Real-Time Spatio-Temporal Forecasting with Dynamic Urban Event and Vehicle-Level Flow Information*

Chris Conlan[1,†], Joe Oakley[1,†], Gunduz Vehbi Demirci[2], Alexandros Sfyridis[3] and Hakan Ferhatosmanoglu[1,‡]

[1]University of Warwick, Coventry, UK
[2]Imagination Technologies, London, UK
[3]Imperial College London, London, UK

Abstract

Building a real-time spatio-temporal forecasting system is a challenging problem which has many practical applications such as traffic and road network management. Most forecasting research typically focuses on the average quality of predictive models, with much less attention paid to building a practical pipeline and achieving timely and accurate forecasts when the network is under heavy load. Additionally, transport authorities face the issue of how to effectively leverage various dynamic data sources, such as urban events (e.g., scheduled roadworks on the road network, cultural events) and vehicle-level flow data. In this paper, we investigate the practical challenges of real-time forecasting, and present Foresight, a cloud-based system for spatio-temporal forecasting developed in collaboration with Transport for the West Midlands (TfWM). Foresight can ingest, aggregate and process streamed traffic data to produce road network forecasts continuously. We adapt spatio-temporal machine learning methods to incorporate dynamic urban events and vehicle-level flow data, and experimentally evaluate a variety of predictive models in our setting. We employ a data-driven approach to identify peak times in the network, and provide insights on how the performance of forecasting solutions varies for these times when accurate forecasts are most important. We observe that incorporating roadworks into a Graph Neural Network (GNN) model can provide up to a 29.1% performance improvement (MAPE) at a 60-minute forecasting horizon. Further, modelling traffic propagation using vehicle-level flow data in order to support graph-based learning can yield performance gains of 8.8% (MAE) at peak times.

Keywords

Real-Time Traffic Forecasting, Deep Learning, Graph Neural Networks, Dynamic Urban Events, Roadworks Data, Flow Information, Peak Traffic Conditions, Streaming Data

1. Introduction

Traffic data collected at roadside sensors can offer significant value to transport managers. The raw data is typically transformed into a time series format, capturing a metric such as the vehicle count or average speed over the road network. This information can be used to make forecasts about the state of the road network in the near future, which can enable proactive responses when heavy or unusual load on the network is predicted.

A wide range of forecasting approaches have been applied to the traffic prediction task, from statistical methods such as ARIMA [1], to Deep Learning (DL) models such as LSTM [2]. In recent years, Graph Neural Network (GNN) approaches have achieved state-of-the-art results, due to their ability to capture spatial dependencies between sensors [3, 4, 5, 6, 7]. GNNs typically model the road sensor network as a graph structure, whose weighted adjacency matrix reflects the strength of inter-sensor relationships.

Despite an extensive body of work on traffic forecasting, there are still several major challenges in building a practical traffic forecasting system. First, there are challenges around the scalable handling and pre-processing of the streaming traffic data, as well as in its use for real-time forecasting. Typically, most forecasting models are developed offline, without considering the challenges of producing forecasts on streaming data. The real-time forecasting problem requires that the forecasting process takes place continuously within a given time lag of each real-world traffic event occurring. This is an important problem to study for practical data-driven systems, as transport managers need to be able to take action based on responsive short-term forecasts. It has also been identified as an open research issue, and entails significant data management challenges, particularly when DL models are employed [8]. Second, the focus of many previous works is the minimization of the average error across

Proceedings of the Workshop on Big Mobility Data Analytics (BMDA) co-located with EDBT/ICDT 2023 Joint Conference (March 28-31, 2023), Ioannina, Greece

* This publication describes work performed at the University of Warwick, prior to Gunduz Vehbi Demirci joining Imagination Technologies, Alexandros Sfyridis joining Imperial College London, and Hakan Ferhatosmanoglu joining Amazon Web Services.

† These authors contributed equally.

‡ Also with Amazon Web Services.

Chris.Conlan@warwick.ac.uk (C. Conlan); J.Oakley@warwick.ac.uk (J. Oakley); Gunduz.Demirci@imgtec.com (G. V. Demirci); A.Sfyridis@imperial.ac.uk (A. Sfyridis); Hakan.F@warwick.ac.uk (H. Ferhatosmanoglu)

CEUR Workshop Proceedings (CEUR-WS.org)

all test samples. In practice, however, accurate forecasts are most important when the network is under heavy load, and/or when the traffic patterns on the network are unusual compared with historical norms. Finally, in addition to data captured at roadside sensors, dynamic urban events (**DUE**) and vehicle-level flow data should also be incorporated dynamically into forecasting models to improve predictive performance.

Towards addressing these challenges, we present Foresight, a cloud-based spatio-temporal forecasting system developed in collaboration with Transport for the West Midlands (TfWM). Foresight ingests real-time traffic data, and constructs a Flow Aggregated Adjacency Matrix (**FAAM**) based on the observed vehicle-level flow between the network sensors, which supports graph-based learning. To ensure timely inference results, it leverages an automated ingestion, aggregation, and MLOps pipeline. To focus on the times when forecasting is most impactful, we develop a data-driven approach to examine the performance of models at peak times. Finally, we incorporate roadworks/road closures in Foresight. While previous works incorporate roadwork information into predictive models [9, 5], this is usually not done as part of a real-time pipeline. The dynamic processing of these DUEs into a format suitable for continuous inference presents non-trivial challenges.

The contributions of this work are as follows:

- We study the real-time forecasting problem, which seeks to perform inference (and the necessary data aggregation and pre-processing) within a given time lag.
- We take a novel approach to the construction of the adjacency matrix as a core component of graph-based ML models and leverage vehicle-level flow data in order to model traffic propagation through the network. Translating this vehicle-level flow data into a **FAAM** presents non-trivial data management challenges. GNN forecasting errors at peak times are reduced by up to 8.8% under this scheme.
- We explore several approaches for transforming real-time **DUE** data (namely roadworks) into a time series format, and incorporate them into a deep learning spatio-temporal forecasting model. Experiments demonstrate performance improvements of up to 29.1%.
- We present Foresight, a cloud-based system for addressing the real-time traffic forecasting problem, which can continuously stream traffic data, leverage up-to-date roadwork information, and exploit traffic flow patterns to enhance forecasting performance.
- We perform experimental analysis of a variety of time series forecasting methods in a new environment. We compare the performance of multiple models across both normal and peak traffic conditions. We discover that LSTM is often a viable alternative to a GNN based approach; although LSTM produces poorer predictions on average, it can achieve lower MAPE during crucial peak time hours, which is typically more important in practice. This indicates that LSTM yields fewer large outlier errors, and as such may be better at capturing unusual traffic patterns.

