	10.711	16.290	**10.220**	7.240	3.566	3.602	**3.417**
15 min	**MAE**	25.758	18.249	27.548	**16.824**	4.430	1.785	1.817	**1.620**
	RMSE	31.529	25.432	37.865	**23.680**	5.177	2.790	2.842	**2.617**
	MAPE	14.602	10.740	16.949	**10.393**	7.262	3.568	3.629	**3.420**
Longer-Term Prediction									
30 min	**MAE**	28.051	18.758	31.544	**17.531**	4.486	1.795	1.837	**1.623**
	RMSE	33.968	26.426	43.475	**24.465**	5.236	2.791	2.890	**2.616**
	MAPE	16.236	11.179	19.141	**10.876**	7.350	3.673	3.814	**3.421**
45 min	**MAE**	30.106	18.835	35.698	**17.841**	4.498	1.795	1.858	**1.623**
	RMSE	36.278	26.640	48.655	**24.630**	5.252	2.796	2.912	**2.611**
	MAPE	17.723	11.372	21.475	**11.050**	7.372	3.693	3.785	**3.420**
60 min	**MAE**	31.413	18.999	38.841	**18.393**	4.495	1.852	1.858	**1.633**
	RMSE	37.932	26.777	52.924	**25.376**	5.254	2.843	2.909	**2.626**
	MAPE	18.604	11.518	23.479	**11.326**	7.370	3.814	3.837	**3.436**

4.4.5 Output Visualization

Figure 4.5 plots the microwave sensor readings (ground truth), the probe vehicle data (input features), the probe vehicle data scaled by the location-wise penetration rate, and the Seq2seq GCN-LSTM model estimation results. The graphs illustrate traffic volume and speed at three different road segments from 3 different expressways in the study area, namely SR-408 milepost 6.0 eastbound, SR-528 milepost 6.4 eastbound, and Florida Turnpike milepost 265.2 northbound.

The 5-minute aggregated total volume estimations are demonstrated in Figure 4.5 (a), (c), and (e). Despite the low penetration rate, the model was able to sufficiently capture the volume trend. The probe vehicle volumes, when scaled using the penetration rate ratio, exhibit high variance, and indicate that they cannot be directly used to compute the volume, thereby confirming

the importance of utilizing a highly nonlinear model.

Figure 4.5 (b), (d), and (f) illustrate the 5-minute average speed estimations. As depicted in the graphs, the probe vehicle average speeds provide a high variance input speed reading to the proposed model. Nevertheless, by relying on the adjacent traffic readings in the spatial and temporal dimensions to accurately estimate traffic volume, the Seq2seq GCN-LSTM was able to capture the speed trend. Moreover, Figure 4.5 (f) and (d) demonstrate erratic input speed readings from the probe vehicle dataset, for instance, the probe vehicle speed reading in Figure 4.5 (f) at 18H. However, the proposed model's output demonstrates robustness against outlier input speed values as demonstrated by the estimation plots in the figures. One shortcoming of the proposed model is its lack of ability to capture the sharp drop in traffic speed during the peak period. As shown in Figure 4.5 (d), the model estimated a drop in speed, but not as deep as the corresponding ground truth values.

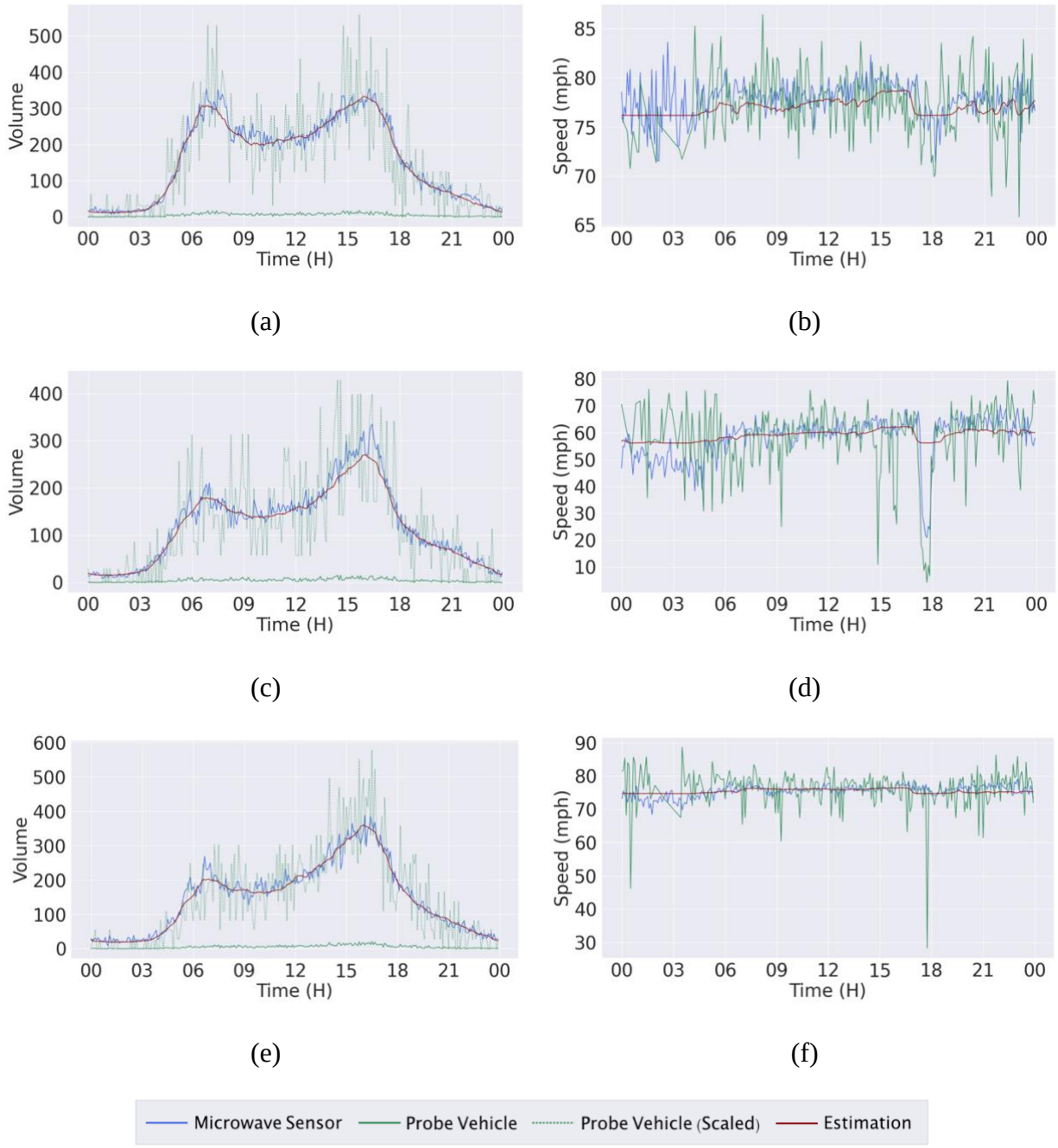


Figure 4.5 Speed and volume estimation plots

4.4.6 Penetration Rate Analysis

Figure 4.6 was plotted in order to understand the effect of the probe vehicle penetration rate on the model's ability to estimate and predict traffic volume and speed. The figures illustrate

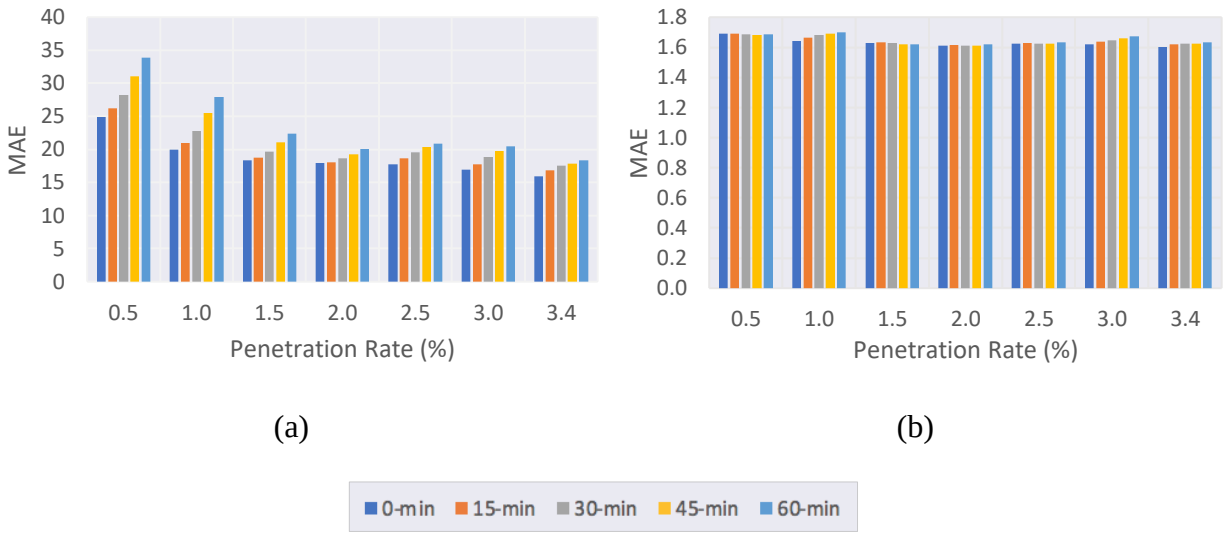
the output evaluation results at different target horizons: 0-min (estimation), 15-min, 45-min, and 60-min. To emulate the lower penetration rate, the number of unique trips in the probe vehicle dataset was manually discounted to reach the desired penetration rate. Journey IDs were randomly selected, and their corresponding GPS were removed. Table 4.4 summarizes the probe vehicle feature statistics under different penetration rate values. The Wejo probe vehicle fleet average penetration rate of 3.4% was used as the baseline for comparison. It can be noted that, while the volume feature values decrease with the penetration rate as expected, the mean speed value remains very similar, and the speed standard deviation increases slightly due to the lower number of vehicles.

Figure 4.6 (a) illustrates the effect of penetration rate on traffic volume estimation and prediction. The model maintained a steady output MAE until the penetration rate was lowered to 1.5%. The average increase in MAE across all estimation and prediction horizons was 15.6%. The model output rapidly degraded below 1% penetration rate. At 0.5% penetration rate, the average MAE increased by 66.2%. Moreover, it can be observed that at lower penetration rate values, longer-horizon traffic prediction performance declined more compared to traffic estimation.

Figure 4.6 (b) demonstrates the effect of probe vehicle penetration rate on the model's traffic speed estimation and prediction capability. Unlike the pattern observed with traffic volume, the penetration rate had little effect on the model's speed prediction results. At 0.5% penetration rate, the average MAE across all estimation and prediction horizons increased by 4.2%. It can be concluded that, even with a low penetration rate, probe vehicle data has the ability to represent the population speed. Thus, the model was able to use the probe vehicle data to accurately estimate and predict traffic speed.

Table 4.4 Summary statistics of the probe vehicle features under different penetration rate values

Penetration Rate (%)	Sum Volume		Avg Speed	
	Mean	Std Dev	Mean	Std Dev
0.5	0.780	0.818	68.196	9.241
1.0	1.566	1.213	68.722	8.708
1.5	2.345	1.550	68.659	8.375
2.0	3.099	1.820	68.653	8.469
2.5	3.864	2.110	68.368	8.001
3.0	4.646	2.377	68.433	7.890
3.4 (baseline)	4.622	2.361	68.134	7.210



(a) (b)

Figure 4.6 Penetration rate analysis for (a) volume and (b) speed estimation and prediction

4.4.7 Perturbation Analysis

Probe vehicle data is prone to noise from various sources. For instance, since the traffic data depends on the mobility of the equipped fleet, the numbers might vary depending on the penetration rate of the probe vehicle fleet at a certain location throughout the day. To examine the fault tolerance of the proposed model, a perturbation analysis was conducted. A gaussian noise signal $\mathcal{N}\sim(0,\sigma)$ with mean 0 and standard deviation σ was added to the testing dataset before reversing the data normalization. The standard deviation was varied to test the model robustness

under different noise scales. Figure 4.7 (a) and (b) demonstrate the output MAE after adding noise with different standard deviation values for volume and speed, respectively. The figures illustrate the output evaluation results at different target horizons: 0-min (estimation), 15-min, 45-min, and 60-min. The model output exhibits very little deterioration when gaussian noise with up to 0.6 standard deviation was added to the input: 0.70% average increase in MAE for volume across all estimation and prediction horizons and 0.94% for speed. Adding 1 standard deviation of gaussian noise increased the average output error by 5.5% and 4.4% for volume and speed, respectively. Expectedly, the largest increase in error was for the 60-minute ahead prediction, which was 8.15% and 4.65% for volume and speed, respectively. In general, the plots indicate that the proposed model was resilient against gaussian noise perturbations.

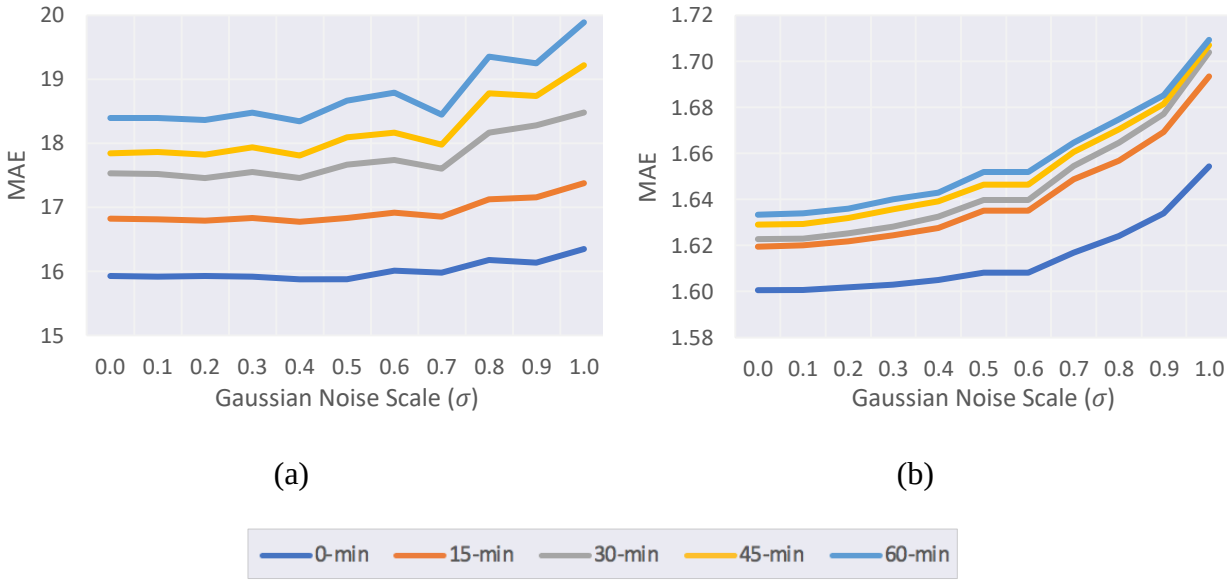


Figure 4.7 Perturbation analysis using gaussian noise for (a) volume and (b) speed estimation and prediction

4.5 Conclusions

A sequence-to-sequence deep learning architecture was proposed for traffic estimation and

prediction. The model, named Seq2seq GCN-LSTM, relied on Graph Convolution Networks for spatial dependency modeling, and Long Short-Term Memory for temporal dependency modeling. The methodology was utilized to perform probe-vehicle-based traffic estimation and prediction. The model utilized short-term historical data collected from a low-penetration probe vehicle fleet to estimate the ongoing traffic volume and speed and to perform up to 60-minutes ahead traffic parameters prediction. The model was validated using roadside microwave sensor data deployed at 182 road segments on 4 expressways in Orlando, Florida. The proposed methodology generated the best estimation and prediction results compared to the baseline models. Despite the high variance of the input probe vehicle volume and speed, the model was able to capture the traffic spatiotemporal dependencies and was more successful at forecasting longer-horizon predictions. Additionally, the proposed method demonstrated robustness against outlying probe vehicle readings and gaussian noise perturbations. The probe vehicle penetration rate was varied in order to test its effect on the proposed method's modeling capability. The results indicated that the model was able to maintain traffic volume estimation and prediction performance until a penetration rate of 1.5% within a 15.6% margin of error. Moreover, the model was able to maintain speed estimation and prediction performance when the penetration rate was as low as 0.5% within a 4.2% margin of error.

In the context of connected vehicles, the proposed method can be utilized for probe vehicle-based real-time traffic estimation and prediction. The generated traffic parameters can be used for many online applications such as real-time trip planning, navigation, and incident detection. Furthermore, the proposed model can be used to reduce the reliance on infrastructure-based traffic data collection. It can additionally be used in conjunction with static sensors to increase the spatial granularity of the traffic data collection and to impute the traffic data at missing or dysfunctional

sensor locations.

CHAPTER 5: USING VISION TRANSFORMERS FOR SPATIAL-CONTEXT-AWARE RAIN AND ROAD SURFACE CONDITION DETECTION ON FREEWAYS

5.1 Introduction

Adverse weather conditions, specifically rain precipitation levels and the resulting wet pavement conditions, have a detrimental impact on driver capabilities, vehicle maneuverability, and roadway infrastructure conditions. Numerous studies have indicated that the presence of rain and/or wet road surfaces has a negative impact on roadway capacity, traffic speed, and density (Chung, Abdel-Aty, and Lee 2018; Maze, Agarwal, and Burchett 2006). Additionally, studies have shown that inclement weather conditions increase traffic crash risk (Abdel-Aty and Pemmanaboina 2006; Yuan et al. 2019). The FHWA has reported that between 2007 and 2016, 15% of fatal crashes, 19% of injury crashes, and 22% of property-damage-only (PDO) crashes occurred in the presence of adverse weather and/or slick pavement (FHWA 2020). Weather and pavement conditions are also used to inform operational decisions such as variable speed limit control, traffic signal timing, and evacuation plan strategies (Rämä 1999; Tahir and Rashid 2020).