The rest of the paper is organized as follows. Section 2 presents related work. Section 3 formalizes the traffic forecasting problem, and describes its real-time extensions. Section 4 illustrates Dynamic Urban Events. Section 5 describes the Flow-based GNN Adjacency Matrix. Section 6 introduces Foresight, our cloud-based real-time forecasting system. Section 7 covers our experimental analysis. Finally, Section 8 concludes the paper.

2. Related Work

Numerous forecasting methods have been applied in the traffic prediction domain. Forecasting is typically undertaken with the use of statistical and machine learning models. The Autoregressive Integrated Moving Average (ARIMA) and its variations are the most widely used time-series models [1, 10, 11]. In addition, many Machine Learning (ML) models have been applied, with the Support Vector Machine (SVM) [12, 13] and the Random Forest [12, 14, 15] being the most common. Deep Learning (DL) solutions based on Artificial Neural Networks have become increasingly popular due to their improved forecasting accuracy and the ability to account for non-linear dependencies [16]. Long Short-Term Memory (LSTM) and Feed Forward Neural Networks (FFNN) are among the models most frequently applied to forecast traffic flows [2, 17, 18], with several hybrid approaches also investigated [19, 20]. Finally, Graph Neural Networks (GNNs), which can capture the spatial dependencies between the traffic monitoring sensors by representing the road network as a graph structure, have further improved prediction accuracy. Hence, multiple GNN applications for traffic flow forecasting have been presented in recent years [3, 4, 5, 6, 7, 21, 8].

2.1. Dynamic Urban Events

Urban events such as roadworks have been shown to significantly impact traffic flow [22, 23]. Hence, the incorporation of auxiliary information about such events can further improve traffic forecasting performance. For example, roadwork and accident information has been utilized in traffic simulation systems, ML models and GNNs [24, 9, 5]. A combination of roadworks and weather con-

ditions have also been added to a bi-directional LSTM Autoencoder for short-term traffic prediction [25].

2.2. GNN Adjacency Matrix

In GNN models, the underlying graph structures are usually represented with an adjacency matrix which represents the spatial relationships between the nodes of a graph [4]. Although GNN adjacency matrices are typically binary [26], multiple variations have been proposed [21]. For example, a real-valued distance-based adjacency matrix is a common alternative for representing the spatial dependencies between nodes, and has been applied in numerous traffic forecasting studies with GNNs [27, 28, 29, 30, 31]. The travel time between nodes has also been considered as an alternative to distance-based metrics [32]. More recently, the integration of dynamic matrices has been introduced, that captures the dynamic changes in spatial dependencies of the graph and tends to improve forecasting [33, 34]. Coarse origin-destination (OD) data has been applied as a substitute for a distance-based adjacency matrix [35]. However, to the best of our knowledge, no previous work on GNN based forecasting has leveraged granular vehicle-level flow data to model inter-sensor relationships.

2.3. Forecasting Systems

In addition to statistical and ML/DL modelling approaches, forecasting systems have also been developed as general tools for time series prediction and road management optimization. For example, the AutoAI for Time Series Forecasting (AUTOAI-TS) [36] automates forecasting techniques and addresses specific requirements for time series data. DeepTRANS [37] combines the DeepTTE system [38] with DCRNN [7] for bus travel time estimation. The system uses archive information about bus and traffic flow from sensor data, and DCRNN is used to estimate traffic speed at buses' locations. The TrafficStream forecasting system leverages GNNs and Continual Learning (CL) [39]. It constructs a sub-graph to capture network expansion, and constraints are applied on the current training model to integrate information from historical data.

3. Real-Time Spatio-Temporal Forecasting

In this section, we first describe the traffic forecasting problem, before introducing its real-time variant. A key requirement of this procedure is that the aggregation, pre-processing and inference of the traffic data must take place within a certain time lag of the real traffic events occurring. These practical aspects of forecasting have attracted relatively little attention in the large body of research on the topic.

3.1. Traffic Forecasting Problem

We first present the definition of the traffic forecasting problem, where the goal is to predict the future state of the road network, given a sequence of previously observed time series readings. Traffic information is typically obtained from roadside sensors, which can capture features such as traffic flow or average speed, to form a (multivariate) time series. Given a set of sensors S, we denote the traffic information observed across all sensors as $X \in \mathbb{R}^{|S| \times P}$, where P is the total number of predictive features used. Let $X^{(t)} \in \mathbb{R}^{|S| \times P}$ denote the traffic signal observed at time t, and $Y^{(t')} \in \mathbb{R}^{|S| \times Q}$ denote the traffic signal to be predicted at time t'. Note that the number of target features Q may be different to P. We aim to learn a function $f(\cdot)$ which maps from T' historical traffic signals to T future traffic signals:

$$[X^{(t-T'+1)}, \ldots, X^{(t)}] \xrightarrow{f(\cdot)} [Y^{(t+1)}, \ldots, Y^{(t+T)}] \tag{1}$$

3.2. Real-Time Forecasting

The real-time variant of the traffic forecasting problem adds the constraint that all processing takes place within a specified duration following the end of each time bin. In Foresight, anonymized streaming data is collected at road cameras and ingested into the platform via an API endpoint. Further details of this procedure are illustrated in Section 6.1.

The real-time forecasting inference routine begins at the end of each time bin, which are each B minutes long. First, the raw vehicle-level data (held in cloud storage) is aggregated for the most recent time bin (i.e., the B minutes from $X^{(t-1)}$ to $X^{(t)}$). We denote the time taken for this aggregation as T_{Agg}. Next, the aggregated data is pre-processed so that it is in the correct format for model inference. This includes fetching and processing the aggregated traffic count information for the last T' time bins, as well as retrieving any additional model-specific data used for inference (e.g., roadwork time series, adjacency matrix). The time taken for this phase is referred to as $T_{PreProc}$. Once the required data have been produced, the inference API endpoint is invoked to perform the forecast. The time taken for inference processing to occur, as in Equation 1, is denoted by T_{Inf}.