In order to alleviate the effects of rainy weather and slippery road surface conditions on traffic flow and crash risk, these conditions must be accurately monitored in real-time and with high spatial granularity. Traditionally, weather conditions are monitored through weather forecasts or roadside weather information systems (RWIS). Weather forecasts combine readings from ground weather stations, satellite sensors and other sources as inputs for weather prediction models. However, the output of weather forecasts is too coarse and too infrequent for real-time fine-grained monitoring of road segments (Sun et al. 2020). Moreover, weather forecast systems

are not equipped for road surface conditions detection which remain slippery after rain stops or during patchy rain. RWIS comprises of Environmental Sensor Stations (ESS), which are equipped with specialized apparatus to accurately assess weather and pavement conditions. The high financial investment required for the deployment and maintenance of RWIS stations limits their spatial distribution and coverage density (Sharma, Wu, and Kwon 2021; Ewan and Al-Kaisy 2017; Manfredi et al. 2005; Garrett et al. 2008). In contrast, roadside traffic CCTV cameras are cheaper and more densely distributed. They can be utilized to detect rain and road surface precipitation states frequently and inexpensively. Traffic cameras can also operate in conjunction with existing systems like RWIS to increase rain/pavement condition detection accuracy, coverage and granularity.

Various research efforts have focused on using cameras for general purpose weather detection (Allamano, Croci, and Laio 2015; Bossu, Hautière, and Tarel 2011; Dong et al. 2017; Pan et al. 2018, 2019; Zen et al. 2019). Many studies have proposed machine learning models to classify hand-crafted image features such as color, texture, and reflection. In the past few years, convolutional neural networks (CNNs) have been widely utilized for image-based rain detection using deep visual features due to their exhibited superior performance on many computer vision tasks (Krizhevsky, Sutskever, and Hinton 2012; Szegedy et al. 2015; He et al. 2016). However, in the recent past, Transformers have been widely adopted for tasks where deep learning has been the established state-of-the-art methodology. Vision Transformers (ViT) have subsequently been used to achieve state-of-the-art results on the Imagenet dataset challenge (Dosovitskiy et al. 2020). Additionally, studies in the literature have targeted stand-alone-camera-based detection. The dense physical distribution of roadside traffic cameras and the relationship between their detection outputs can be utilized to add spatial context awareness to the detection model and enhance the

overall output accuracy.

In this research, a Vision Transformer model is proposed for 3-level rain detection (*heavy_rain*, *light_rain*, and *no_rain*). The proposed method was extended for 2-level road surface condition classification (*wet_road* and *dry_road)*. The ViT was pre-trained on the Imagenet21k (Russakovsky et al. 2015) and the original ImageNet (Deng et al. 2009) datasets. Subsequently, a Spatial Self-Attention (SSA) network is proposed to add spatial context awareness to the stand-alone ViT outputs. The ViT-SSA model observes a sequence of images from adjacent cameras concurrently and detects rain and road surface condition for each image in a sequence-to-sequence manner. The contributions of this work are summarized below:

1) An image dataset that exclusively consists of roadside freeway scenes under different rain conditions was established. The dataset was labeled for a twofold classification task: 3-level rain and 2-level road condition detection. A Vision Transformer that leverages pre-trained weights was fine-tuned, tested, and compared against baseline models on the self-established dataset. To our knowledge, this is the first work that uses Vision Transformers for a weather detection task.
2) The proposed Vision Transformer was combined with a Spatial Self-Attention network to create a sequence-to-sequence model that observes multiple consecutive images. The SSA network adds spatial context awareness by modeling the geographical relationship between the independent ViT detection outputs of individual traffic cameras. The ViT-SSA model simultaneously detects rain and road surface condition for all images. The robustness and interpretability of the proposed model are demonstrated.

5.2 Methodology

The proposed methodology is split into two parts. First, a Vision Transformer was constructed, loaded with pre-trained Imagenet weights, and fine-tuned on the collected image dataset. Secondly, M fine-tuned ViT instances were invoked to classify a sequence of images simultaneously. M corresponds to the number of target adjacent road segments; each was covered by 1 traffic camera and hence each was represented using 1 input image. The M generated results were fed into the Spatial Self-Attention network, which enhances the outputs of the isolated ViT instances by adding spatial context awareness. During the training process of the SSA network, the ViT weights were frozen to retain the model weights learned by the fine-tuning process. Figure 5.1 demonstrates the overall architecture of the proposed ViT-SSA network.

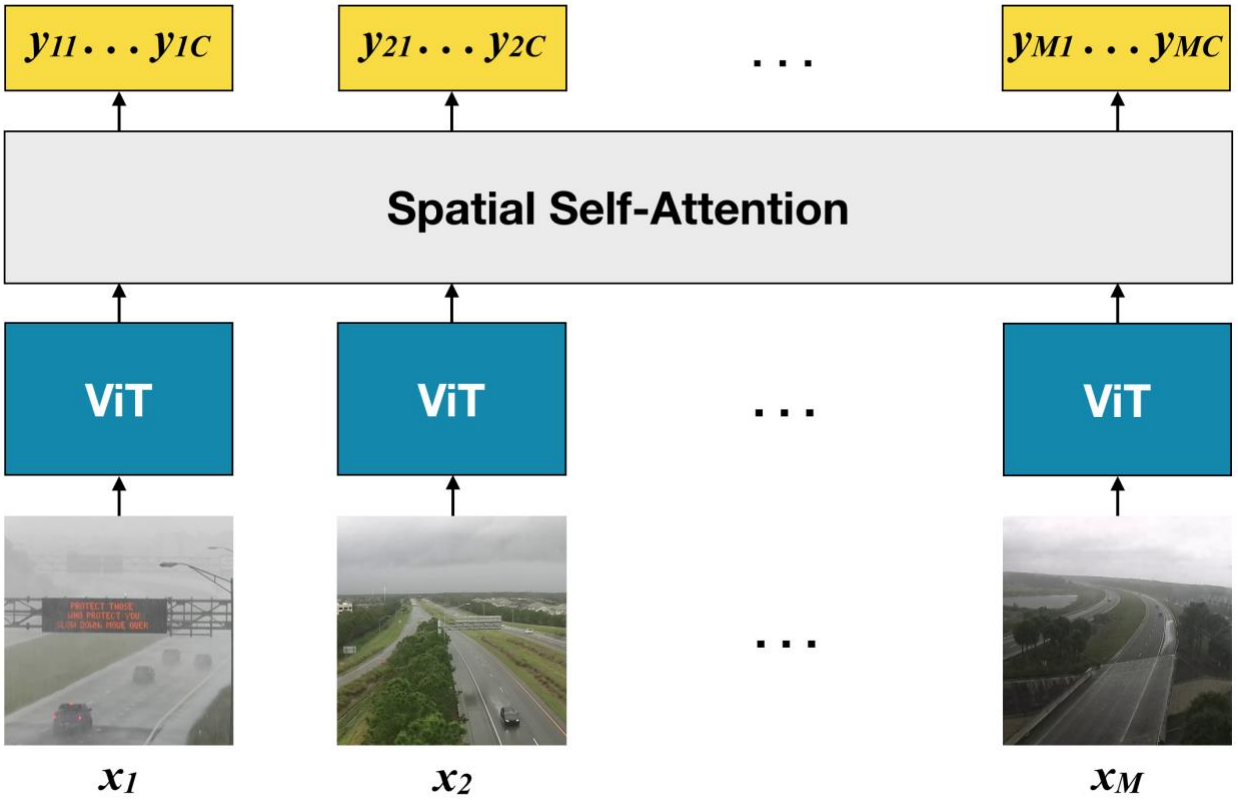


Figure 5.1 The overall structure of the proposed ViT-SSA network

5.2.1 Vision Transformer

Vision Transformers (Dosovitskiy et al. 2020) convert images into classifiable features by first dividing an image into N patches of dimensions $P * P$ $(x_p^1, \dots, x_p^N)$. Next, the trainable parameters $\boldsymbol{E_{lin}}$, $\boldsymbol{E_{pos}}$, and $\boldsymbol{T_{class}}$ are used extract the token embeddings z_0 as described in Equation 1. The token embeddings are then used as input to the Transformer Encoder layer. The Transformer Encoder consists of alternating Multi-Head Self-Attention (MSA) layers and Multi-Layer Perceptron (MLP) layers. Each layer is preceded by Layer Normalization (LN) (Ba, Kiros, and Hinton 2016) and succeeded by a residual connection. The MLP in each encoder is a simple feed-forward neural network with 2 hidden layers activated using a GELU non-linearity (Hendrycks and Gimpel 2016). The ViT architecture employs L encoders connected in series. Equations 2 and 3 describe the calculations performed by the Transformer Encoder layers.

$$z_0 = \left[\boldsymbol{T_{class}};\ x_p^1\boldsymbol{E_{lin}}; \dots; x_p^N\boldsymbol{E_{lin}}\right] + \boldsymbol{E_{pos}} \tag{5.1}$$

$$z_l' = \mathrm{MSA}\big(\mathrm{LN}(z_{l-1})\big) + z_{l-1} \qquad l = 1 \dots L \tag{5.2}$$

$$z_l = \mathrm{MLP}\big(\mathrm{LN}(z_l')\big) + z_l' \qquad l = 1 \dots L \tag{5.3}$$

The final module in the ViT architecture is an MLP that serves as a feature classifier. The MLP classifier extracts the classification token vector from the final Transformer Encoder layer ($z_L[0]$). The original ViT model feeds the token vector directly to the softmax-activated output layer. However, the number of layers and nodes in the proposed model's MLP classifier were fine-tuned in the model training phase. The output vector y^{ViT} has the same size as the number of target classes C and each value y_c^{ViT} is the probability that the input image belongs to the class c. The largest probability in the output vector indicates the index of the detected class. In addition, the value of the probability serves as a confidence score.

$$y^{ViT} = \mathrm{MLP}(z_L[0]) \tag{5.4}$$

During the fine-tuning process, the ViT model weights were optimized using backpropagation through time. Similar to the fine-tuning procedure followed by (Dosovitskiy et al. 2020), Stochastic Gradient Descent (SGD) was used to adjust the trainable model weights. Additionally, to fit the model's classification objective, categorical cross entropy was used as the loss function.

5.2.2 Spatial Self-Attention

The Spatial Self-Attention (SSA) network adds spatial context awareness by modeling the geospatial relationship between M ViT detection results using Multi-Head Self Attention. Self-Attention (SA) (Liu et al. 2021) computes the pairwise attention value for the M inputs by determining the similarity score between each pair of input vectors. Each token y_i^{ViT}, which represents image i in the input sequence, is assigned query (q_i), key (k_i), and value (v_i) representation vectors. To calculate the attention scores for image representation vector y_i^{ViT} the alignment scores between y_i^{ViT} and all other input images in the sequence y_j^{ViT} are calculated as a function of q_i and k_j and v_i. During the training phase, this computation allows the self-attention network to learn the similarity between all pairs of ViT image scores in the input sequence. Therefore, the SA network quantifies each image's influence on all other image classifications in the sequence. Consequently, the trained model augments the image representation vectors with spatial contextual relevance. The SSA network generates an output sequence of size M that enhances the output of the stand-alone ViT results using the learned spatial context. Equations 5, 6, and 7 demonstrate the computations carried out by the Multi-Head Self-Attention layers. The internal size of each attention head is D, and H is the number of self-attention heads in the MSA

layer. $\boldsymbol{W_Q}$, $\boldsymbol{W_K}$, $\boldsymbol{W_V}$, and $\boldsymbol{W_O}$ are the query, key, value, and output trainable parameters, respectively.

$$\mathrm{MSA}(X) = [\mathrm{SA}^1(X); \dots; \mathrm{SA}^H(X)]\boldsymbol{W_O} \tag{5.5}$$

$$\mathrm{SA}^h(X) = softmax\left(\frac{\mathrm{Q}^h {K^h}^\top}{\sqrt{D}}\right)V^h \qquad h = 1 \dots H \tag{5.6}$$

$$(\mathrm{Q}^h, K^h, V^h) = (\boldsymbol{W_Q^h}X, \boldsymbol{W_K^h}X, \boldsymbol{W_V^h}X) \qquad h = 1 \dots H \tag{5.7}$$

The SSA network takes the detection result vectors of M Vision Transformers as input. It then adds spatial context awareness by modeling the geospatial relationship between different ViT detection results. The network was implemented using a series of S Multi-Head Self-Attention layers, each preceded by Layer Normalization (Ba, Kiros, and Hinton 2016). The outputs of the final MSA layer are passed through M softmax-activated Fully Connected (FC) layers, which act as the output layer. The FC layers were constructed using the learnable parameters ($\boldsymbol{W_{FC}^1}, \dots, \boldsymbol{W_{FC}^M}$). The output vectors $(y_1, \dots, y_M)$ represent the detection results for segments 1 to M respectively. Equations 8, 9, and 10 demonstrate the computations carried out by the Spatial Self-Attention network to obtain the output at each segment. Figure 5.2 illustrates the structure of the Spatial Self-Attention network.

$$z_0^{SSA} = (y_1^{ViT}, \dots, y_M^{ViT}) \tag{5.8}$$

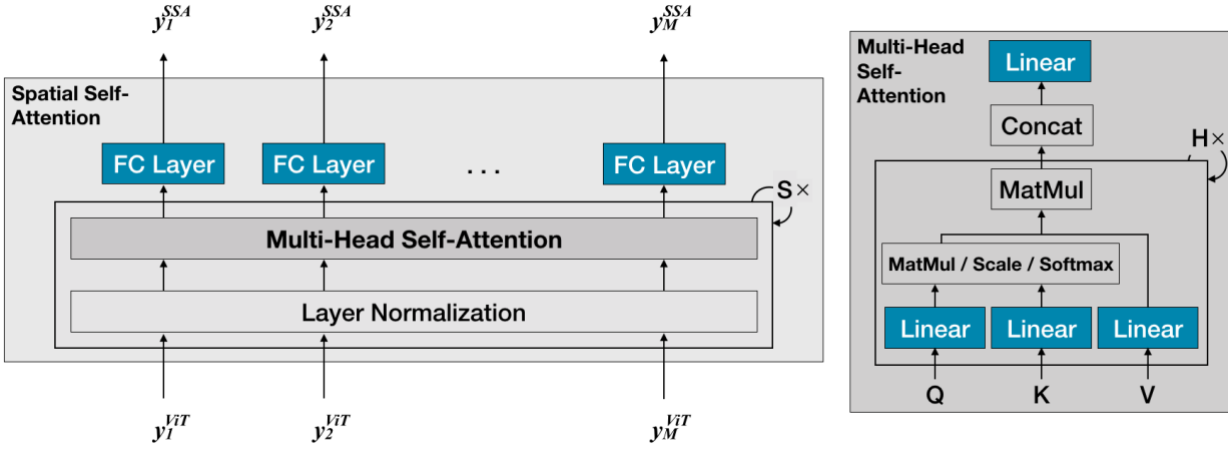
$$z_s^{SSA} = \mathrm{MSA}\left(\mathrm{LN}(z_{s-1}^{SSA})\right) \qquad s = 1 \dots S \tag{5.9}$$

$$y_M = \boldsymbol{W_{FC}^m} z_S^{SSA} \qquad m = 1 \dots M \tag{5.10}$$

The SSA network trainable weights were adjusted using backpropagation through time. Categorical cross entropy was used as a loss function as described in equation 11. N is the number of samples and y_i is a one-hot-encoded vector of size (M, C) where M is the number of segments

and C is the number of target classes. $\hat{y}_i$ is the corresponding predicted output vector Moreover, the Adam optimizer (Kingma and Ba 2014) was used to adjust the model weights during the training phase.

$$L(y, \hat{y}) = -\sum_{i=1}^{N}\sum_{m=1}^{M}\sum_{c=1}^{C} y_{imc} \cdot \log(\hat{y}_{imc}) \tag{5.11}$$



(a) (b)

Figure 5.2 (a) Spatial Self-Attention Model architecture (b) Multi-Head Self-Attention network structure

5.3 Data Description

The proposed network was trained and tested on a self-established image dataset collected from roadside CCTV cameras. Each image had 2 associated class labels, one for rain and the other for road surface conditions. It is unrealistic that rainy conditions are paired with dry pavements. Hence, rainy images were automatically considered to have wet road surface condition. In summary, the collected images were divided between 4 class pairs: *no_rain + dry_road*, *no_rain + wet_road*, *light_rain + wet_road*, and *heavy_rain + wet_road*.

In order to isolate the rain detection output results, the class probabilities of the *no_rain + dry_road* and *no_rain + wet_road* labels were aggregated. The resulting output vector has size 3 and represent each of the rain condition classes *heavy_rain, light_rain, and no_rain*. Similarly, to set apart the road surface condition detection output, the class probabilities of *no_rain + wet_road, light_rain + wet_road,* and *heavy_rain + wet_road* were combined. The aggregated vector has size 2, representing the road condition classes *wet_road and dry_road*. The extracted class labels identify hazardous driving conditions such as heavy rain, which impacts visibility, or wet road surface condition, which negatively affects vehicle maneuverability during and after rainy conditions.