We require the following expression to be satisfied for a system to be capable of real-time forecasting:

$$T_{Total} = T_{Agg} + T_{PreProc} + T_{Inf} \leq B \tag{2}$$

A value of $T_{Total} \leq B$ ensures that the shortest forecasting horizon still pertains to information that is yet to be aggregated in the system, and is therefore relevant to network managers.

4. Dynamic Urban Events

Foresight is able to leverage DUE data dynamically to improve the accuracy of its forecasts. We use roadworks data as an illustrative example, but other such information (e.g., social event data) could readily be applied in a similar fashion. In the context of traffic forecasting, planned and unplanned roadworks frequently influence the volume and nature of traffic propagation through the road network [22, 23], and so incorporating information about them into predictive models is important for accurate predictions. Foresight automatically ingests DUE data and processes it into a format which forecasting models can easily exploit.

Roadworks data is ingested into Foresight via the Street Manager API[1], which is invoked daily to receive a feed of planned roadwork events. We denote the set of all roadworks listed by a given daily API call as R. For each roadwork $r \in R$, we obtain its latitude/longitude, as well as its start and end dates T_s and T_e. In order to associate the live roadworks on a given day T with the road sensor network S, we first select only those roadworks where $T_s \leq T \leq T_e$. Next, we calculate the road network distance (using an indicative driving speed over a shortest path calculation on the road network) between each $r \in R$ and each $s \in S$. These distances populate an $|R| * |S|$ matrix W, with each entry (i, j) denoting the road network distance from live roadwork i to traffic sensor j in the network.

To incorporate this roadwork-to-camera influence information into the forecasting models, we convert W into a time series format at the same temporal granularity as the observed traffic data. This has been shown to be an effective method for adding roadwork data to forecasting models [5]. We define this as a new feature set with the same dimensions as X, formally $\hat{X} \in \mathbb{R}^{|S|}$. Each entry $\hat{x_i} \in \hat{X}^{(t)}$ has a value between 0 and 1 which denotes the strength of the influence of the nearest active roadwork to sensor i at time t.

We consider two approaches to approximate this influence. The first is a binary thresholding approach, where entries are activated if there is a roadwork within threshold distance d metres of the sensor. The second method involves first calculating the distance from each sensor to its nearest live roadwork, before normalizing these distances into $[0, 1]$. We perform this normalization using a thresholded Gaussian kernel, with threshold k.

[1] https://www.gov.uk/guidance/find-and-use-roadworks-data

Combining X and $\hat{X}$, a new matrix $\tilde{X} = \begin{bmatrix} X \\ \hat{X} \end{bmatrix}$ is constructed, which is the new feature vector passed to the forecasting models. We evaluate these approaches within the context of a GNN model in Section 7.

5. Flow Aggregated Adjacency Matrix

Graph Neural Networks (GNNs) are popularly used in state-of-the-art forecasting models [27, 28, 29, 30, 31, 3, 34, 7]. These methods typically represent the traffic sensor network as a graph structure, whose adjacency matrix aims to capture spatial relationships between the sensors. The principle of GNN message passing and node aggregation approaches in the context of traffic forecasting, such as diffusion convolution [7], is to simulate traffic propagation in the network. This method of extracting features is typically embedded into a wider learning structure so that temporal features can be learnt along with spatial features in an integrated fashion.

The graph structure which models the traffic sensor network is described by an $|S| \times |S|$ (weighted) adjacency matrix. The value at position (i, j) approximates the strength of the relationship between sensor s_i and sensor s_j. A popular method to assign weights in the adjacency matrix is to calculate pairwise sensor distances measured in the road network [31, 3, 7].

The aim of this approach is to more realistically reflect the actual flow of traffic in the network, compared to blunt sensor separation measures such as Euclidean distance. However, these measures alone are insufficient, as sensor separation per-se does not necessarily indicate traffic flow levels. Even though two sensors are spatially co-located, traffic might rarely pass between them consecutively, or may flow in one direction significantly more than the other; these properties cannot be easily captured by this approach.

We therefore develop a method for computing the adjacency matrix weights which uses vehicle-level flow data to more accurately determine the relationships between sensors. By leveraging the properties of granular ANPR (Automatic Number Plate Recognition) data, our method can anonymously capture (in order) the sequence of sensors which the cars pass as they traverse the road network. By aggregating this information at the network level, we are able to determine actual flows within the network. The new adjacency matrix is designed to retain the same dimensions used in most GNN methods for spatio-temporal forecasting, so it can be directly applicable within these methods.

The Flow Aggregated Adjacency Matrix (FAAM), denoted as $F \in \mathbb{R}^{|S| \times |S|}$, is constructed by aggregating observed flow between cameras within a given time frame.

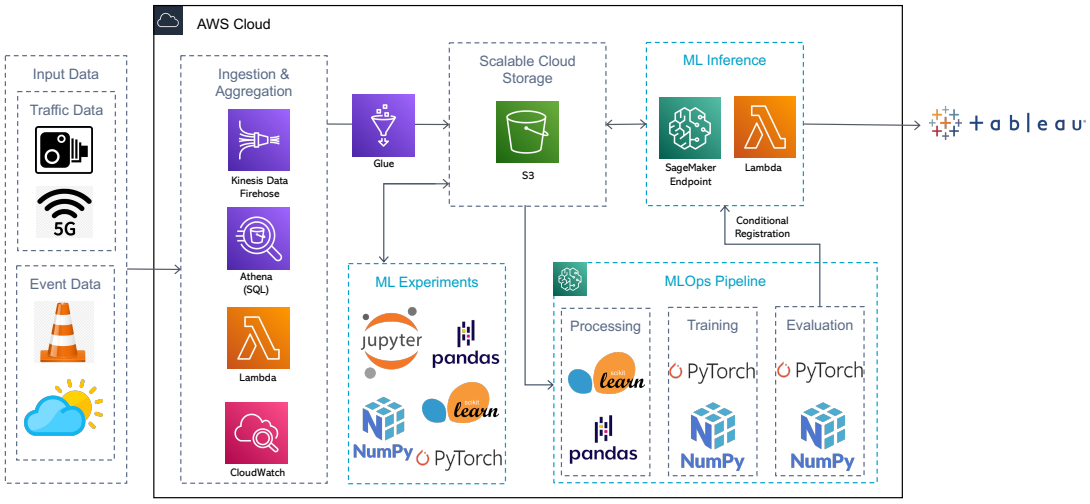

Figure 1: High-level architecture of the Foresight cloud-based forecasting system

1 unit of flow is recorded between cameras i and j when a car is observed at camera $s_i \in S$ at time t, and is then next observed $s_j \in S$ no later than $t + \tau$, where τ is a parameter given in seconds which denotes the acceptable transition period. To construct F, each entry $F_{i,j}$ is incremented by 1 for each observed unit of flow. $F_{i,j}$ is then averaged over all the time periods during which flow was observed, before being normalized into $[0, 1]$. Each entry $F_{i,j}$ thus approximates the likelihood of a vehicle transitioning *directly* from s_i to s_j within transition period τ. This can be periodically updated to reflect changes in the network over time, such as seasonality. We note that a more granular time scale would be possible in this formulation, e.g., to capture shifting traffic patterns throughout the day, but we leave this to be explored in future work.