Two separate image datasets were created: 1) a stand-alone image dataset to fine-tune the Vision Transformer network and 2) a sequential image dataset to train the Spatial Self-Attention network. Both datasets were divided into 50% training set, 25% validation set, and 25% testing set. The stand-alone image dataset contains 10,000 images collected using CCTV traffic cameras deployed on multiple freeways in Orlando, Florida. The image labels were distributed equally between the four class pairs. The corresponding output for each image was a one-hot-encoded vector of size 4, one for each class pair. The resolution of the captured images was 1920x1080.

To create the sequential image dataset, the collected images were structured into a vector of shape (M, image width, image height, color channels) where M is the number of adjacent road segments in the detection sequence. Each datapoint contained M images that were captured from the segments at the same instant. The sequential image dataset was captured from 10 roadside traffic cameras on the SR-417 freeway in Orlando, Florida. The cameras were roughly spaced 1 mile apart. The dataset contains 3712 datapoints comprised of 37,120 images. The class distribution of the images was 7870, 8488, 10466, 10296 for the class pairs *heavy_rain +*

wet_road, light_rain + wet_road, no_rain + dry_road, no_rain + wet_road, respectively. The output for each datapoint has dimensions $(M, 4)$: a one-hot-encoded vector representing a class pair for each image in the sequence.

The described multi-dataset approach has two advantages. Firstly, the ViT model was trained and tested on a diverse roadside image dataset and was not restricted to the images used for the SSA network. This prevents the ViT model from overfitting a less diverse image set which was collected using a limited number of cameras. Secondly, the ViT was trained on a dataset that contained a uniform distribution of samples per class. This eliminates potential model bias that can be caused by an imbalanced dataset.

5.4 Experimentation

5.4.1 Setup

5.4.1.1 Vision Transformer

The proposed model utilizes the ViT-B/16 network architecture devised by Dosovitsky et al. (Dosovitskiy et al. 2020). The ViT-B/16 model uses 16x16 pixel patches ($P = 16$). The network employs an intermediate representation vector of size $D = 768$ and contains $L = 12$ Transformer Encoder layers. Furthermore, the Multi-Head Self-Attention layers in the encoders use $H = 12$ Self-Attention heads. The input image size was set to 512x512. Hence, the number of patches N is equal to 1024. Transfer learning was utilized by loading model weights that were pretrained on the Imagenet21k dataset (14M images, 21k target classes) and fine-tuned on the Imagenet dataset (1.3M images, 1k target classes). To fine-tune the transformer for rain and road condition detection, all pre-trained weights were frozen except for the final MLP feature classifier. To achieve the best possible detection results, the model's MLP hyperparameters were tuned.

Table 5.1 lists the tuned hyperparameters and their corresponding search spaces. Image augmentation was performed on the training set by performing random rotations, translations, flips, and zooms. Image augmentation reduces overfitting by introducing geometric intra-class variance. The model was trained using the Adam optimizer.

Table 5.1 ViT model's MLP hyperparameters

Hyperparameter	**Range**	**Step**
Batch size	[32, 128]	32
Number of layers	[1, 10]	1
Number of nodes	[32, 256]	32
Learning rate	[10^{-4}, 10^{-2}]	0.5 (log scale)

The ViT model was compared to other convolution-based image classifiers namely VGG16, ResNet50, and InceptionV3. Similar to the ViT model setup, all convolution-based models were pretrained on the Imagenet dataset and fine-tuned on the collected image dataset. Image augmentation transformations were applied to the training set. For all convolution-based models the pretrained weights were frozen and then connected to a trainable MLP, and each set of hyperparameters was tuned separately.

5.4.1.2 Spatial Self-Attention

The Spatial Self-Attention network employs $S = 2$ Multi-Head Self-Attention layers, each containing $H = 4$ SA heads. The size of the attention heads was regarded as a hyperparameter and tuned using a search space with a range between [256, 1024] and a step size of 256. Other hyperparameters, namely batch size and learning rate, were tuned using the same search space described in Table I. The model performance was compared against the following sequence-to-sequence algorithms:

- ***Moving Average (MA):*** The MA algorithm computes the final output by moving a weighted sliding window across the detection results generated by adjacent ViTs. Window sizes of width 3, 5, and 7 were tested with different weight distributions. The best results were achieved with a sliding window of width 5.
- ***Bidirectional LSTM (Bi-LSTM):*** Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber 1997) is a class of Recurrent Neural Networks (RNNs) that was designed to model sequential data. Each sequential layer in the network has a bidirectional connection to both the preceding and succeeding sequential layer. The utilized Bi-LSTM used the detection results from the M ViTs as input.
- ***Bidirectional GRU (Bi-GRU):*** Gated Recurrent Unit (GRU) (Cho et al. 2014) is another class of RNNs. The internal structure of the GRU node is different than that of the LSTM. The Bi-GRU model also utilizes a bidirectional connection between its sequential layers.

5.4.2 Evaluation Metrics

Three common classification metrics were used to evaluate the performance of the proposed models and compare their performance against the baseline algorithms. These metrics are precision, recall, and F1-score. The F1-score is a measure of accuracy. It's computed as the harmonic mean of the precision and recall. The precision, recall, and F1-score metrics can only handle binary classification. Therefore, the weighted-averages of the one-vs-all scores for each class were calculated.

5.4.3 Detection Results

Table 5.2 lists the results of the different image classification algorithms on the stand-alone image testing dataset. The results indicate that deeper network architectures perform better, highlighting the complexity of the classification tasks. The Vision Transformer model produced

superior results compared to convolution-based deep learning architectures. The nature of the rain and road condition detection classification tasks requires attending to global image features and to the relationship between those features. A rainy image might consist of multiple, decentralized visual features such as dark sky, water splash around vehicles, reflective road surface, and rain drops on camera lens. The Self-Attention mechanisms in the ViT architecture allow the model to attend to global image features and to extract the relations between them starting at the lowest layer. Additionally, different attention heads can utilize different size attention spans, allowing the model to concurrently attend to local and global features starting the earliest layers. In contrast, convolution-based architectures utilize small kernels that extract local features from the unprocessed image at early layers and later apply global convolutions on deep image features towards the end layers.

Table 5.2 Classification results on the stand-alone image dataset

Model	**Rain Condition**			**Road Surface Condition**		
	Precision	**Recall**	**F1-Score**	**Precision**	**Recall**	**F1-Score**
VGG16	0.8855	0.8536	0.8631	0.8767	0.9098	0.8912
ResNet50	0.8968	0.8605	0.8704	0.8792	0.9146	0.8945
InceptionV3	0.9135	0.8932	0.9014	0.9006	0.9272	0.9127
ViT	**0.9215**	**0.9127**	**0.9169**	**0.9258**	**0.9370**	**0.9312**

All models consistently perform better on the road surface condition classification task compared to the rain detection task. This is because the 2-level road condition classification task is inherently easier than the 3-level rain detection problem. Visually, the inter-class variance between dry and wet roads is larger than the inter-class variance between heavy rain, light rain, and even no rain in wet conditions right after rain has ceased.

Table 5.3 demonstrates the performances of different sequence-to-sequence classification algorithms when applied to the detection results of the ViT models. The first row details the

classification result of the stand-alone ViT model on the sequential image dataset. The stand-alone ViT results represent the sequence-modeling-free baseline. The image lists in each datapoint were unraveled and used to assess the performance of the stand-alone ViT. The results serve as the baseline score of stand-alone rain and road surface condition detection. The subsequent rows in Table 5.3 indicate that adding a sequential component to combine the results of individual ViTs improves classification performance. Applying a moving average enhanced the detection results, however not as much as the more elaborate baseline models, which illustrates the intricacy of the sequence-to-sequence detection task. The Bi-LSTM and Bi-GRU models produced comparable classification results, with the Bi-GRU model obtaining a slight edge. The proposed Sequential Self-Attention model produced superior results compared to the moving average and RNN models. The addition of an SSA network improved the F1-score of the stand-alone ViT models by 5.61% and 5.97% for the rain and road condition classification tasks, respectively.

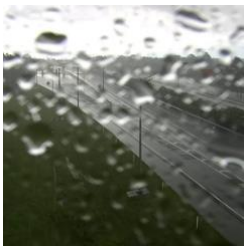
Table 5.3 Classification results on the sequential image dataset

Model	**Rain Condition**			**Road Surface Condition**		
	Precision	**Recall**	**F1-Score**	**Precision**	**Recall**	**F1-Score**
ViT	*0.9168*	*0.9063*	*0.9113*	*0.9156*	*0.9280*	*0.9210*
ViT + MA	0.9174	0.9254	0.9209	0.9410	0.9660	0.9510
ViT + Bi-LSTM	0.9463	0.9483	0.9471	0.9728	0.9764	0.9746
ViT + Bi-GRU	0.9447	0.9544	0.9512	0.9732	0.9803	0.9767
ViT-SSA	**0.9672**	**0.9677**	**0.9674**	**0.9751**	**0.9870**	**0.9807**

Table 5.4 demonstrates samples from a sequential image datapoint at positions 2 – 5. The images' corresponding stand-alone ViT and ViT-SSA classification outputs and confidence scores are also displayed. The samples were collected during a heavily rainy day. The stand-alone ViT was able to correctly classify the images at positions 2, 3, and 5. However, the image at position 4 was not as clear as the surrounding cameras. Consequently, it was misclassified as *no_rain*. The

ViT-SSA was able to rectify this misclassification case. Given the heavy rain conditions of the surrounding segments, the model predicted that the segment at position 4 was heavily raining as well. Furthermore, since it was able to look at multiple images at the same time, the classification confidence score of the ViT-SSA model was higher compared to the stand-alone ViT.

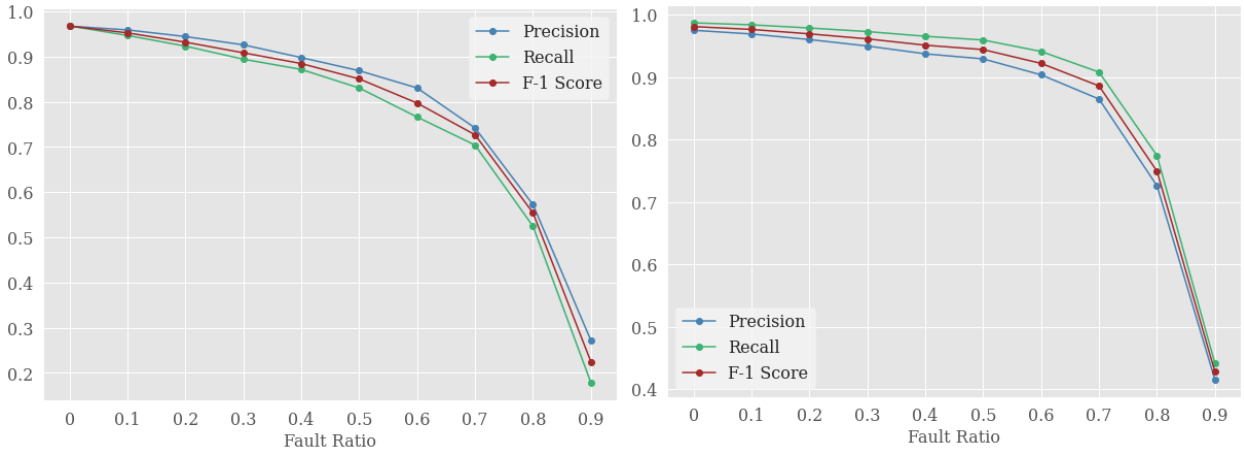
Table 5.4 Sample images from a sequential image datapoint and their corresponding ViT and ViT-SSA classifications and confidence scores

Position	**2**	**3**	**4**	**5**
Image				
Ground Truth	**heavy_rain wet_road**	**heavy_rain wet_road**	**heavy_rain wet_road**	**heavy_rain wet_road**
ViT Output	heavy_rain (c = 0.892) wet_road (c = 0.986)	heavy_rain (c = 0.804) wet_road (c = 0.983)	*no_rain (c = 0.788)* wet_road (c = 0.992)	heavy_rain (c = 0.959) wet_road (c = 0.988)
ViT-SSA Output	heavy_rain (c = 0.997) wet_road (c = 0.998)	heavy_rain (c = 0.989) wet_road (c = 0.997)	heavy_rain (c = 0.994) wet_road (c = 0.998)	heavy_rain (c = 0.982) wet_road (c = 0.992)

5.4.4 Fault Tolerance

Any system that depends on image sequences from roadside traffic cameras is prone to receiving faulty inputs or missing values. Cameras might not transmit images due to sensor failure, network error, or scheduled maintenance. Figure 5.3 illustrates the robustness of the proposed methodology to missing values. Input images were randomly removed from the sequential image dataset with varying fault ratios. A fault ratio of 0.1 indicates that 10% of the images in the test set were removed. The trained ViT-SSA model was subsequently used to detect the generated datasets. The missing inputs to the SSA module were imputed by linearly interpolating the detection results of the ViT instances. For rain detection, Figure 5.3 (a) illustrates that the model

was able to maintain an F1-score above 0.9 for a fault ratio of up to 0.3. Furthermore, Figure 5.3 (b) demonstrates that road condition detection is even more fault tolerant. The model maintains an F1-score above 0.9 for a fault ratio of up to 0.6.



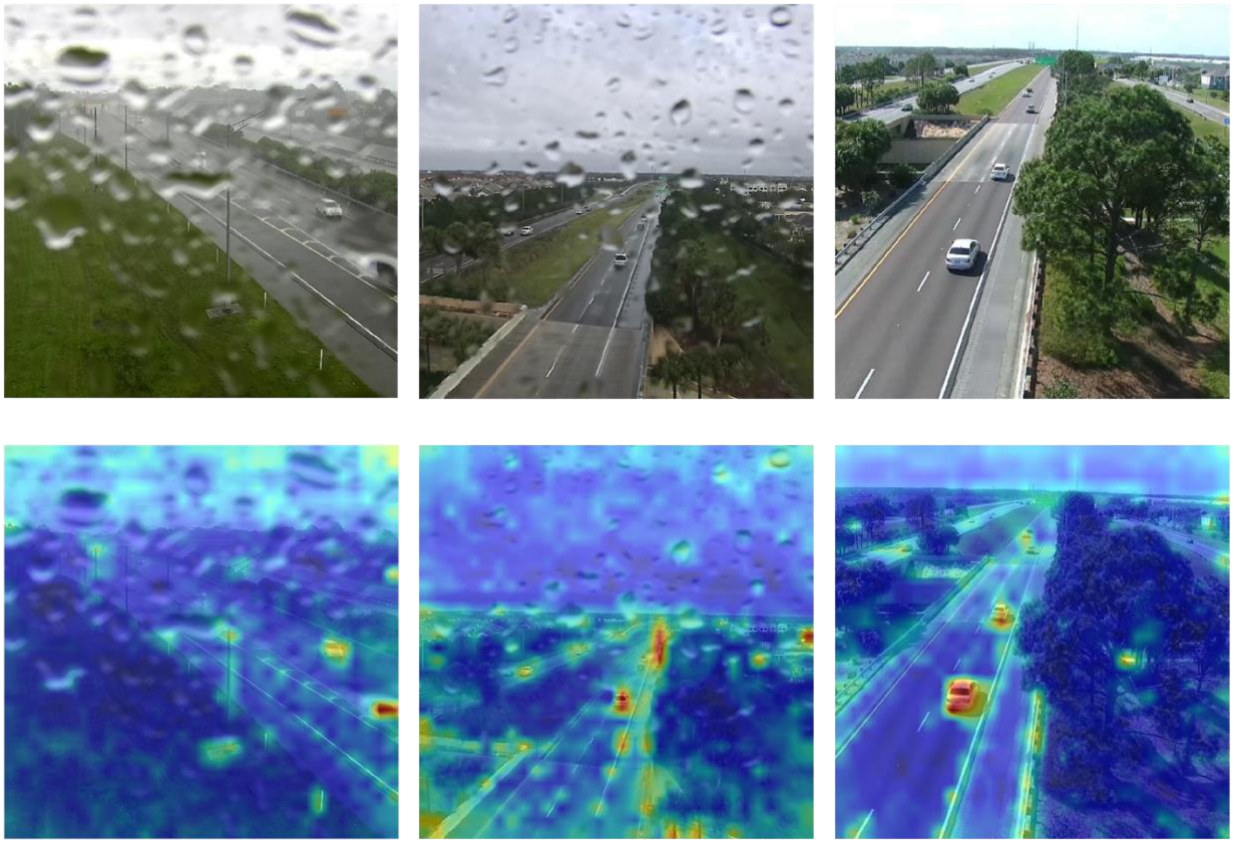
(a) (b)

Figure 5.3 Performance of ViT-SSA on sequential image dataset with missing values for (a) rain and (b) road surface condition detection

5.4.5 ViT Attention Visualization

Figure 5.4 demonstrates sample input images and the corresponding ViT attention heatmaps. The ViT attention values were obtained using the attention rollout technique proposed by Abnar et al. (Abnar and Zuidema 2020). For each MSA layer, the weights of all attention heads were averaged, and the computed average attention weights were recursively applied to the input image. The purpose of the attention visualization is to understand the visual cues that prompted the model to make its classification decisions, and to confirm that the model attends to visual features that are semantically valid. As illustrated in Fig. 4, the ViT model was able to attend to multiple, decentralized raw visual features that are relevant to rain and road condition detection. The model focused on localized features such as the individual rain drops on the camera lens,

splash around vehicles, road surface reflectivity, and as well as more general features such as sky condition and sharpness of the horizon. The ViT's internal self-attention mechanism allows the model to integrate high-level information across the entire image starting from the first attention layer.