6. System Architecture

In this section we present an overview of the Foresight system, as illustrated in Figure 1. The overall goal of Foresight is to provide continuous forecasts for transport managers by leveraging streaming traffic data as well as dynamic urban event and flow information. We will first describe how streaming traffic data is ingested and aggregated, before presenting the MLOps pipeline and forecasting inference procedure. Details of how DUE data and flow information are processed are given in Sections 4 and 5 respectively.

6.1. Streaming Data Ingestion, Aggregation and Storage

Foresight's primary data source is anonymized ANPR vehicle capture information in the West Midlands road network managed by TfWM. This data flows into Foresight using a POST request to an API endpoint, before being forwarded to a streaming ETL service (Kinesis Data Firehose[2]). Individual vehicle captures (including a timestamp, salted hash of vehicle registration, camera/lane of observation and vehicle type) are buffered using this service, and are periodically flushed to object storage (once the buffer fills, or a short time period elapses). The buffered file is also converted to a columnar format (Apache Parquet[3]) for improved query performance.

We next use a serverless data integration offering (AWS Glue[4]), to crawl the object storage buckets containing these intermediate files periodically. This enables the use of serverless SQL queries (via AWS Athena[5]) over the columnar Parquet data. These queries generate ag-

[2]https://aws.amazon.com/kinesis/data-firehose/
[3]https://parquet.apache.org
[4]https://aws.amazon.com/glue/
[5]https://aws.amazon.com/athena/

gregated traffic count data, illustrating the total number of vehicles of each type (e.g., petrol car, HGV) that have passed each roadside camera within the current time bin, i.e., the last B minutes. We use scheduling functionality in a cloud monitoring service (AWS CloudWatch[6]) to trigger the SQL processing (via lightweight serverless functions) for the current time bin. This procedure writes a single file to object storage (AWS S3[7]) per the current time bin, which can later be used as an input to ML workflows.

6.2. MLOps Pipeline and Training

We leverage an AWS SageMaker[8] MLOps pipeline to create and deploy forecasting models. Data scientists can run experiments (e.g., in SageMaker Studio Notebooks) over data held in object storage, using standard libraries such as NumPy, PyTorch, TensorFlow, etc. Once a model has successfully been designed, its source code can be pushed to one of two Git repositories (test, production) hosted in AWS CodeCommit[9].

Once a code update is performed in either repository, the MLOps pipeline provisions a compute instance (whose size is specified by the data scientist) to perform the necessary pre-processing and training of the model. The trained model is then deployed to a SageMaker endpoint (which resides on a provisioned compute instance), where it can first be tested in a 'staging' environment before being made available for live inference (if the production repository was updated). The MLOps pipeline can be configured to re-train the model periodically, e.g., once per week, to continually incorporate the latest traffic data.

6.3. Real-Time Forecasting Inference

As described in Section 6.1, the ANPR traffic data is aggregated in periodic B-minute time bins. We configure a serverless function (AWS Lambda[10]) to be triggered each time a new per-bin traffic file arrives in the specified S3 location. Once the serverless function is invoked, it first retrieves the latest T' historical traffic files (see Section 3) and executes the required pre-processing logic. Note that while this serverless function is lightweight, the small data volume makes this processing efficient. Next, any auxiliary data (e.g., adjacency matrices, roadworks time series) required for inference is also fetched. The resultant payload is sent to the live inference endpoint described above, where traffic predictions for the next T time bins over all sensors S are made. These predictions are then written to object storage, where they can be retrieved for downstream visualization.

[6]https://aws.amazon.com/cloudwatch/
[7]https://aws.amazon.com/s3/
[8]https://aws.amazon.com/sagemaker/
[9]https://aws.amazon.com/codecommit/
[10]https://aws.amazon.com/lambda/

7. Experimental Analysis

In this section we present the results of our experiments to test the effectiveness of popular traffic forecasting methods in a new setting. We then evaluate the impact of incorporating DUE data as an additional dimension to the input feature vector. We also consider the performance impact of using the FAAM, in place of a distance-based adjacency matrix, in a GNN forecasting model. Finally, we explore the error profiles of our models, and their efficiency within Foresight.

7.0.1. Road Camera Dataset

The anonymized and aggregated data used for the experiments is from a set of ANPR cameras in the West Midlands region of the UK, covering several large conurbations including Birmingham and Coventry. The precise locations of cameras remain private. The set of cameras are spread over a variety of different road types, including many roads from inner city locations and smaller connecting roads. This is different to many prior datasets, such as METR-LA [40], where road sensors are typically located on freeways where one can expect a high volume of free-flowing traffic. The quality of this data is high; the rate of missingness is only 2.3%, compared with 8.1% for METR-LA. We use linear interpolation to impute these missing values.

7.0.2. Experimental Setup

The vehicle count data used in the experiments was collected between August 5th and December 5th 2021 (inclusive), and was aggregated at 15 minute intervals. DUE data was collected for the same period. The flow was measured between August and November 2021 in order to compute the FAAM. The data was split into training, validation and test sets in a 70/10/20 ratio. We evaluate performance using mean absolute error (MAE) and mean absolute percentage error (MAPE). We also calculate the error distribution's coefficient of variation, which we refer to as the error coefficient of variation (ECV). We refer to the set of absolute errors across all test samples as $\mathcal{E}$, and hence $ECV = \frac{\sigma(\mathcal{E})}{\mu(\mathcal{E})}$. The ECV allows us to compare the dispersion of the error terms across different distributions (i.e., the sets of errors made by different models), as it normalizes by the mean error. A high ECV indicates that predictions are inconsistent.