(a) (b) (c)

Figure 5.4 ViT activation visualization in (a) heavy rain + wet road, (b) no rain + wet road, and (c) no rain + dry road conditions

5.4.6 SSA Attention Visualization

The SSA attention values were obtained using the same attention rollout technique described in the previous section. Figure 5.5 demonstrates the mean attention activations of all datapoints in the sequential image testing dataset. The attention map provides a general illustration

of how the SSA network combines the classification scores of the independent ViTs. The attention map shows that all attention weights are greater than 0, revealing that the network factors in ViT scores from all segments. The SSA network applies increasingly strong attention weights to closer segment clusters and applies even higher attention weights to neighboring segments. Expectedly, the highest attention weights are distributed across the diagonal, which indicates that the SSA network focuses most on the scores of the ViT at position m to compute the output at segment m.

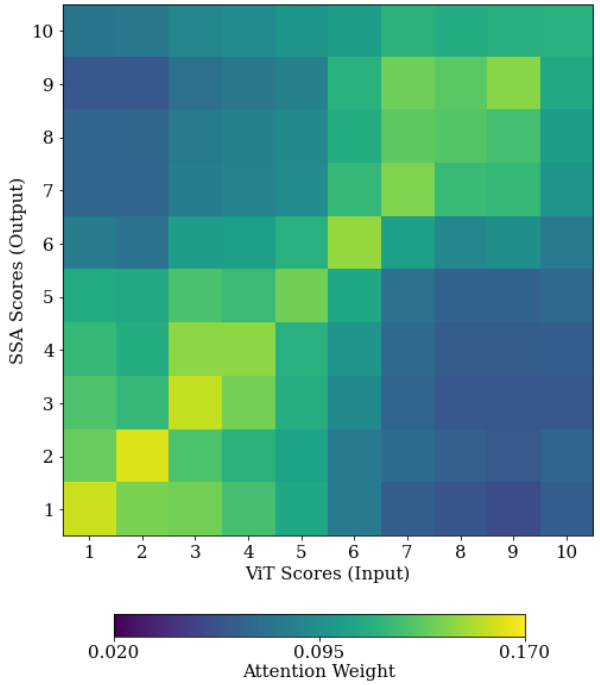
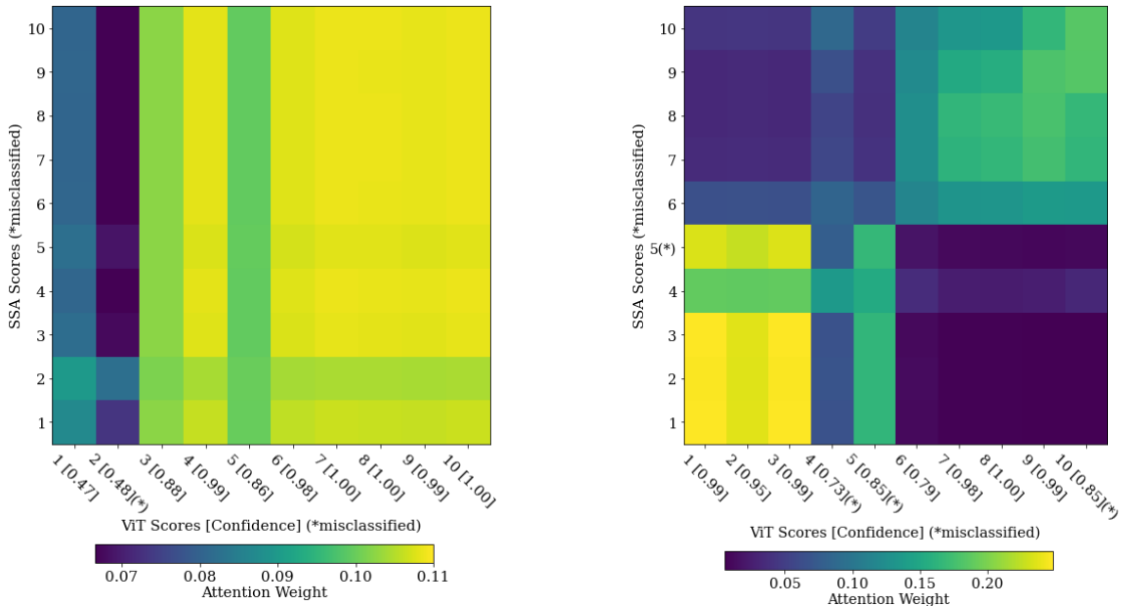


Figure 5.5 Mean attention plot of testing set MSA layer weights

Figure 5.6 illustrates the attention maps of 2 datapoints from the sequential image testing set. In the first data point, the ground truth of the entire sequence belonged to the classes no_rain and dry_road. Figure 5.6 (a) demonstrates how the SSA network can correct the classification errors committed by the stand-alone ViTs. Due to the detected sequential anomaly and the low confidence score at segment 2, the SSA assigned a lower attention weight to its corresponding ViT result and more weight to neighboring ViT scores. Furthermore, the SSA suppressed the

contribution of the ViT scores at segment 2 to the rest of the sequence. Similarly, the SSA network assigned lower attention weights to the inputs coming from segments 1, 3, and 5 due to their lower ViT confidence scores. Figure 5.6 (b) demonstrates how the SSA network improved the depicted datapoint's classification accuracy from 70% to 90%. The ground truth classes for segments 1 to 5 indicated heavy_rain + wet_road, and no_rain + wet_road for segments 6 to 10. The class pair sequence describes a situation where the rain was gradually stopping along the road segments. The SSA network was able to determine the infliction point and distribute 2 clusters of attention weights accordingly. It can be observed that the SSA output at segment 4, which was misclassified by the ViT, was computed from an attention weight distribution drawn from its surrounding cluster. Similarly, lower ViT classification confidence scores at segments 6 and 10 prompted the SSA to assign a dilated attention weight distribution at these locations.



(a) (b)

Figure 5.6 SSA attention map of (a) a datapoint with a misclassified stand-alone ViT output and (b) a datapoint captured when the rain was gradually stopping along the road segments

5.5 Conclusions

In this research effort, a pre-trained Vision Transformer model was fine-tuned on a self-established image dataset for rain and road surface condition detection on freeways using traffic CCTV cameras. Experiments indicated that the ViT model yields superior classification performance when compared to CNN-based models for both detection tasks. Furthermore, the research work presents a novel sequence-to-sequence technique to perform rain and road surface condition detection using a series of adjacent traffic cameras. A Spatial Self-Attention network was proposed to leverage the geographical distribution of the traffic cameras by capturing the relationship between their image classification results, thus adding spatial context awareness to the stand-alone ViT outputs. Experiments proved that adding a spatial component to combine the independent outputs enhances classification performance. The proposed SSA was compared to other RNN-based models and was able to achieve a higher F1-score. In addition to boosting performance, the ViT-SSA combination exhibited robustness to potential missing images in the input sequence. Moreover, the self-attention layers utilized in both parts of the proposed methodology enhanced the interpretability of the ViT's visual classification features and the SSA's sequence modeling behavior.

The proposed methodology can provide an economical method to monitor rain and road surface condition in real-time and with high geospatial granularity. Potentially dangerous driving conditions such as lowered visibility due to heavy rain and the consequent slippery road surface that persists after rain stops can be observed. The model utilizes images which are generated from pre-existing wide-spread traffic CCTV camera infrastructure, eliminating the need for expensive mass deployment of hardware components. The methodology can be used to provide real-time

weather-related route advisory to drivers and connected vehicles. It can consequently suggest safer and less congested routing options. The algorithm can also be used to support traffic operation decisions such as variable speed limits and evacuation plan strategies. One drawback of the proposed study is the experimentation on a limited number of sequential segments, which was circumscribed by the limited access to adjacent CCTV camera images. Given image data from a larger CCTV network, further work should investigate the performance of the proposed methodology on larger spatial networks.

CHAPTER 6: REAL-TIME VIDEO-BASED TRAFFIC INCIDENT IDENTIFICATION USING ROADSIDE CCTV CAMERAS

6.1 Introduction

Real-time identification of traffic incidents is critical for traffic safety and operations. Swift incident response actions can curtail crash-related fatalities and injuries, reduce the risk of secondary crashes, mitigate incident-related congestion, and reduce the exposure of first responders (Burgess, Garinger, and Carrick 2021). Roadside traffic CCTV cameras are ubiquitously deployed on roads and can be utilized for rapid incident detection. As a traffic sensor, video cameras are suitable for real-time detection problems since they are capable of high frequency sampling (usually between 15-30 frames per seconds). Video cameras generate a stream of images, which, compared to the outputs of other traffic sensors such as microwave or loop detector readings, are high-dimensional and rich feature spaces.

Nonetheless, video-based traffic incident detection poses many challenges. Firstly, traffic incidents are defined by the Federal Highway Administration (FHWA) as "unplanned roadway events that affect or impede the normal flow of traffic" (FHWA 2021). There are high variabilities between cases that constitute traffic incidents. They can range from vehicle collisions, which require immediate attention to minimize fatalities and injuries, to stalled/abandoned vehicles, which induce traffic turbulence and increase the risk of secondary crashes (Chimba et al. 2014). Secondly, the compilation of a traffic video dataset that contain traffic incidents from a roadside perspective is a challenging task. The accessibility of public-facing roadside CCTV data is restricted due to sensitivity and privacy concerns. In addition, the manual collection of videos using private cameras is difficult due the scarcity of traffic incidents, especially crashes. Thirdly,

different camera perspectives, lighting condition, and weather can affect detection performance. Fourthly, traffic cameras are usually Pan-Tilt-Zoom (PTZ) cameras, which entails that their viewpoints are not static. PTZ camera perspectives are subject to constant rotations, translations, and scaling. This eliminates the possibility adding manually annotated markings such as interest zones or virtual loops, which can simplify the incident detection process. Finally, most deployed traffic CCTV cameras are low-resolution, which limits the performance of advanced computer vision algorithms.

Various research efforts focused on using videos to detect traffic incident. Many authors used pixel-based approaches such as optical flow algorithms to estimate motion fields in the video stream (Maaloul et al. 2017; Ahmadi, Tabandeh, and Gholampour 2016). Other research efforts used object detection and tracking to model normal vehicle trajectories and identify abnormal paths (Chakraborty, Sharma, and Hegde 2018; Yu et al. 2018). However, state-of-the-art traffic incident detection performance has been demonstrated by top-scoring teams for the NVIDIA AI City Challenge – Traffic Anomaly Detection Track (Naphade et al. 2021). To tackle the high inter-class variability of traffic incident cases, many authors deduced that traffic incidents lead to stationary vehicles where traffic is expected to flow. Using this hypothesis, previous efforts were able to detect traffic incidents with high accuracy (Zhao et al. 2021; Li et al. 2020). However, since the challenge does not require participants to detect incidents in real time, most proposed methods employed high-computation procedures and post-processing techniques to optimize detection performance. Since these methods are not suitable for real-time deployment, they have limited applications for Traffic Incident Management (TIM) programs and Traffic Management Center (TMC) operators and practitioners.

In this study, a methodology has been proposed to detect traffic incidents in videos with a

focus on real-time applicability. The proposed algorithm adopts the stationary vehicle identification technique, which produced the best detection results in the literature. Computation time was assessed, and no post-processing techniques were utilized. Additionally, to demonstrate the usability of the proposed method, the algorithm incident detection delay was evaluated, which is an important factor to minimize incident response time in real-world scenarios. In summary, the contribution of this work is threefold:

1) A video-based traffic incident identification algorithm was proposed. The performance of the algorithm was evaluated using low-resolution roadside CCTV videos collected from US-based roads.
2) The average detection delay, which refers to the time between traffic incident occurrence and traffic incident discovery, was examined and assessed to ensure the practicality of the proposed methodology in real-world scenarios.
3) The method's applicability in real-time was demonstrated by calculating the average computation time and ensuring that the processing throughput of the algorithm is less than the video frame rates of roadside traffic CCTV cameras.

6.2 Methodology

6.2.1 Overview

The proposed methodology was split into two main stages: preprocessing and real-time detection. The preprocessing stage was responsible for automatically extracting a movement mask, creating an expected occupancy heatmap, and compiling the video bounding box statistics. The preprocessing phase used the first N^{pre} frames of a video segment to extract the required assets. During the real-time detection phase, the video background was extracted, and subsequently,

stationary vehicles were detected, tracked, and investigated. The real-time incident identification module used the data extracted from the preprocessing phase, as well as the data computed in real-time, and determined whether a traffic incident has occurred. Figure 6.1 illustrates the overview of the proposed methodology.

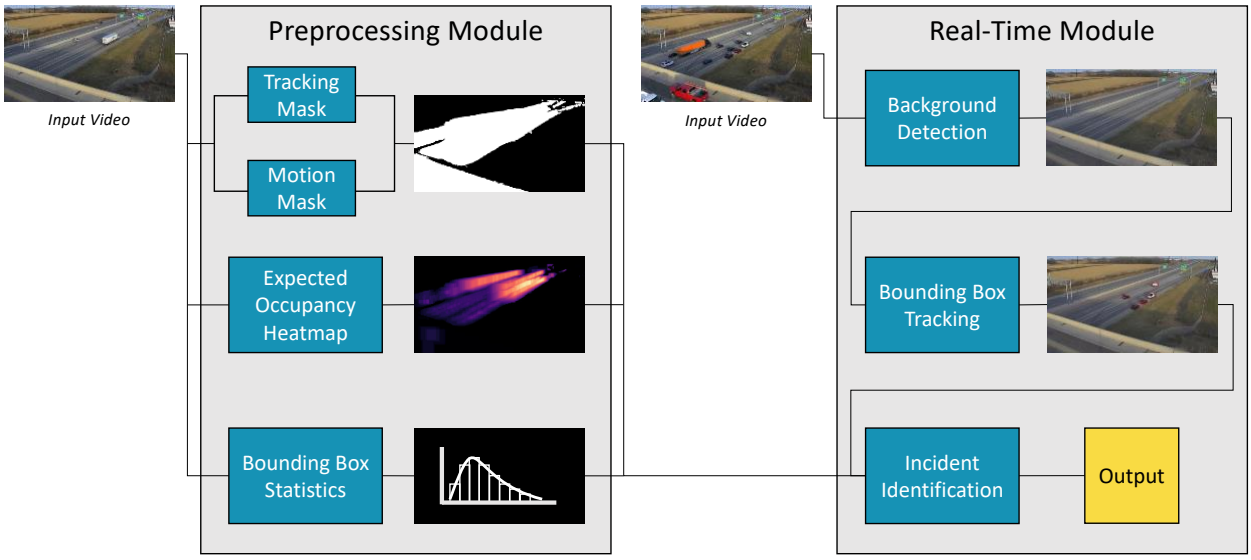


Figure 6.1 Overview of proposed traffic incident detection methodology

6.2.1.1 Object Detection

Object detection is a major component used in multiple places throughout the proposed methodology. To detect vehicles in a given frame, the You Only Look Once (YOLOv4) algorithm was utilized (Bochkovskiy, Wang, and Liao 2020). The YOLOv4 model was chosen for its accuracy and real-time processing capability. Furthermore, the model was pre-trained on the Microsoft COCO dataset, which contains vehicle labels such as “car”, “motorbike”, “bus”, and “truck” (Lin et al. 2014).

6.2.2 Preprocessing

6.2.2.1 Movement Mask Extraction

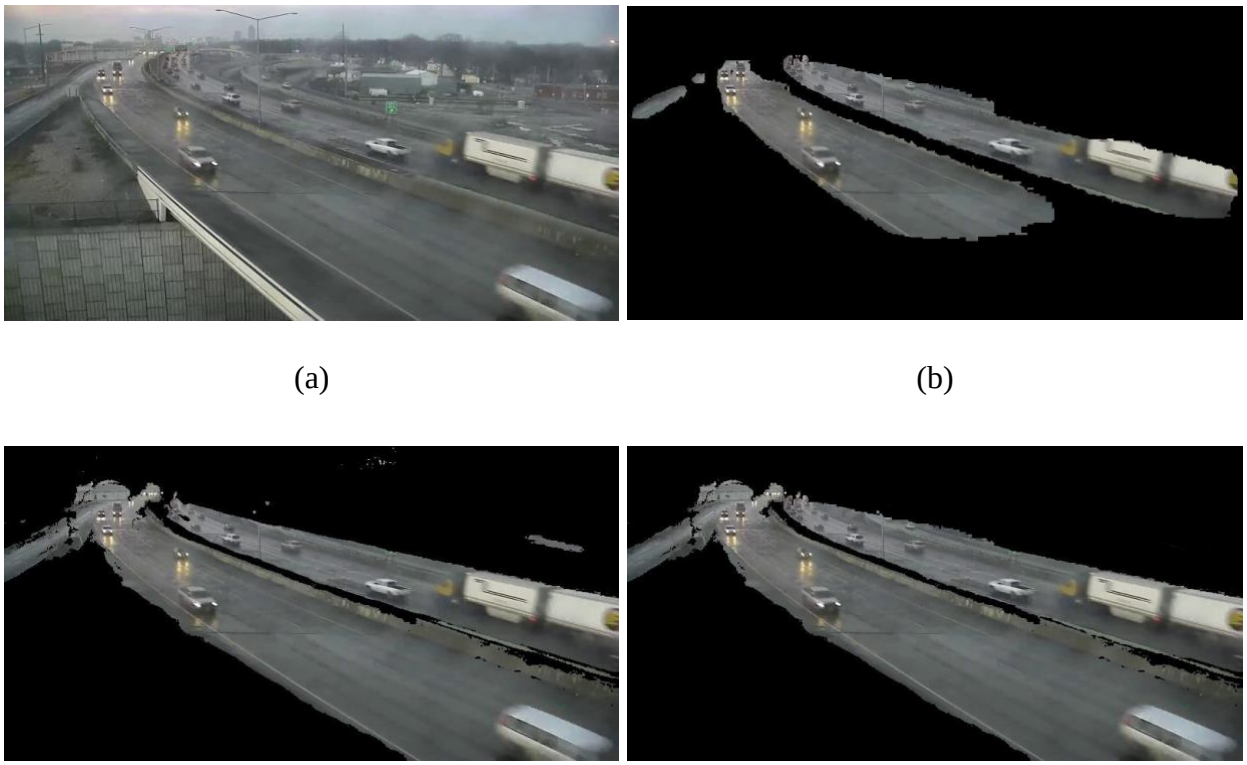
In general, traffic anomalies occur on or close to the main road driving areas. To eliminate the detection noise generated by static vehicles in nearby residential parking or stationary trucks in truck rest areas, a movement mask was automatically extracted from the traffic video scene. The purpose of the movement mask was to identify the area in the video scene where vehicles are expected to be perpetually flowing. To compute the mask, 2 different techniques were adopted: tracking-based mask extraction and motion-based mask extraction.

Tracking-based masks were constructed by extracting the trajectories of vehicles in the preprocessing video segment. The target was to identify the areas in the video frame that contain moving vehicles. Objects were detected using the pre-trained YOLOv4 model. Detected objects that were classified as "car", "bus", "truck", or "motorbike" were then tracked using the deepSORT algorithm (Wojke, Bewley, and Paulus 2017). To capture moving objects of interest, the difference between the first and last centroid of each trajectory was computed and objects that moved for a distance less than the threshold value $\mathrm{TH}^{distance}$ were discarded. Moreover, to suppress the effect of erroneous detections, trajectories with a total length less than TH^{length} were also eliminated.

Motion-based masks were formulated using the real-time foreground/background segmentation algorithm proposed by KadewTraKuPong et al. (KaewTraKulPong and Bowden 2002). Their procedure uses Expectation-Maximization-optimized Adaptive Gaussian Mixture Models to model the probability of each pixel's value through the input video frames. A gaussian blur was applied to the output segmentation mask to alleviate the error caused by salt-and-pepper

noise. For each frame in the preprocessing video segment, the areas of the segmented foreground blobs were recorded on the motion-based mask.

While both mask extraction methods were effective on their own, each suffered from particular shortcomings. The tracking-based mask was fully dependent on the outputs of the detection and tracking algorithms. Hence, misclassified vehicles, occluded objects, and missed detections introduced error into the extracted mask. On the other hand, the motion-based mask did not depend on vehicle detection and thus did not suffer from this problem. However, the motion-mask was prone to errors caused by camera shaking due to camera pole vibration or inclement weather. It was also prone to errors caused by vehicles moving too slowly on congested roadways. Both drawbacks were atypical of the tracking-based mask. To obtain a superior motion mask, both masks were combined. Figure 6.2 demonstrates an example frame and the extracted tracking-based mask, motion-based mask, and combined mask. The tracking-based mask was deficient due to several missed detections, while the motion mask-based mask contained some undesired areas. The combined mask generated the best-fitting movement mask.



(a) (b)

(c) (d)

Figure 6.2 (a) Example video frame and the resulting (b) tracking-based mask, (c) motion-based mask, and (d) combined mask

6.2.2.2 Expected Occupancy Heatmap

Stationary vehicles on roadways do not necessarily indicate traffic incidents. Vehicles stop for traffic control operations such as traffic signals and stop signs. Vehicles might also slow down or stop temporarily during heavy congestion or while merging onto the main road. To accommodate these situations, a waiting period must be set before signaling an incident to avoid false alarms. On the other hand, if an extended grace period for stationary vehicles was set everywhere, the detection delay for traffic incidents on free-flowing roads would be unnecessarily high. To avoid errors caused by the aforementioned scenarios, an expected occupancy heatmap was created. The objective of the expected occupancy heatmap was to keep track of the maximum

expected amount of time for vehicle to remain stationary in a particular area of the video frame. To achieve this task, an occupancy matrix $OCC \in \mathbb{R}^{T \times W \times H}$ was created where T is the number of trajectories and W and H are the width and height of the input video frame, respectively. For each trajectory t extracted from the preprocessing segment, each bounding box was annotated in the occupancy matrix OCC^t. Finally, the expected occupancy heatmap was extracted by computing the maximum value at each pixel along the T-axis.

6.2.2.3 Bounding Box Statistics

The low-resolution nature of roadside traffic CCTVs hinders the performance of the YOLOv4 object detection algorithm. Computing bounding box statistics is a simple yet effective method for reducing the resulting false positives. The target of this step was to identify the "normal" range of bounding box sizes, and consequently eliminating boxes with extreme dimensions. During the preprocessing phase, the YOLOv4 bounding box widths and heights were aggregated into two lists. For each dimension, the 75th percentile value (Q_3^{dim}) and the interquartile range (IQR^{dim}), were computed. The extreme value upper-limit thresholds were set according to equation 1. Due to the nature of diminishing vehicle sizes as they drive away from the camera, no restrictions on the lower limits were set.

$$\mathrm{TH}^{dim} = \mathrm{Q}_3^{dim} + 3 \times \mathrm{IQR}^{dim} \qquad dim \in \{width, height\} \qquad (6.1)$$

6.2.3 Real-time Incident Identification

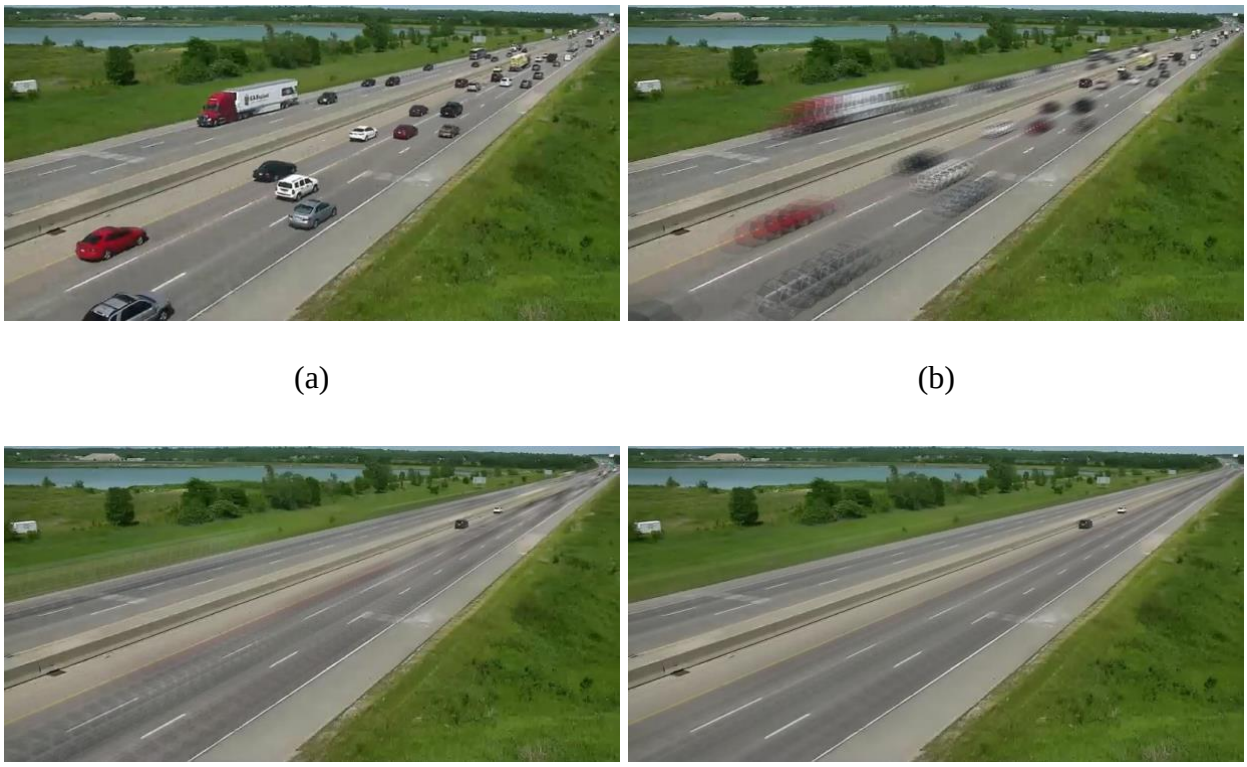
6.2.3.1 Background Detection

The background detection step was developed to erase moving objects from the image. Consequently, vehicles that came to a stop blended into the background. To achieve this task, a background frame queue of length N^q was created. When the queue was at maximum capacity,

the background was initialized by averaging the frames in the queue as shown in equation 2. Afterwards, the background at each successive frame $i > N^q$ was updated by removing the first frame in the queue and appending frame i as shown in equation 3. Figure 6.3 depicts an example frame and the extracted backgrounds at different values of N^q. Selecting an appropriate value for N^q was essential. If the value was too small, it triggered false detections. If it was too large, it had an adverse effect on the incident detection delay.

$$\text{background}_{N^q} = \frac{1}{N^q}\sum_{i=0}^{N^q} frame_i \tag{6.2}$$

$$\text{background}_i = \text{background}_{i-1} - \frac{1}{N^q} frame_{i-N^q} + \frac{1}{N^q} frame_i \tag{6.3}$$



(a) (b)

(c) (d)

Figure 6.3 (a) Example video frame and the extracted backgrounds at queue lengths (b) 15 frames (1.5s), (c) 150 frames (5s), and (d) 300 frames (10s)

6.2.3.2 Bounding Box Tracking

After extracting the background frame, the next step was to identify stationary vehicles. The YOLOv4 object detection algorithm was employed to detect the bounding boxes of vehicles in the extracted background frames. When a vehicle was detected, its bounding box was tracked over time, and a record of the detection frame number was made. If bounding boxes in consecutive background frames had an intersection over union (IoU) value greater than the threshold TH^{IoU}, they were considered the same vehicle. The bounding box tracking step maintained a list of bounding boxes and their respective starting frame numbers.

6.2.3.3 Traffic Incident Identification

Finally, the traffic incident identification step combined the preprocessed assets and real-time elements to make the traffic incident detection decision. Firstly, bounding boxes with dimensions that lied outside the normal range were discarded. Next, the bitwise intersection between each detected bounding box and the combined movement mask was checked. Bounding boxes were discarded if their intersection with the mask yielded zero pixels. The expected occupancy of each detected static vehicle was determined by computing the maximum value of the bitwise intersection between the bounding-box-in-question and the expected occupancy heatmap. Finally, if the bounding box age was greater than the expected occupancy, a traffic incident was triggered.

6.3 Experimentation

6.3.1 Data Description

The methodology was evaluated using 100 videos from the 2021 NVIDIA 5th AI City Challenge – Track 4 (Naphade et al. 2021). Each video was 15 minutes long and had a resolution of 810x450 sampled at 30 frames per second. The video data was collected from roadside traffic CCTVs deployed in the state of Iowa. The dataset contains 56 different traffic incidents, including single and multiple vehicle crashes, and stalled vehicles. Furthermore, the videos contained complex scenes such as congested roads, traffic lights on auxiliary roads, and peripheral residential areas.

A non-incident datapoint is defined as any traffic video segment that does not contain a traffic incident. Therefore, the total number of non-incident datapoints in the dataset is not limited to the videos where no traffic incidents ensue. Incident misdetections that occur before or after

traffic incidents are also considered false alarms. Hence, the aggregate number of non-incident instances (true negatives) in the dataset is innumerable.

6.3.2 Setup

For each video, the first $N^{pre} = 3600$ frames (120s) were used as preprocessing segment. The segment was used to compute the movement mask, the expected occupancy heatmap, and the bounding box statistics. The YOLOv4 model confidence score threshold was set to 0.2 to account for the low video resolution. The distance threshold for the tracking-based mask extraction step $\mathrm{TH}^{distance}$was set to 10px and the length threshold TH^{length} was set to 10 detections. To detect the background frame in the real-time stage, the queue length N^q was specified as 300 frames (10s). Finally, the bounding box tracking IoU threshold TH^{IoU} was set to 0.95.

6.3.3 Evaluation Metrics

Two common evaluation metrics were considered for measuring the methodology's detection accuracy: sensitivity and false alarm rate (FAR). Sensitivity measures the percentage of incidents that were successfully detected out of all incidents in the dataset, while the false alarm rate measures the ratio of non-incident scenarios that were flagged as traffic incidents. The sensitivity and false alarm rate were computed according to equations 4 and 5, respectively.

$$\text{sensitivity} = \frac{True\ Positive}{True\ Positive + False\ Negative} \tag{6.4}$$

$$\text{false alarm rate} = \frac{False\ Positive}{True\ Positive + False\ Positive} \tag{6.5}$$

To measure incident detection speed, delay mean absolute error (MAE) was calculated. The delay MAE assesses the average amount of time between the occurrence and the detection of an incident. Delay MAE was calculated according to equation 6, where t_{det} was the detection

time, t was the actual time of the traffic incident, and n_{det} was the total number of detected incidents. Finally, the mean computation time per frame was measured to evaluate the real-time applicability of the proposed methodology. The computation time of each frame was measured using the processor's internal clock.

$$\text{delay mean absolute error} = \frac{1}{n_{det}} \sum_{i=0}^{n} |t_{det} - t| \tag{6.6}$$

6.3.4 Detection Accuracy

Table 6.1 specifies the values of the different evaluation metrics obtained when the proposed methodology was applied to the video dataset. The proposed method detected traffic incidents in the video dataset with a sensitivity of 85.71% and a false alarm rate of 11.10% while maintaining an average detection delay of 27.53 seconds. Furthermore, Table 6.2 describes the confusion matrix of the output detection. The method detected 48 out of the 56 incidents in the dataset and misclassified 6 non-incident cases as traffic incidents. Due to the unbounded nature of non-incident cases, the total number of true negatives were not counted.

Table 6.1 Detection results

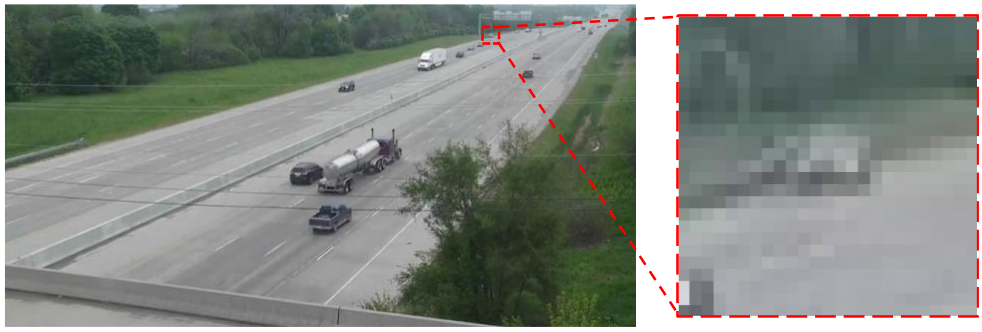
Sensitivity	**FAR**	**Delay MAE (s)**	**Computation Time (s) / (fps)**
0.8571	0.1110	27.53	0.01585 / 63.16

Table 6.2 Incident detection confusion matrix

Actual / **Predicted**	**Positive**	**Negative**
Positive	48	6
Negative	8	-

Figure 6.4 depicts 2 cases of traffic incidents that were missed by the proposed method (false negatives). Given the low-resolution of the CCTV cameras, stationary vehicles that were positioned far away from the camera sensors were too small to be detected. The vehicle sizes in Figure 6.4 (a) and (b) are $7px \times 6px$ and $6px \times 5px$, respectively. As shown in the figure, the roadside camera sensors were unable to retain the vehicle shapes or contours in the output frames. Hence, the YOLOv4 model failed to extract enough features from the few available pixels to detect and classify the objects as vehicles.

(a)



(b)

Figure 6.4 Snapshots of missed traffic incidents (false negatives)

6.3.5 Detection Delay

The proposed method detected traffic incidents with a delay MAE of 27.53 seconds. Table

6.3 describes the distribution of the incident delay values. The standard deviation was 22.07 seconds, and the maximum delay was 101.48 seconds. Additionally, the first, second, and third quartile values were 12.41s, 21.32s, and 42.26s, respectively. The method was able to identify 86.94% of the detected traffic incidents in under 60 seconds.

Table 6.3 Descriptive statistics of incident detection delay

	Delay (s)
Mean	27.53
Standard Deviation	22.07
Minimum	4.01
25th-percentile	12.41
50th-percentile	21.32
75th-percentile	42.26
Maximum	101.48

A major contributing component to the value of detection delay was the background queue length N^q. Figure 6.5 plots the relationship between N^q and the sensitivity, false alarm rate, and delay. As shown, lower N^q values resulted in a reduced average detection delay. However, the false alarm rate rapidly increased as more non-incident cases such as vehicles on congested roads start appearing in the extracted background. Furthermore, the sensitivity remained constant, indicating that the background does not affect the detection performance with respect to true incidents.