We evaluate the results firstly over all time periods in the test data, which we refer to as 'Any Time' (**AT**) experiments. We also perform evaluation focusing only on 'Peak Times' (**PT**). We identify peak times as those that

Table 1
Table of results for all time periods (AT), as well as peak time (PT) periods only. Evaluation metrics include Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE), and Error distribution Coefficient of Variation (ECV).

	MAE				MAPE				ECV			
AT Results	15m	30m	45m	60m	15m	30m	45m	60m	15m	30m	45m	60m
Historical Average	28.22934	28.22934	28.22934	28.22934	1.247711	1.247711	1.247711	1.247711	1.353424	1.353424	1.353424	1.353424
ARIMA	17.21155	20.89709	24.38902	27.32631	0.29155	0.38807	0.484831	0.58018	1.176414	1.182061	1.165117	1.147103
FFNN	22.45165	26.00545	28.01167	30.38627	0.39497	0.475704	0.531725	0.596754	**1.074771**	**1.094029**	**1.096674**	**1.086678**
LSTM	16.47735	18.32613	21.04601	21.67277	**0.263589**	0.332847	0.402864	0.481703	1.259913	1.262905	1.247801	1.2313
DCRNN	13.04059	14.54253	15.71616	16.74402	0.321632	0.418699	0.488751	0.536253	1.163956	1.2184	1.260844	1.285605
DCRNN-RW-T	12.93978	14.27762	15.50452	16.62283	0.27605	0.342481	0.383549	0.426135	1.190628	1.252297	1.289899	1.305235
DCRNN-RW-G	12.88376	14.38243	15.75823	17.11553	0.272726	**0.31275**	**0.342886**	**0.380381**	1.181661	1.261551	1.298554	1.307771
DCRNN-F	**12.77166**	**14.13801**	15.42213	**16.21135**	0.27862	0.326762	0.368899	0.40535	1.197509	1.256911	1.28901	1.311176
DCRNN-RW-F	13.03387	14.29395	**15.3533**	16.29535	0.316565	0.353716	0.392546	0.425808	1.173395	1.250771	1.288463	1.315586

PT Results	15m	30m	45m	60m	15m	30m	45m	60m	15m	30m	45m	60m
Historical Average	46.56269	46.56269	46.56269	46.56269	2.339376	2.339376	2.339376	2.339376	1.085763	1.085763	1.085763	1.085763
ARIMA	24.00209	31.96453	40.5593	49.46829	**0.292056**	0.377998	0.457831	0.551944	1.100716	1.09567	1.098422	1.05831
FFNN	30.15297	35.27718	37.86231	41.33159	0.386185	0.460063	0.502773	0.575422	**1.063224**	**1.056382**	**1.07224**	**1.024506**
LSTM	19.82316	21.69737	23.50758	23.81264	0.301443	**0.373158**	0.414629	0.532906	1.193412	1.206935	1.210903	1.185619
DCRNN	16.1567	18.43098	20.17074	21.84788	0.415111	0.50987	0.585289	0.6306	1.066371	1.08657	1.103456	1.118936
DCRNN-RW-T	15.73651	18.33858	20.46045	22.31611	0.343233	0.450596	0.528813	0.609767	1.08885	1.116707	1.118316	1.124693
DCRNN-RW-G	16.00873	18.6837	21.26546	24.33342	0.318907	0.377966	**0.409407**	**0.457951**	1.095381	1.123309	1.131258	1.135483
DCRNN-F	**15.28226**	**16.98344**	**18.73013**	**19.91762**	0.349778	0.429841	0.485283	0.527475	1.102259	1.127692	1.145436	1.169329
DCRNN-RW-F	15.68353	17.47026	19.70059	21.80648	0.385255	0.43705	0.492557	0.553111	1.106778	1.145753	1.141361	1.142052

have historically shown high average traffic counts, but also high levels of variability. High average traffic counts indicate heavy load on the network, which we assume are periods of interest for transport managers. High levels of variability are a sign of challenging forecasting conditions, and may denote periods of unusual traffic conditions on the network. We identify these periods of interest by first dividing the dataset into weekends and weekdays, and then further splitting each of these into hourly subsets. The mean and coefficient of variation of each subset is then calculated. Any of these subsets with both mean and coefficient of variation in the upper two quartiles is classified as peak time. The only time periods which satisfy this are 7am-8am, and 8am-9am on weekdays, hence we select these as our peak times. This selection also conforms closely to the domain knowledge of our partners at TfWM.

7.1. Forecasting Models

The following forecasting models have been evaluated on the road camera dataset.

- **Historical Average (HA)**: We produce a historical average matrix based on the training set. The average reading over the training set is calculated at each sensor in S for each of the 672 (4x24x7) weekly time steps. To perform inference, we give the historical average value of the target time period as our prediction (the notion of T' historical traffic signals is not applicable to this method).
- **ARIMA**: We iterate over all sensors and all test examples. In each iteration, we train an ARIMA model[11], using the previous $T' = 100$ values as the training input.
- **Feed Forward Neural Network (FFNN)**: We implement an FFNN, where the input consists of the previous T' readings across all sensors $s \in S$. The model produces predictions for the next T forecasting horizons. The network is constructed with two hidden linear layers, with ReLU activation functions. Model parameters are learned using backpropagation, with an L1 loss function.
- **Long Short Term Memory (LSTM)**: This is implemented similarly to FFNN, except using LSTM layers in place of linear layers. Within the LSTM layers, input data is treated as a sequence and temporal patterns are learnt using an additional hidden layer to capture the cell state, which passes information along the sequence.
- **Diffusion Convolutional Recurrent Neural Network (DCRNN)**: We select DCRNN [7] as an illustrative example of an effective GNN method. This method has been previously identified as one of the best-performing (GNN) approaches for the traffic forecasting on benchmark datasets [8]. The model utilizes a distance-based adjacency matrix to model the spatial relationships between road sensors, and employs diffusion convolution and bidirectional random walks to simulate traffic propagation in the network. We utilize the PyTorch implementation of DCRNN [7].
- **DCRNN-RW-T / DCRNN-RW-G**: DCRNN with DUE adaption to include roadwork data. **DCRNN-RW-T** associates live roadworks to all sensors within a 1000m distance threshold. **DCRNN-RW-G** uses thresholded Gaussian kernel normalization (threshold $k = 0.1$).
- **DCRNN-F**: DCRNN with the FAAM representing

[11] $p = 1, d = 0, q = 1$

the underlying graph structure. An acceptable transition period τ between sensors is given as 3600 seconds and thresholded Gaussian kernel normalization ($k = 0.1$) is applied on the matrix.
- **DCRNN-RW-F**: DCRNN with roadworks (using Gaussian kernel method) and FAAM.