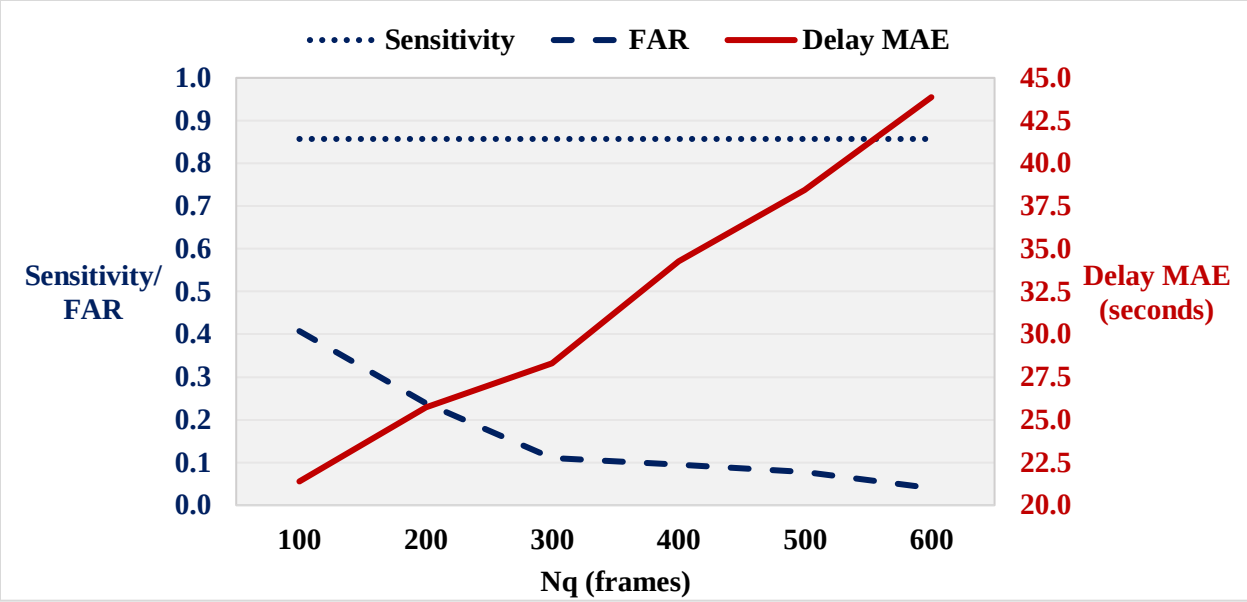


Figure 6.5 Effect of background queue length on sensitivity, FAR, and delay MAE

6.3.6 Computation Time

The experiment was conducted using the YOLOv4 Tensorflow implementation (Hùng 2020) and was executed on an NVIDIA GeForce RTX 2080 Ti Graphics Processing Unit (GPU). Additionally, the TensorRT library was utilized for accelerated inference-time performance (NVIDIA 2018). As shown in Table 6.1, the average computation time per frame was 0.01585 seconds. The resulting processing rate was 63.16 frames per seconds, which is consistent with the reported inference speed of the YOLOv4 model given the same hardware setup (Bochkovskiy, Wang, and Liao 2020). Given that the input video frame rate was 30 frames per second, which is less than the processing throughput, it can be entailed that the proposed methodology is capable of handling real-time video streams.

6.4 Conclusions

In this research effort, a methodology for real-time traffic incident identification using low-

resolution roadside CCTV cameras was proposed. Previous methods in the literature with state-of-the-art detection results were not suitable for real-time detection as many authors utilized post-processing techniques to optimize detection accuracy. In this research effort, the state-of-the-art methodologies were adapted for real-time detection. The proposed algorithm relied entirely on computations that can be executed on-the-fly. Using traffic videos that were collected on US roadways, the proposed method detected traffic incidents with a sensitivity of 85.71%, and a false alarm rate of 11.10%. Moreover, since incident discovery time is crucial for TIM response strategies, the incident detection delay was measured. The average detection delay was 27.53 seconds, and the model was able to identify 86.94% of the detected incidents in under 60 seconds. Furthermore, the method computation time was assessed. The processing throughput was demonstrated to be higher than the input video frame rate, which indicated that the proposed method is capable of handling real-time traffic video streams.

Real-time video-based traffic incident detection can vastly increase the scope of monitored road segments without exhausting manpower resources. Prompt identification of traffic incidents can reduce the risk of fatalities or injuries, reduce the risk of secondary crashes, and mitigate crash-related traffic congestion. Since most roadways already have an extensive roadside CCTV camera network, the proposed methodology can be utilized without adding supplementary hardware. Furthermore, the method requires no manual input. Hence, by developing an automatic recalibration strategy for the preprocessing assets, the algorithm can adapt to modifications in the video traffic scene such changing traffic speed, or changes in the camera perspective such as PTZ camera movements.

CHAPTER 7: CONCLUSIONS

7.1 Summary and Conclusions

In this dissertation, the application of advanced deep learning algorithms using spatiotemporal traffic data for several traffic state estimation objectives was investigated. Particularly, three research objectives related to data-driven traffic state estimation were identified and explored. Chapter 3 tackled the first research objective, network-wide traffic parameters estimation and prediction. In this research effort, the underutilization of longer-term periodic traffic characteristics in the literature, particularly daily and weekly traffic behavior, was highlighted. Additionally, the lack of focus on model interpretability in traffic speed prediction literature was identified as a research gap. The research effort proposed a novel neural network architecture: Attention-based Multi-Encoder-Decoder (Att-MED) and explored its utilization for network-wide traffic speed prediction. The proposed architecture comprised of multiple components. Firstly, the encoder module utilized convolutional LSTMs to capture each sequence's spatiotemporal behavior. The Att-MED architecture employed 3 encoders to capture short-term, daily, and weekly traffic characteristics. The 3 encoder outputs were combined and weighed using the attention layer and subsequently forwarded the decoder. Finally, the LSTM decoder modeled the temporal relationship of the output sequence. The proposed methodology was trained end-to-end to predict up to 60-minutes ahead traffic speed using data collected from SR-408 in Orlando, Florida. The model output was evaluated against other baseline models and proved to be superior. The attention layer served as an intermediate evaluator between the model encoders and decoder. Thus, in order to interpret the model output, the attention weights were visualized. The mapped attention weights confirmed the significance of the daily and weekly sequential input, especially

for longer prediction horizons.

Chapter 4 continued to address the traffic parameters estimation and prediction research objective. Most research efforts in the literature focused on static sensor-based traffic parameters estimation and prediction. To address this shortcoming, the research described in Chapter 4 explored the utilization of low-penetration probe-vehicle fleet data for network-wide traffic speed and volume estimation and prediction. To achieve this task, a novel sequence-to-sequence neural network named Seq2Seq GCN-LSTM was proposed. The architecture comprised of an encoder network which accepted a short-term sequence of fleet-based traffic parameters, and a decoder network that estimated the full traffic speed and volume from the encoder output. The proposed network employed graph convolutions for network-wide spatial feature modeling and LSTMs for temporal feature modeling. The Seq2Seq GCN-LSTM model was used for 2 tasks: real-time traffic speed and volume estimation, and up to 60-minutes ahead traffic speed and volume prediction. Despite the under sampled nature of probe-vehicle data, which results in a high variance signal of the real traffic parameters, the proposed method generated volume and speed predictions with an average accuracy of 90.5% and 96.6%, respectively. Furthermore, the model demonstrated robustness against gaussian noise perturbation. In order to measure the effect of the penetration rate on the model accuracy, the number of vehicles in the fleet was manually varied. It was determined that the model was able to maintain traffic volume estimation and prediction performance until a penetration rate of 1.5% within a 15.6% margin of error. Moreover, the model was able to maintain speed estimation and prediction performance when the penetration rate was as low as 0.5% within a 4.2% margin of error.

Chapter 5 describes the research undertaken to investigate vision-based road weather detection, which was this dissertation's second research objective. Road weather has a critical

effect on traffic safety and operations and therefore must be constantly monitored. In this research effort, a road weather dataset, which consisted of freeway images captured from a roadside perspective and annotated with the corresponding 3-level rain and 2-level road surface condition labels, was collected. The dataset was utilized to explore the application of Vision Transformers for rain and road surface condition classification using roadside traffic CCTV cameras. Furthermore, the research effort redefined the problem of road weather classification from a stand-alone computer vision problem to sequence-to-sequence classification problem by utilizing the spatial distribution of neighboring cameras through a Spatial Self-Attention neural network. The ViT model accuracy surpassed the examined CNN-based models. The addition of the SSA module improved the overall model detection accuracy. Furthermore, experiments demonstrated the robustness of the proposed ViT-SSA against potential missing image inputs. The self-attention layer in both the ViT and the SSA networks were mapped to interpret the model results and enhance the explainability of the ViT's visual reasoning and the SSA's sequential modeling decisions.

Chapter 6 delineates the research undertaken to explore and fulfil this dissertation's third research objective: video-based traffic incident detection. Swift incident identification and response is crucial to limit the potential casualties and traffic disturbance. In Chapter 6, a real-time video-based traffic incident identification algorithm was introduced. State-of-the-art incident identification algorithms, which were not designed for live usage, were adapted for real-time application. The algorithm consisted of scene background detection, expected vehicle occupancy heatmap construction, vehicle bounding box statistics computation, and bounding box tracking. The proposed method was tested on a low-resolution roadside traffic incident video dataset collected from US roadways and detected traffic incidents with a sensitivity of 85.71%, and a false

alarm rate of 11.10%. Furthermore, the algorithm's mean detection delay was 27.53 seconds, and the model was able to identify 86.94% of the detected incidents in under 60 seconds. Finally, to confirm the model's real-time applicability, the computation time was measured. The processing throughput was demonstrated to be higher than the input video frame rate, which indicated that the proposed method is capable of handling real-time traffic video streams.

In general, the growing adoption rate of sensor-enabled vehicle on-board units and pedestrian wearable technology will facilitate the sampling and accumulation of more sophisticated, accurate, and granular traffic data features on a larger scale. Traffic big data will continue to fuel the demand for large-scale traffic-data-driven methodologies and applications. Furthermore, the advancement of road user communication paradigms that quickly and efficiently connect vehicles, pedestrians, and infrastructure provides a fertile ground for both real-time and offline data-empowered traffic operation and safety applications. The research proposed in this dissertation taps into the modeling power of deep learning and proves its proficiency for modeling spatiotemporal traffic data. Additionally, it demonstrates the algorithm's applicability in the traffic state estimation technology domain.

7.2 Implications

The network-wide traffic state estimation and prediction algorithms proposed in Chapters 3 and 4 have many valuable applications. The proposed algorithms can predict network-wide traffic information for up to 60 minutes ahead. The proposed modeling techniques, such as employing multi-sequence traffic inputs, can improve the accuracy of existing mapping and navigation software installed on smartphones and connected vehicles. Consequently, individual drivers can utilize these predictions to accurately inform their trip planning decisions and dynamic

navigation in real-time.

Additionally, the proposed models can be useful for traffic management and operation. The 60-minute prediction horizon of the proposed algorithms can provide sufficient time windows for traffic management center operators to mitigate predicted traffic turbulence. In contrast to traditional deep learning models, which are often valued for their performance but criticized for their lack of explainability, the methodology proposed in Chapter 3 provides interpretable visualizations of its decision-making process, which allows operators to make nuanced, informed, and explainable decisions.

The connected-probe-vehicle-based model proposed in Chapter 4 can be utilized to reduce the reliance on static detectors which are susceptible to hardware failures and network outages. For example, it can be used by traffic management center operators to estimate traffic parameters in remote or new construction areas where traffic sensors might not be sufficiently deployed. Alternatively, it can also be implemented during highway construction projects to estimate traffic parameters on unequipped temporary lanes.

Traffic CCTV-based rain and road surface condition detection is an economical way of monitoring potentially dangerous road weather with high geospatial granularity. Chapter 5 describes a methodology for utilizing pre-existing wide-spread CCTV infrastructure to accurately and robustly detect road weather conditions. The proposed methodology can be used to monitor and advise evacuation plans in the cases of weather-related disasters such as hurricanes. It can additionally be used to support traffic operation decisions such as variable speed limits. Moreover, future research efforts can utilize the model's dense road weather output as observations to model traffic safety problems.

Swift automatic video-based incident identification can greatly improve the scope of

monitored road segments without exhausting manpower. The algorithm proposed in Chapter 6 can be used to greatly shorten the response time between incident occurrence, discovery, and response. Furthermore, the methodology utilizes pre-existing camera hardware, and thus provides an economic solution to the problem of traffic incidents identification and verification.

REFERENCES

Abadi, Martín, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, and Michael Isard. 2016. "Tensorflow: A system for large-scale machine learning." In *12th symposium on operating systems design and implementation*, 265-83.

Abdel-Aty, Mohamed A, and Rajashekar Pemmanaboina. 2006. 'Calibrating a real-time traffic crash-prediction model using archived weather and ITS traffic data', *IEEE Transactions on Intelligent Transportation Systems*, 7: 167-74.

Abdelraouf, Amr, Mohamed Abdel-Aty, and Yina Wu. 2022. 'Using Vision Transformers for Spatial-Context-Aware Rain and Road Surface Condition Detection on Freeways', *IEEE Transactions on Intelligent Transportation Systems*.

Abdelraouf, Amr, Mohamed Abdel-Aty, and Jinghui Yuan. 2021. 'Utilizing Attention-Based Multi-Encoder-Decoder Neural Networks for Freeway Traffic Speed Prediction', *IEEE Transactions on Intelligent Transportation Systems*.

Abnar, Samira, and Willem Zuidema. 2020. 'Quantifying attention flow in transformers', *arXiv preprint arXiv:2005.00928*.

Aboah, Armstrong. 2021. "A vision-based system for traffic anomaly detection using deep learning and decision trees." In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4207-12.

Ahmadi, Parvin, Mahmoud Tabandeh, and Iman Gholampour. 2016. 'Abnormal event detection and localisation in traffic videos based on group sparse topical coding', *IET Image Processing*, 10: 235-46.

Ahmed, Mohammed S, and Allen R Cook. 1979. *Analysis of freeway traffic time-series data by using Box-Jenkins techniques*.

Allamano, P, A Croci, and F Laio. 2015. 'Toward the camera rain gauge', *Water Resources Research*, 51: 1744-57.

Amthor, Manuel, Bernd Hartmann, and Joachim Denzler. 2015. "Road condition estimation based on spatio-temporal reflection models." In *German Conference on Pattern Recognition*, 3-15. Springer.

Arceda, Vicente Enrique Machaca, and Elian Laura Riveros. 2018. "Fast car crash detection in video." In *2018 XLIV Latin American Computer Conference (CLEI)*, 632-37. IEEE.

Asif, Muhammad Tayyab, Justin Dauwels, Chong Yang Goh, Ali Oran, Esmail Fathi, Muye Xu, Menoth Mohan Dhanya, Nikola Mitrovic, and Patrick Jaillet. 2013. 'Spatiotemporal patterns in large-scale traffic speed prediction', *IEEE Transactions on Intelligent Transportation Systems*, 15: 794-804.

Ba, Jimmy Lei, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. 'Layer normalization', *arXiv preprint arXiv:1607.06450*.

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. 2014. 'Neural machine translation by jointly learning to align and translate', *arXiv preprint arXiv:1409.0473*.

Bai, Shuai, Zhiqun He, Yu Lei, Wei Wu, Chengkai Zhu, Ming Sun, and Junjie Yan. 2019. "Traffic Anomaly Detection via Perspective Map based on Spatial-temporal Information Matrix." In *CVPR Workshops*, 117-24.

Bewley, Alex, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. 2016. "Simple online and realtime tracking." In *2016 IEEE International Conference on Image Processing (ICIP)*, 3464-68. IEEE.

Biradar, Kuldeep Marotirao, Ayushi Gupta, Murari Mandal, and Santosh Kumar Vipparthi. 2019. 'Challenges in time-stamp aware anomaly detection in traffic videos', *arXiv preprint arXiv:1906.04574*.

Bochkovskiy, Alexey, Chien-Yao Wang, and Hong-Yuan Mark Liao. 2020. 'YOLOv4: Optimal Speed and Accuracy of Object Detection', *arXiv preprint arXiv:2004.10934*.

Bossu, Jérémie, Nicolas Hautière, and Jean-Philippe Tarel. 2011. 'Rain or snow detection in image sequences through use of a histogram of orientation of streaks', *International journal of computer vision*, 93: 348-67.

Burgess, Lisa;, Amy; Garinger, and Grady Carrick. 2021. "Integrating Computer-Aided Dispatch Data with Traffic Management Centers." In.

Cao, Miaomiao, Victor OK Li, and Vincent WS Chan. 2020. "A CNN-LSTM Model for Traffic Speed Prediction." In *2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring)*, 1-5. IEEE.

Castro-Neto, Manoel, Young-Seon Jeong, Myong-Kee Jeong, and Lee D Han. 2009. 'Online-SVR for short-term traffic flow prediction under typical and atypical traffic conditions', *Expert systems with applications*, 36: 6164-73.

Chakraborty, Pranamesh, Anuj Sharma, and Chinmay Hegde. 2018. "Freeway traffic incident detection from cameras: A semi-supervised learning approach." In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, 1840-45. IEEE.

Chandra, Srinivasa Ravi, and Haitham Al-Deek. 2009. 'Predictions of freeway traffic speeds and volumes using vector autoregressive models', *Journal of Intelligent Transportation Systems*, 13: 53-72.