All models are implemented in AWS SageMaker Studio using Python 3.6, on a ml.g4dn.xlarge instance. We use PyTorch 1.8 to implement **FFNN**, **LSTM**, and **DCRNN** (including all variants). Unless stated otherwise, $T' = T = 4$, and $B = 15$ minutes. In practice, we make predictions over horizons of 15, 30, 45 and 60 minutes (henceforth referred to as 15m, 30m, 45m, 60m). The more distant forecasting horizons (i.e., 45m, 60m) offer transport managers more time to implement pre-emptive interventions on the road network. Hence, performance gains here are particularly valuable.

7.2. Experimental Results

We describe the key findings from our experimental results, which are presented in Table 1. First, we compare the performance of several existing forecasting approaches in our new data setting. We then consider the impact of incorporating roadworks as an exogenous input feature, as well as using flow to determine edge weights in the adjacency matrix. Next, we discuss our findings pertaining to prediction reliability using ECV. Finally, we analyze the efficiency of Foresight.

7.2.1. Analysis of Existing Approaches

We first consider the performance of several existing spatio-temporal forecasting approaches in our new data setting. During the AT experiments it can be observed that DCRNN makes more accurate predictions across all four time horizons (MAE improvements - 15m: 20.9%, 30m: 20.6%, 45m: 25.3%, 60m: 22.7%) compared to the next closest model (LSTM), with the largest improvements seen at the longest forecasting horizons. Similarly, during the PT experiments DCRNN remains the most accurate model. However, it is interesting to note that the improvements compared to LSTM are now much smaller (MAE improvements - 15m: 18.5%, 30m: 15.1%, 45m: 14.2%, 60m: 8.3%), and the trend at longer horizons is reversed where we see the smallest MAE improvements. ARIMA tends to be a competitive model for shorter horizons, during both PT and AT experiments, however the performance deteriorates quickly at longer forecasting horizons, which indicates that this model requires fresh data to support accurate predictions. HA and FFNN make the least accurate predictions across all forecasting horizons.

Different trends emerge when MAPE performance is considered. DCRNN now exhibits poorer performance than LSTM across all forecasting horizons during the AT experiments (MAPE degradation - 15m: 22%, 30m: 25.8%, 45m: 21.3%, 60m: 11.3%); these discrepancies are further exacerbated in PT experiments (MAPE degradation - 15m: 37.7%, 30m: 36.6%, 45m: 41.2%, 60m: 18.3%). This is an interesting result as it suggests that while LSTM makes poorer predictions on average (i.e., MAE), it also makes fewer mistakes of a significant margin, leading to a lower MAPE (this metric is highly sensitive to outliers in the error term). It may therefore be inferred that LSTM is better than DCRNN at predicting unusual traffic patterns, especially at peak times. In terms of MAPE, ARIMA was shown to be a highly competitive model across all forecasting horizons, outperforming DCRNN in all cases, with more pronounced gains in PT experiments. As ARIMA is retrained on the most recent data when evaluating each test sample (see Section 7.1), it will naturally be more responsive to unusual traffic patterns than models trained using a conventional train/test split. LSTM still largely outperforms ARIMA in regards to MAPE. HA performs particularly poorly on this metric, due to its inability to dynamically respond to current network conditions.

7.2.2. DUE Analysis

We also evaluate the impact of adding dynamic urban events to GNN models. Mixed results are achieved when MAE is considered. DCRNN-RW-G, which associates roadworks using a thresholded Gaussian kernel, generally yields higher MAE than DCRNN across both AT and PT experiments. These discrepancies in MAE are particularly pronounced for long forecasting horizons during peak times (MAE degradation - 45m: 5.4%, 60m: 11.4%). On the other hand, DCRNN-RW-T (binary thresholding) achieves lower MAE compared to DCRNN over all AT experiments. However, it still yields inferior performance at more distant forecasting horizons at peak times (MAE degradation - 45m: 1.4%, 60m: 2.1%).

The results for MAPE present a contrasting picture, where DCRNN-RW-G outperforms DCRNN-RW-T across all experiments. During AT experiments (especially at longer horizons), DCRNN-RW-G achieves significant improvements compared to DCRNN (MAE improvements - 45m: 29.8%, 60m: 29.1%). We observe a similar pattern during peak times. These findings indicate that using a thresholded Gaussian kernel during the construction of a FAAM yields a reduction in large outlier errors (likely resulting in improved performance under unusual road network conditions).

7.2.3. FAAM Analysis

Our experimental results indicate that using vehicle-level flow data to model inter-sensor relationships is an effec-

tive strategy. For AT experiments, DCRNN-F achieves lower MAE than DCRNN across all time horizons, and is particularly effective at long forecasting horizons (MAE improvement - 60m: 3.2%). Further, we observe even larger MAE gains for DCRNN-F compared to DCRNN at peak times (MAE improvement - 60m: 8.8%). These findings support the inclusion of vehicle-level flow data into GNN models for improved predictive performance.

We note that leveraging the FAAM in place of a distance-based adjacency matrix (i.e., DCRNN) yields MAPE improvements in all cases. However, the most significant MAPE gains are still experienced by DCRNN-RW-G, indicating that incorporating roadworks is a more effective strategy for minimizing outlier errors. It should be noted that DCRNN-F mitigates much of the degradation in MAPE performance at peak times that DCRNN suffers in comparison to LSTM, while also offering leading MAE results.

7.2.4. Error Coefficient of Variation

As illustrated in Table 1, DL models, particularly those which have been enhanced by DUE data or the FAAM, experience the highest ECV (especially at longer forecasting horizons). As shown above, it is at these more distant horizons that the biggest performance improvements (MAE/MAPE) are observed for our augmented models. This suggests that while these solutions produce the best forecasts on average, their errors are the least consistent. This finding is noteworthy, and we would recommend further investigation to better understand its implications.

7.2.5. Efficiency Analysis

As discussed in Section 3.2, the real-time forecasting task requires that $T_{Total} = T_{Agg} + T_{PreProc} + T_{Inf} \leq B$. In the current version of Foresight, we allow for $T_{Agg} \leq 40$ seconds. For all of the implemented forecasting models, $T_{PreProc} \leq 6$ seconds. Each model except ARIMA achieves $T_{Inf} \leq 2$ seconds. As discussed above, at inference time we train an ARIMA model over the previous $T' = 100$ values for each $s \in S$. For ARIMA, $T_{Inf} \leq 16$ seconds. Hence, all of the presented models achieve $T_{Total} \approx 1$ minute, satisfying Equation 2 with significant headroom for $B = 15$ minutes. Further, these results conform to alternative notions of real-time forecasting [41], where predictions were produced in a single-digit order of minutes.

8. Conclusion

In this work, we study the real-time spatio-temporal forecasting problem, and develop a cloud-based system in collaboration with Transport for the West Midlands. We utilize this system to develop methods for the incorporation of Dynamic Urban Event (DUE) and vehicle-level flow information into predictive models. We consider several approaches for associating roadworks to the roadside sensors which they impact, and present a FAAM based on granular and anonymized vehicle-level flow data. We implement and evaluate a range of time series forecasting models in a new urban data setting, enabled by ANPR infrastructure in the West Midlands region. We observe performance improvements of up to 29.1% for GNN models when DUE and flow data are leveraged. Our experimental results provide further insights on the forecasting accuracy during peak times when accurate results are more critical, compared to the traditional approach of targeting the average accuracy. Further work may apply our approach for handling DUEs to more varied data sources beyond planned roadworks, such as weather or cultural event information. Our data-driven method for identifying peak times could be compared to a relevant ground truth. Foresight's response time could be improved, particularly via acceleration of T_{Agg}. Finally, an additional temporal dimension could be added to our flow-based adjacency matrix.

Acknowledgments

We thank Transport for the West Midlands and our collaborators there, particularly Tim Katheru, Andrew Burns and Stuart Lester in the Data Insights team. This research is also supported in part by the Feuer International Scholarship in Artificial Intelligence, WM5G, and the UK Engineering and Physical Sciences Research Council under Grant No. EP/L016400/1.

References

[1] K. L. Li, C. J. Zhai, J. M. Xu, Short-term traffic flow prediction using a methodology based on arima and rbf-ann, Proceedings - 2017 Chinese Automation Congress, CAC 2017 2017-January (2017) 2804–2807. doi:10.1109/CAC.2017.8243253.

[2] R. Fu, Z. Zhang, L. Li, Using lstm and gru neural network methods for traffic flow prediction, Proceedings - 2016 31st Youth Academic Annual Conference of Chinese Association of Automation, YAC 2016 (2017) 324–328. doi:10.1109/YAC.2016.7804912.

[3] Z. Wu, S. Pan, G. Long, J. Jiang, X. Chang, C. Zhang, Connecting the dots: Multivariate time series forecasting with graph neural networks, Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining 20 (2020) 753–763. URL: https://arxiv.org/abs/2005.11650v1. doi:10.48550/arxiv.2005.11650.

[4] L. Franceschi, M. Niepert, M. Pontil, X. He, Learning discrete structures for graph neural networks, 36th International Conference on Machine Learning, ICML 2019 2019-June (2019) 3481–3493. URL: https://arxiv.org/abs/1903.11960v4. doi:10.48550/arxiv.1903.11960.

[5] Y. Lu, P. Kamranfar, D. Lattanzi, A. Shehu, Traffic flow forecasting with maintenance downtime via multi-channel attention-based spatio-temporal graph convolutional networks (2021). URL: https://arxiv.org/abs/2110.01535v1. doi:10.48550/arxiv.2110.01535.

[6] L. Zhao, Y. Song, C. Zhang, Y. Liu, P. Wang, T. Lin, M. Deng, H. Li, T-gcn: A temporal graph convolutionalnetwork for traffic prediction, IEEE Transactions on Intelligent Transportation Systems 21 (2018) 3848–3858. URL: http://arxiv.org/abs/1811.05320http://dx.doi.org/10.1109/TITS.2019.2935152. doi:10.1109/TITS.2019.2935152.

[7] Y. Li, R. Yu, C. Shahabi, Y. Liu, Diffusion convolutional recurrent neural network: Data-driven traffic forecasting, International Conference on Learning Representations, ICLR, 2017. URL: https://arxiv.org/abs/1707.01926v3. doi:10.48550/arxiv.1707.01926.

[8] K. H. N. Bui, J. Cho, H. Yi, Spatial-temporal graph neural network for traffic forecasting: An overview and open research issues, Applied Intelligence 52 (2021) 2763–2774. URL: https://link.springer.com/article/10.1007/s10489-021-02587-w. doi:10.1007/S10489-021-02587-W/TABLES/5.

[9] W. Alajali, W. Zhou, S. Wen, Y. Wang, Intersection traffic prediction using decision tree models, Symmetry 2018, Vol. 10, Page 386 10 (2018) 386. URL: https://www.mdpi.com/2073-8994/10/9/386/htmhttps://www.mdpi.com/2073-8994/10/9/386. doi:10.3390/SYM10090386.

[10] P. Duan, G. Mao, W. Yue, S. Wang, A unified starima based model for short-term traffic flow prediction, IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC 2018-November (2018) 1652–1657. doi:10.1109/ITSC.2018.8569964.

[11] T. Alghamdi, K. Elgazzar, M. Bayoumi, T. Sharaf, S. Shah, Forecasting traffic congestion using arima modeling, 2019 15th International Wireless Communications and Mobile Computing Conference, IWCMC 2019 (2019) 1227–1232. doi:10.1109/IWCMC.2019.8766698.

[12] D. Xu, Y. Shi, A combined model of random forest and multilayer perceptron to forecast expressway traffic flow, Proceedings of 2017 IEEE 7th International Conference on Electronics Information and Emergency Communication, ICEIEC 2017 (2017) 448–451. doi:10.1109/ICEIEC.2017.8076602.

[13] Z. Zheng, L. Shi, L. Sun, J. Du, Short-term traffic flow prediction based on sparse regression and spatio-temporal data fusion, IEEE Access 8 (2020) 142111–142119. doi:10.1109/ACCESS.2020.3013010.