Chang, H, Youngjoo Lee, B Yoon, and Sanghoon Baek. 2012. 'Dynamic near-term traffic flow prediction: system-oriented approach based on past experiences', *IET Intelligent Transport Systems*, 6: 292-305.

Chen, Jingyuan, Guanchen Ding, Yuchen Yang, Wenwei Han, Kangmin Xu, Tianyi Gao, Zhe Zhang, Wanping Ouyang, Hao Cai, and Zhenzhong Chen. 2021. "Dual modality vehicle anomaly detection via bidirectional-trajectory tracing." In *Proc. CVPR Workshops, Virtual*.

Chen, Yu, Yuanlong Yu, and Ting Li. 2016. "A vision based traffic accident detection method using extreme learning machine." In *2016 International Conference on Advanced Robotics and Mechatronics (ICARM)*, 567-72. IEEE.

Chimba, Deo, Boniphace Kutela, Gary Ogletree, Frank Horne, and Mike Tugwell. 2014. 'Impact of abandoned and disabled vehicles on freeway incident duration', *Journal of transportation engineering*, 140: 04013013.

Cho, Kyunghyun, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. 'Learning phrase representations using RNN encoder-decoder for statistical machine translation', *arXiv preprint arXiv:1406.1078*.

Chollet, François. 2015. "keras." In.

Chung, Junyoung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. 'Empirical evaluation of gated recurrent neural networks on sequence modeling', *arXiv preprint arXiv:1412.3555*.

Chung, Whoibin, Mohamed Abdel-Aty, and Jaeyoung Lee. 2018. 'Spatial analysis of the effective coverage of land-based weather stations for traffic crashes', *Applied geography*, 90: 17-27.

Cong, Yang, Junsong Yuan, and Ji Liu. 2011. "Sparse reconstruction cost for abnormal event detection." In *CVPR 2011*, 3449-56. IEEE.

Cui, Zhiyong, Ruimin Ke, Ziyuan Pu, and Yinhai Wang. 2018. 'Deep bidirectional and unidirectional LSTM recurrent neural network for network-wide traffic speed prediction', *arXiv preprint arXiv:1801.02143*.

Dai, Jifeng, Yi Li, Kaiming He, and Jian Sun. 2016. "R-fcn: Object detection via region-based fully convolutional networks." In *Advances in neural information processing systems*, 379-87.

Davis, Gary A, and Nancy L Nihan. 1991. 'Nonparametric regression and short-term freeway traffic forecasting', *Journal of transportation engineering*, 117: 178-88.

Deng, Jia, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. "Imagenet: A large-scale hierarchical image database." In *2009 IEEE conference on computer vision and pattern recognition*, 248-55. Ieee.

Derrow-Pinion, Austin, Jennifer She, David Wong, Oliver Lange, Todd Hester, Luis Perez, Marc Nunkesser, Seongjae Lee, Xueying Guo, and Brett Wiltshire. 2021. "Eta prediction with graph neural networks in google maps." In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, 3767-76.

Dong, Rong, Juan Liao, Bo Li, Huiyu Zhou, and Danny Crookes. 2017. "Measurements of rainfall rates from videos." In *2017 10th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*, 1-9. IEEE.

Doshi, Keval, and Yasin Yilmaz. 2020. "Fast unsupervised anomaly detection in traffic videos." In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 624-25.

———. 2021. "An efficient approach for anomaly detection in traffic videos." In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4236-44.

Dosovitskiy, Alexey, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, and Sylvain Gelly. 2020. 'An image is worth 16x16 words: Transformers for image recognition at scale', *arXiv preprint arXiv:2010.11929*.

Efentakis, Alexandros, Sotiris Brakatsoulas, Nikos Grivas, Giorgos Lamprianidis, Kostas Patroumpas, and Dieter Pfoser. 2013. "Towards a flexible and scalable fleet management service." In *Proceedings of the Sixth ACM SIGSPATIAL International Workshop on Computational Transportation Science*, 79-84.

Ewan, Levi, and Ahmed Al-Kaisy. 2017. "Assessment of Montana Road Weather Information System." In.: Western Transportation Institute, Montana State University.

Farnebäck, Gunnar. 2003. "Two-frame motion estimation based on polynomial expansion." In *Scandinavian conference on Image analysis*, 363-70. Springer.

FHWA. 2020. 'How Do Weather Events Impact Roads?', Accessed 04/22/2021. https://ops.fhwa.dot.gov/weather/q1_roadimpact.htm.

———. 2021. 'Traffic Incident Management (TIM)'. https://ops.fhwa.dot.gov/eto_tim_pse/index.htm].

Fu, Rui, Zuo Zhang, and Li Li. 2016. "Using LSTM and GRU neural network methods for traffic flow prediction." In *2016 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC)*, 324-28. IEEE.

Garrett , J. Kyle, Brenda Boyce, Daniel Krechmer, and William Perez. 2008. "Implementation and Evaluation of RWIS ESS Siting Guide." In.

Girshick, Ross. 2015. "Fast r-cnn." In *Proceedings of the IEEE international conference on computer vision*, 1440-48.

Girshick, Ross, Jeff Donahue, Trevor Darrell, and Jitendra Malik. 2014. "Rich feature hierarchies for accurate object detection and semantic segmentation." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 580-87.

Grabowski, Dariusz, and Andrzej Czyżewski. 2020. 'System for monitoring road slippery based on CCTV cameras and convolutional neural networks', *Journal of Intelligent Information Systems*: 1-14.

Guo, Shengnan, 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*, 922-29.

Hall, Fred L, Brian L Allen, and Margot A Gunter. 1986. 'Empirical analysis of freeway flow-density relationships', *Transportation Research Part A: General*, 20: 197-210.

Hasan, Mahmudul, Jonghyun Choi, Jan Neumann, Amit K Roy-Chowdhury, and Larry S Davis. 2016. "Learning temporal regularity in video sequences." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 733-42.

Haurum, Joakim Bruslund, Chris H Bahnsen, and Thomas B Moeslund. 2019. "Is it Raining Outside? Detection of Rainfall using General-Purpose Surveillance Cameras." In *CVPR Workshops*, 55-64.

He, Kaiming, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. 2017. "Mask r-cnn." In *Proceedings of the IEEE international conference on computer vision*, 2961-69.

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. "Deep residual learning for image recognition." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770--78.

Hecht-Nielsen, Robert. 1992. 'Theory of the backpropagation neural network.' in, *Neural networks for perception* (Elsevier).

Hendrycks, Dan, and Kevin Gimpel. 2016. 'Gaussian error linear units (gelus)', *arXiv preprint arXiv:1606.08415*.

Hochreiter, Sepp, and Jürgen Schmidhuber. 1997. 'Long short-term memory', *Neural computation*, 9: 1735-80.

Hong, Wei-Chiang, Yucheng Dong, Feifeng Zheng, and Shih Yung Wei. 2011. 'Hybrid evolutionary algorithms in a SVR traffic flow forecasting model', *Applied Mathematics and Computation*, 217: 6733-47.

Hospedales, Timothy, Shaogang Gong, and Tao Xiang. 2009. "A markov clustering topic model for mining behaviour in video." In *2009 IEEE 12th International Conference on Computer Vision*, 1165-72. IEEE.

Hui, Zu, Xie Yaohua, Ma Lu, and Fu Jiansheng. 2014. "Vision-based real-time traffic accident detection." In *Proceeding of the 11th World Congress on Intelligent Control and Automation*, 1035-38. IEEE.

Hùng, Việt. 2020. "tensorflow-yolov4-tflite." In. Github Repository, .

Ijjina, Earnest Paul, Dhananjai Chand, Savyasachi Gupta, and K Goutham. 2019. "Computer Vision-based Accident Detection in Traffic Surveillance." In *2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, 1-6. IEEE.

Jin, Yinli, Erlong Tan, Li Li, Guiping Wang, Jun Wang, and Ping Wang. 2018. 'Hybrid traffic forecasting model with fusion of multiple spatial toll collection data and remote microwave sensor data', *IEEE Access*, 6: 79211-21.

KaewTraKulPong, Pakorn, and Richard Bowden. 2002. 'An improved adaptive background mixture model for real-time tracking with shadow detection.' in, *Video-based surveillance systems* (Springer).

Kalman, Rudolph Emil. 1960. 'A new approach to linear filtering and prediction problems'.

Kamarianakis, Yiannis, and Poulicos Prastacos. 2003. 'Forecasting traffic flow conditions in an urban network: Comparison of multivariate and univariate approaches', *Transportation Research Record*, 1857: 74-84.

Ke, Ruimin, Wan Li, Zhiyong Cui, and Yinhai Wang. 2020. 'Two-Stream Multi-Channel Convolutional Neural Network for Multi-Lane Traffic Speed Prediction Considering Traffic Volume Impact', *Transportation Research Record*, 2674: 459-70.

Khan, Asifullah, Anabia Sohail, Umme Zahoora, and Aqsa Saeed Qureshi. 2020. 'A survey of the recent architectures of deep convolutional neural networks', *Artificial intelligence review*, 53: 5455-516.

Kim, Hyunwoo, Seokmok Park, and Joonki Paik. 2020. "Pre-Activated 3D CNN and Feature Pyramid Network for Traffic Accident Detection." In *2020 IEEE International Conference on Consumer Electronics (ICCE)*, 1-3. IEEE.

Kingma, Diederik P, and Jimmy Ba. 2014. 'Adam: A method for stochastic optimization', *arXiv preprint arXiv:1412.6980*.

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

Kratz, Louis, and Ko Nishino. 2009. "Anomaly detection in extremely crowded scenes using spatio-temporal motion pattern models." In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 1446-53. IEEE.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton. 2012. "Imagenet classification with deep convolutional neural networks." In *Advances in neural information processing systems*, 1097-105.

Kuhn, Harold W. 1955. 'The Hungarian method for the assignment problem', *Naval research logistics quarterly*, 2: 83-97.

Kumar, S Vasantha, and Lelitha Vanajakshi. 2015. 'Short-term traffic flow prediction using seasonal ARIMA model with limited input data', *European Transport Research Review*, 7: 21.

Kumaran, Santhosh Kelathodi, Debi Prosad Dogra, and Partha Pratim Roy. 2019. 'Anomaly detection in road traffic using visual surveillance: A survey', *arXiv preprint arXiv:1901.08292*.

Kwon, Jaimyoung, Pravin Varaiya, and Alexander Skabardonis. 2003. 'Estimation of truck traffic volume from single loop detectors with lane-to-lane speed correlation', *Transportation Research Record*, 1856: 106-17.

Lee, Jiwan, Bonghee Hong, Yongdeok Shin, and Yang-Ja Jang. 2016. "Extraction of weather information on road using CCTV video." In *2016 International Conference on Big Data and Smart Computing (BigComp)*, 529-31. IEEE.

Li, Yaguang, Rose Yu, Cyrus Shahabi, and Yan Liu. 2018. "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting." In *ICLR*.

Li, Yang, Dimitrios Gunopulos, Cewu Lu, and Leonidas Guibas. 2017. "Urban travel time prediction using a small number of GPS floating cars." In *Proceedings of the 25th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, 1-10.

Li, Yanshan, Weiming Liu, and Qinghua Huang. 2016. 'Traffic anomaly detection based on image descriptor in videos', *Multimedia tools and applications*, 75: 2487-505.

Li, Yingying, Jie Wu, Xue Bai, Xipeng Yang, Xiao Tan, Guanbin Li, Shilei Wen, Hongwu Zhang, and Errui Ding. 2020. "Multi-granularity tracking with modularlized components for unsupervised vehicles anomaly detection." In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 586-87.

Li, Zewen, Fan Liu, Wenjie Yang, Shouheng Peng, and Jun Zhou. 2021. 'A survey of convolutional neural networks: analysis, applications, and prospects', *IEEE transactions on neural networks and learning systems*.

Lin, Tsung-Yi, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. 2017. "Feature pyramid networks for object detection." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2117-25.

Lin, Tsung-Yi, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. "Microsoft coco: Common objects in context." In *European conference on computer vision*, 740-55. Springer.

Lipton, Zachary C, John Berkowitz, and Charles Elkan. 2015. 'A critical review of recurrent neural networks for sequence learning', *arXiv preprint arXiv:1506.00019*.

Liu, Wei, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. 2016. "Ssd: Single shot multibox detector." In *European conference on computer vision*, 21-37. Springer.

Liu, Yipeng, Haifeng Zheng, Xinxin Feng, and Zhonghui Chen. 2017. "Short-term traffic flow prediction with Conv-LSTM." In *2017 9th International Conference on Wireless Communications and Signal Processing (WCSP)*, 1-6. IEEE.

Liu, Ze, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. 2021. 'Swin transformer: Hierarchical vision transformer using shifted windows', *arXiv preprint arXiv:2103.14030*.

Lucas, Bruce D, and Takeo Kanade. 1981. 'An iterative image registration technique with an application to stereo vision'.

Luo, Weixin, Wen Liu, and Shenghua Gao. 2017. "A revisit of sparse coding based anomaly detection in stacked rnn framework." In *Proceedings of the IEEE International Conference on Computer Vision*, 341-49.

Lv, Yisheng, Yanjie Duan, Wenwen Kang, Zhengxi Li, and Fei-Yue Wang. 2014. 'Traffic flow prediction with big data: a deep learning approach', *IEEE Transactions on Intelligent Transportation Systems*, 16: 865-73.

Ma, Xiaolei, Zhimin Tao, Yinhai Wang, Haiyang Yu, and Yunpeng Wang. 2015. 'Long short-term memory neural network for traffic speed prediction using remote microwave sensor data', *Transportation Research Part C: Emerging Technologies*, 54: 187-97.

Ma, Xiaolei, Haiyang Yu, Yunpeng Wang, and Yinhai Wang. 2015a. 'Large-scale transportation network congestion evolution prediction using deep learning theory', *PloS one*, 10: e0119044.

———. 2015b. 'Large-scale transportation network congestion evolution prediction using deep learning theory', *PloS one*, 10.

Maaloul, Boutheina, Abdelmalik Taleb-Ahmed, Smail Niar, Naim Harb, and Carlos Valderrama. 2017. "Adaptive video-based algorithm for accident detection on highways." In *2017 12th IEEE International Symposium on Industrial Embedded Systems (SIES)*, 1-6. IEEE.

Mahmoud, Nada, Mohamed Abdel-Aty, Qing Cai, and Jinghui Yuan. 2021a. 'Estimating cycle-level real-time traffic movements at signalized intersections', *Journal of Intelligent Transportation Systems*: 1-24.

———. 2021b. 'Predicting cycle-level traffic movements at signalized intersections using machine learning models', *Transportation Research Part C: Emerging Technologies*, 124: 102930.

Manfredi, John, Thomas Walters, Gregory Wilke, Leon Osborne, Robert Hart, Tom Incrocci, and Tom Schmitt. 2005. "Road Weather Information System Environmental Sensor Station Siting Guidelines." In.

Maze, Thomas H, Manish Agarwal, and Garrett Burchett. 2006. 'Whether weather matters to traffic demand, traffic safety, and traffic operations and flow', *Transportation Research Record*, 1948: 170-76.

Naphade, Milind, Ming-Ching Chang, Anuj Sharma, David C Anastasiu, Vamsi Jagarlamudi, Pranamesh Chakraborty, Tingting Huang, Shuo Wang, Ming-Yu Liu, and Rama Chellappa. 2018. "The 2018 nvidia ai city challenge." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 53-60.

Naphade, Milind, Zheng Tang, Ming-Ching Chang, David C Anastasiu, Anuj Sharma, Rama Chellappa, Shuo Wang, Pranamesh Chakraborty, Tingting Huang, and Jenq-Neng Hwang. 2019. "The 2019 AI City Challenge." In *CVPR Workshops*, 452-60.

Naphade, Milind, Shuo Wang, David C Anastasiu, Zheng Tang, Ming-Ching Chang, Xiaodong Yang, Yue Yao, Liang Zheng, Pranamesh Chakraborty, and Christian E Lopez. 2021. "The 5th ai city challenge." In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4263-73.

Naphade, Milind, Shuo Wang, David C Anastasiu, Zheng Tang, Ming-Ching Chang, Xiaodong Yang, Liang Zheng, Anuj Sharma, Rama Chellappa, and Pranamesh Chakraborty. 2020. "The 4th AI City Challenge." In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 626-27.

Ng, Pauline C, and Steven Henikoff. 2003. 'SIFT: Predicting amino acid changes that affect protein function', *Nucleic acids research*, 31: 3812-14.

NVIDIA. 2018. "TensorRT." In.

O'Malley, Tom, Elie Bursztein, James Long, François Chollet, Haifeng Jin, Luca Invernizzi, and others. 2019. 'Keras Tuner'.

Omer, Raqib, and Liping Fu. 2010. "An automatic image recognition system for winter road surface condition classification." In *13th international IEEE conference on intelligent transportation systems*, 1375-79. IEEE.

Ozcan, Koray, Anuj Sharma, Skylar Knickerbocker, Jennifer Merickel, Neal Hawkins, and Matthew Rizzo. 2019. "Road Weather Condition Estimation Using Fixed and Mobile Based Cameras." In *Science and Information Conference*, 192-204. Springer.

Pan, Bei, Ugur Demiryurek, and Cyrus Shahabi. 2012. "Utilizing real-world transportation data for accurate traffic prediction." In *2012 IEEE 12th International Conference on Data Mining*, 595-604. IEEE.

Pan, Guangyuan, Liping Fu, Ruifan Yu, and Matthew Muresan. 2018. 'Winter road surface condition recognition using a pretrained deep convolutional network', *arXiv preprint arXiv:1812.06858*.

———. 2019. "Evaluation of Alternative Pre-trained Convolutional Neural Networks for Winter Road Surface Condition Monitoring." In *2019 5th International Conference on Transportation Information and Safety (ICTIS)*, 614-20. IEEE.

Park, Jungme, Dai Li, Yi Murphey, Johannes Kristinsson, Ryan McGee, Ming Kuang, and Tony Phillips. 2011. "Real time vehicle speed prediction using a neural network traffic model." In *The 2011 International Joint Conference on Neural Networks*, 2991-96. IEEE.

Pfoser, Dieter, Nectaria Tryfona, and Agnes Voisard. 2006. "Dynamic travel time maps-enabling efficient navigation." In *18th International Conference on Scientific and Statistical Database Management (SSDBM'06)*, 369-78. IEEE.

Polson, Nicholas G, and Vadim O Sokolov. 2017. 'Deep learning for short-term traffic flow prediction', *Transportation Research Part C: Emerging Technologies*, 79: 1-17.

Qian, Yiming, Emilio J Almazan, and James H Elder. 2016. "Evaluating features and classifiers for road weather condition analysis." In *2016 IEEE International Conference on Image Processing (ICIP)*, 4403-07. IEEE.

Rämä, Pirkko. 1999. 'Effects of weather-controlled variable speed limits and warning signs on driver behavior', *Transportation Research Record*, 1689: 53-59.

Redmon, Joseph, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2016. "You only look once: Unified, real-time object detection." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 779-88.

Redmon, Joseph, and Ali Farhadi. 2017. "YOLO9000: better, faster, stronger." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 7263-71.

———. 2018. 'Yolov3: An incremental improvement', *arXiv preprint arXiv:1804.02767*.

Ren, Jianqiang, Yangzhou Chen, Le Xin, Jianjun Shi, Baotong Li, and Yinan Liu. 2016. 'Detecting and positioning of traffic incidents via video-based analysis of traffic states in a road segment', *IET Intelligent Transport Systems*, 10: 428-37.

Ren, Shaoqing, Kaiming He, Ross Girshick, and Jian Sun. 2015. "Faster r-cnn: Towards real-time object detection with region proposal networks." In *Advances in neural information processing systems*, 91-99.

Roychowdhury, Sohini, Minming Zhao, Andreas Wallin, Niklas Ohlsson, and Mats Jonasson. 2018. "Machine learning models for road surface and friction estimation using front-camera images." In *2018 International Joint Conference on Neural Networks (IJCNN)*, 1-8. IEEE.

Russakovsky, Olga, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, and Michael Bernstein. 2015. 'Imagenet large scale visual recognition challenge', *International journal of computer vision*, 115: 211-52.

Sadeky, Samy, Ayoub Al-Hamadiy, Bernd Michaelisy, and Usama Sayed. 2010. "Real-time automatic traffic accident recognition using hfg." In *2010 20th International Conference on Pattern Recognition*, 3348-51. IEEE.

Schmidhuber, Jürgen. 2015. 'Deep learning in neural networks: An overview', *Neural networks*, 61: 85-117.

Sharma, Davesh, Mingjian Wu, and Tae J Kwon. 2021. "Safety Effects of Road Weather Information System (RWIS) - A Cost-Benefit Analysis." In *Transportation Research Board 100th Annual Meeting*. Washington DC, United States.

Shi, Xingjian, and Dit-Yan Yeung. 2018. 'Machine learning for spatiotemporal sequence forecasting: A survey', *arXiv preprint arXiv:1808.06865*.

Shine, Linu, and Jiji CV. 2020. "Fractional data distillation model for anomaly detection in traffic videos." In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 606-07.

Simonyan, Karen, and Andrew Zisserman. 2014. 'Very deep convolutional networks for large-scale image recognition', *arXiv preprint arXiv:1409.1556*.

Singh, Dinesh, and Chalavadi Krishna Mohan. 2018. 'Deep spatio-temporal representation for detection of road accidents using stacked autoencoder', *IEEE Transactions on Intelligent Transportation Systems*, 20: 879-87.

Sirirattanapol, Chairath, Masahiko Nagai, Apichon Witayangkurn, Surachet Pravinvongvuth, and Mongkol Ekpanyapong. 2019. 'Bangkok CCTV image through a road environment extraction system using multi-label convolutional neural network classification', *ISPRS International Journal of Geo-Information*, 8: 128.

Sultani, Waqas, Chen Chen, and Mubarak Shah. 2018. "Real-world anomaly detection in surveillance videos." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 6479-88.

Sun, Zhonghua, and Kebin Jia. 2013. "Road surface condition classification based on color and texture information." In *2013 Ninth International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, 137-40. IEEE.

Sun, Zhu, Ping Wang, Yinli Jin, Jun Wang, and Lei Wang. 2020. 'A Practical Weather Detection Method Built in the Surveillance System Currently Used to Monitor the Large-Scale Freeway in China', *IEEE Access*, 8: 112357-67.

Szegedy, Christian, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. "Going deeper with convolutions." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1-9.

Tahir, Muhammad Naeem, and Urooj Rashid. 2020. "Intelligent Transport System (ITS) Assisted Road Weather & Traffic Services." In *2020 IEEE Vehicular Networking Conference (VNC)*, 1-2. IEEE.

Tang, Jinjun, Fang Liu, Yajie Zou, Weibin Zhang, and Yinhai Wang. 2017. 'An improved fuzzy neural network for traffic speed prediction considering periodic characteristic', *IEEE Transactions on Intelligent Transportation Systems*, 18: 2340-50.

Tian, Yongxue, and Li Pan. 2015. "Predicting short-term traffic flow by long short-term memory recurrent neural network." In *2015 IEEE international conference on smart city/SocialCom/SustainCom (SmartCity)*, 153-58. IEEE.

Tran, Du, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. 2015. "Learning spatiotemporal features with 3d convolutional networks." In *Proceedings of the IEEE international conference on computer vision*, 4489-97.

Uijlings, Jasper RR, Koen EA Van De Sande, Theo Gevers, and Arnold WM Smeulders. 2013. 'Selective search for object recognition', *International journal of computer vision*, 104: 154-71.

Ullah, Habib, Mohib Ullah, Hina Afridi, Nicola Conci, and Francesco GB De Natale. 2015. "Traffic accident detection through a hydrodynamic lens." In *2015 IEEE International Conference on Image Processing (ICIP)*, 2470-74. IEEE.

Van Der Voort, Mascha, Mark Dougherty, and Susan Watson. 1996. 'Combining Kohonen maps with ARIMA time series models to forecast traffic flow', *Transportation Research Part C: Emerging Technologies*, 4: 307-18.

Van Lint, JWC, SP Hoogendoorn, and Henk J van Zuylen. 2005. 'Accurate freeway travel time prediction with state-space neural networks under missing data', *Transportation Research Part C: Emerging Technologies*, 13: 347-69.

Van Lint, JWC, and CPIJ Van Hinsbergen. 2012. 'Short-term traffic and travel time prediction models', *Artificial Intelligence Applications to Critical Transportation Issues*, 22: 22-41.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. "Attention is all you need." In *Advances in neural information processing systems*, 5998-6008.

Veni, S, R Anand, and B Santosh. 2020. 'Road Accident Detection and Severity Determination from CCTV Surveillance.' in, *Advances in Distributed Computing and Machine Learning* (Springer).

Veres, Matthew, and Medhat Moussa. 2019. 'Deep learning for intelligent transportation systems: A survey of emerging trends', *IEEE Transactions on Intelligent Transportation Systems*.

Vlahogianni, Eleni, Matthew Karlaftis, and John Golias. 2014. 'Short-term traffic forecasting: Where we are and where we're going', *Transportation Research Part C: Emerging Technologies*, 43: 3-19.

Vu, Nam, and Cuong Pham. 2017. 'Traffic incident recognition using empirical deep convolutional neural networks model.' in, *Context-Aware Systems and Applications, and Nature of Computation and Communication* (Springer).

Wang, Chen, Yulu Dai, Wei Zhou, and Yifei Geng. 2020. 'A vision-based video crash detection framework for mixed traffic flow environment considering low-visibility condition', *Journal of Advanced Transportation*, 2020.

Wang, Gaoang, Xinyu Yuan, Aotian Zheng, Hung-Min Hsu, and Jenq-Neng Hwang. 2019. "Anomaly Candidate Identification and Starting Time Estimation of Vehicles from Traffic Videos." In *CVPR Workshops*, 382-90.

Wang, Jingyuan, Qian Gu, Junjie Wu, Guannan Liu, and Zhang Xiong. 2016. "Traffic speed prediction and congestion source exploration: A deep learning method." In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, 499-508. IEEE.

Wang, Yilun, Yu Zheng, and Yexiang Xue. 2014. "Travel time estimation of a path using sparse trajectories." In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 25-34.

Wei, JiaYi, JianFei Zhao, YanYun Zhao, and ZhiCheng Zhao. 2018. "Unsupervised anomaly detection for traffic surveillance based on background modeling." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 129-36.

Wejo. 2021a. 'Wejo: For Automotive', Accessed 11/30/2021. https://www.wejo.com/for-automotive.

———. 2021b. 'Wejo: Vehicle Movements', Accessed 11/30/2021. https://www.wejo.com/products-vehicle-movements.

Wilkie, David, Jason Sewall, and Ming Lin. 2013. 'Flow reconstruction for data-driven traffic animation', *ACM Transactions on Graphics (TOG)*, 32: 1-10.

Williams, Billy M. 2001. 'Multivariate vehicular traffic flow prediction: evaluation of ARIMAX modeling', *Transportation Research Record*, 1776: 194-200.

Williams, Billy M, and Lester A Hoel. 2003. 'Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: Theoretical basis and empirical results', *Journal of transportation engineering*, 129: 664-72.

Wojke, Nicolai, Alex Bewley, and Dietrich Paulus. 2017. "Simple online and realtime tracking with a deep association metric." In *2017 IEEE international conference on image processing (ICIP)*, 3645-49. IEEE.

Wu, Chun-Hsin, Jan-Ming Ho, and Der-Tsai Lee. 2004. 'Travel-time prediction with support vector regression', *IEEE Transactions on Intelligent Transportation Systems*, 5: 276-81.

Wu, Jie, Xionghui Wang, Xuefeng Xiao, and Yitong Wang. 2021. "Box-level tube tracking and refinement for vehicles anomaly detection." In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4112-18.

Wu, Yuankai, and Huachun Tan. 2016. 'Short-term traffic flow forecasting with spatial-temporal correlation in a hybrid deep learning framework', *arXiv preprint arXiv:1612.01022*.

Wu, Zonghan, Shirui Pan, Guodong Long, Jing Jiang, and Chengqi Zhang. 2019. 'Graph wavenet for deep spatial-temporal graph modeling', *arXiv preprint arXiv:1906.00121*.

Xia, Li-min, Xiang-jie Hu, and Jun Wang. 2018. 'Anomaly detection in traffic surveillance with sparse topic model', *Journal of Central South University*, 25: 2245-57.

Xingjian, Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. 2015. "Convolutional LSTM network: A machine learning approach for precipitation nowcasting." In *Advances in neural information processing systems*, 802-10.

Xu, Mingxing, Wenrui Dai, Chunmiao Liu, Xing Gao, Weiyao Lin, Guo-Jun Qi, and Hongkai Xiong. 2020. 'Spatial-Temporal Transformer Networks for Traffic Flow Forecasting', *arXiv preprint arXiv:2001.02908*.

Xu, Yan, Xi Ouyang, Yu Cheng, Shining Yu, Lin Xiong, Choon-Ching Ng, Sugiri Pranata, Shengmei Shen, and Junliang Xing. 2018. "Dual-mode vehicle motion pattern learning for high performance road traffic anomaly detection." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 145-52.

Ye, Qing, Wai Yuen Szeto, and Sze Chun Wong. 2012. 'Short-term traffic speed forecasting based on data recorded at irregular intervals', *IEEE Transactions on Intelligent Transportation Systems*, 13: 1727-37.

Yu, Bing, Haoteng Yin, and Zhanxing Zhu. 2017. 'Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting', *arXiv preprint arXiv:1709.04875*.

Yu, Byeonghyeop, Yongjin Lee, and Keemin Sohn. 2020. 'Forecasting road traffic speeds by considering area-wide spatio-temporal dependencies based on a graph convolutional neural network (GCN)', *Transportation Research Part C: Emerging Technologies*, 114: 189-204.

Yu, Lijun, Dawei Zhang, Xiangqun Chen, and Alexander Hauptmann. 2018. "Traffic danger recognition with surveillance cameras without training data." In *2018 15th IEEE*

International Conference on Advanced Video and Signal Based Surveillance (AVSS), 1-6. IEEE.

Yuan, Jinghui, Mohamed Abdel-Aty, Yaobang Gong, and Qing Cai. 2019. 'Real-time crash risk prediction using long short-term memory recurrent neural network', *Transportation Research Record*, 2673: 314-26.

Yuan, Yuan, Dong Wang, and Qi Wang. 2016. 'Anomaly detection in traffic scenes via spatial-aware motion reconstruction', *IEEE Transactions on Intelligent Transportation Systems*, 18: 1198-209.

Yun, Kimin, Hawook Jeong, Kwang Moo Yi, Soo Wan Kim, and Jin Young Choi. 2014. "Motion interaction field for accident detection in traffic surveillance video." In *2014 22nd International Conference on Pattern Recognition*, 3062-67. IEEE.

Zen, Remmy, Dewa Made Sri Arsa, Ruixi Zhang, Ngurah Agus Sanjaya Er, and Stéphane Bressan. 2019. "Rainfall Estimation from Traffic Cameras." In *International Conference on Database and Expert Systems Applications*, 18-32. Springer.

Zhan, Xianyuan, Samiul Hasan, Satish V Ukkusuri, and Camille Kamga. 2013. 'Urban link travel time estimation using large-scale taxi data with partial information', *Transportation Research Part C: Emerging Technologies*, 33: 37-49.

Zhan, Xianyuan, Ruimin Li, and Satish V Ukkusuri. 2015. 'Lane-based real-time queue length estimation using license plate recognition data', *Transportation Research Part C: Emerging Technologies*, 57: 85-102.

Zhan, Xianyuan, Satish V Ukkusuri, and Chao Yang. 2016. 'A Bayesian mixture model for short-term average link travel time estimation using large-scale limited information trip-based data', *Automation in construction*, 72: 237-46.

Zhang, Junping, Fei-Yue Wang, Kunfeng Wang, Wei-Hua Lin, Xin Xu, and Cheng Chen. 2011. 'Data-driven intelligent transportation systems: A survey', *IEEE Transactions on Intelligent Transportation Systems*, 12: 1624-39.

Zhang, Zheng, Huadong Ma, Huiyuan Fu, and Cheng Zhang. 2016. 'Scene-free multi-class weather classification on single images', *Neurocomputing*, 207: 365-73.

Zhang, Zhihao, Yunpeng Wang, Peng Chen, Zhengbing He, and Guizhen Yu. 2017. 'Probe data-driven travel time forecasting for urban expressways by matching similar spatiotemporal traffic patterns', *Transportation Research Part C: Emerging Technologies*, 85: 476-93.

Zhao, Ling, 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: 3848-58.

Zhao, Yuxiang, Wenhao Wu, Yue He, Yingying Li, Xiao Tan, and Shifeng Chen. 2021. "Good practices and a strong baseline for traffic anomaly detection." In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 3993-4001.

Zhao, Zheng, Weihai Chen, Xingming Wu, Peter CY Chen, and Jingmeng Liu. 2017. 'LSTM network: a deep learning approach for short-term traffic forecast', *IET Intelligent Transport Systems*, 11: 68-75.

Zhao, Zhong-Qiu, Peng Zheng, Shou-tao Xu, and Xindong Wu. 2019. 'Object detection with deep learning: A review', *IEEE transactions on neural networks and learning systems*, 30: 3212-32.

Zheng, Chuanpan, Xiaoliang Fan, Cheng Wang, and Jianzhong Qi. 2020. "Gman: A graph multi-attention network for traffic prediction." In *Proceedings of the AAAI Conference on Artificial Intelligence*, 1234-41.

Zheng, Fangfang, and Henk Van Zuylen. 2013. 'Urban link travel time estimation based on sparse probe vehicle data', *Transportation Research Part C: Emerging Technologies*, 31: 145-57.

Zhong, Jia-Xing, Nannan Li, Weijie Kong, Shan Liu, Thomas H Li, and Ge Li. 2019. "Graph convolutional label noise cleaner: Train a plug-and-play action classifier for anomaly detection." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1237-46.

Zhu, Li, Fei Richard Yu, Yige Wang, Bin Ning, and Tao Tang. 2018. 'Big data analytics in intelligent transportation systems: A survey', *IEEE Transactions on Intelligent Transportation Systems*, 20: 383-98.

Zhu, Zheng, Bo Peng, Chenfeng Xiong, and Lei Zhang. 2016. 'Short-term traffic flow prediction with linear conditional Gaussian Bayesian network', *Journal of Advanced Transportation*, 50: 1111-23.

Zhu, Ziqi, Li Zhuo, Panling Qu, Kailong Zhou, and Jing Zhang. 2016. "Extreme weather recognition using convolutional neural networks." In *2016 IEEE International Symposium on Multimedia (ISM)*, 621-25. IEEE.