[14] O. Mohammed, J. Kianfar, A machine learning approach to short-term traffic flow prediction: A case study of interstate 64 in missouri, 2018 IEEE International Smart Cities Conference, ISC2 2018 (2019). doi:10.1109/ISC2.2018.8656924.

[15] A. Barredo-Arrieta, I. Lana, J. D. Ser, What lies beneath: A note on the explainability of black-box machine learning models for road traffic forecasting, 2019 IEEE Intelligent Transportation Systems Conference, ITSC 2019 (2019) 2232–2237. doi:10.1109/ITSC.2019.8916985.

[16] LiYaguang, ShahabiCyrus, A brief overview of machine learning methods for short-term traffic forecasting and future directions, SIGSPATIAL Special 10 (2018) 3–9. URL: https://dl.acm.org/doi/10.1145/3231541.3231544. doi:10.1145/3231541.3231544.

[17] X. Luo, D. Wang, D. Ma, S. Jin, Grouped travel time estimation in signalized arterials using point-to-point detectors, Transportation Research Part B: Methodological 130 (2019) 130–151. doi:10.1016/J.TRB.2019.10.007.

[18] D. Ma, X. Song, P. Li, Daily traffic flow forecasting through a contextual convolutional recurrent neural network modeling inter- and intra-day traffic patterns, IEEE Transactions on Intelligent Transportation Systems 22 (2021) 2627–2636. doi:10.1109/TITS.2020.2973279.

[19] Y. Tian, K. Zhang, J. Li, X. Lin, B. Yang, Lstm-based traffic flow prediction with missing data, Neurocomputing 318 (2018) 297–305. doi:10.1016/J.NEUCOM.2018.08.067.

[20] Q. Zhaowei, L. Haitao, L. Zhihui, Z. Tao, Short-term traffic flow forecasting method with m-b-lstm hybrid network, IEEE Transactions on Intelligent Transportation Systems 23 (2020) 225–235. doi:10.1109/TITS.2020.3009725.

[21] W. Jiang, J. Luo, Graph neural network for traffic forecasting: A survey, Expert Systems with Applications 207 (2021). URL: http://arxiv.org/abs/2101.11174http://dx.doi.org/10.1016/j.eswa.2022.117921. doi:10.1016/j.eswa.2022.117921.

[22] M. M. Bruwer, S. J. Andersen, W. Merrick, Measuring the impact of roadworks on traffic progression using floating car data, 2021.

[23] C. Schwietering, M. Feldges, Improving traffic flow at long-term roadworks, Transportation Research Procedia 15 (2016) 267–282. doi:10.1016/J.TRPRO.2016.06.023.

[24] J. Brugmann, M. Schreckenberg, W. Luther, Real-time traffic information system using microscopic traffic simulation, Proceedings - 8th EUROSIM Congress on Modelling and Simulation, EUROSIM 2013 (2015) 448–453. doi:10.1109/EUROSIM.2013.83.

[25] A. Essien, I. Petrounias, P. Sampaio, S. Sampaio, A deep-learning model for urban traffic flow prediction with traffic events mined from twitter, World Wide Web 24 (2021) 1345–1368. URL: https://link.springer.com/article/10.1007/s11280-020-00800-3. doi:10.1007/S11280-020-00800-3/TABLES/4.

[26] J. Haworth, T. Cheng, Non-parametric regression for space–time forecasting under missing data, Computers, Environment and Urban Systems 36 (2012) 538–550. doi:10.1016/J.COMPENVURBSYS.2012.08.005.

[27] L. Zhao, M. Chen, Y. Du, H. Yang, C. Wang, Spatial-temporal graph convolutional gated recurrent network for traffic forecasting (2022). URL: https://arxiv.org/abs/2210.02737v1. doi:10.48550/arxiv.2210.02737.

[28] S. Shleifer, C. McCreery, V. Chitters, Incrementally improving graph wavenet performance on traffic prediction (2019). URL: https://arxiv.org/abs/1912.07390v1. doi:10.48550/arxiv.1912.07390.

[29] W. Chen, L. Chen, Y. Xie, W. Cao, Y. Gao, X. Feng, Multi-range attentive bicomponent graph convolutional network for traffic forecasting, AAAI 2020 - 34th AAAI Conference on Artificial Intelligence (2019) 3529–3536. URL: https://arxiv.org/abs/1911.12093v1. doi:10.48550/arxiv.1911.12093.

[30] Z. Pan, Y. Liang, W. Wang, Y. Yu, Y. Zheng, J. Zhang, Urban traffic prediction from spatio-temporal data using deep meta learning, Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2019) 1720–1730. URL: https://dl.acm.org/doi/10.1145/3292500.3330884. doi:10.1145/3292500.3330884.

[31] X. Wang, Y. Ma, Y. Wang, W. Jin, X. Wang, J. Tang, C. Jia, J. Yu, Traffic flow prediction via spatial temporal graph neural network, The Web Conference 2020 - Proceedings of the World Wide Web Conference, WWW 2020 (2020) 1082–1092. URL: https://dl.acm.org/doi/10.1145/3366423.3380186. doi:10.1145/3366423.3380186.

[32] T. Wu, F. Chen, Y. Wan, Graph attention lstm network: A new model for traffic flow forecasting, Proceedings - 2018 5th International Conference on Information Science and Control Engineering, ICISCE 2018 (2019) 241–245. doi:10.1109/ICISCE.2018.00058.

[33] X. Zhou, Y. Shen, L. Huang, Revisiting flow information for traffic prediction, 2019. URL: https://arxiv.org/abs/1906.00560v1. doi:10.48550/arxiv.1906.00560.

[34] Z. Diao, X. Wang, D. Zhang, Y. Liu, K. Xie, S. He, Dynamic spatial-temporal graph convolutional neural networks for traffic forecasting, Proceedings of the AAAI Conference on Artificial Intelligence 33 (2019) 890–897. URL: https://ojs.aaai.org/index.php/AAAI/article/view/3877. doi:10.1609/AAAI.V33I01.3301890.

[35] G. Yeghikyan, F. L. Opolka, M. Nanni, B. Lepri, P. Lio', Learning mobility flows from urban features with spatial interaction models and neural networks (2020) 57–64. URL: https://arxiv.org/abs/2004.11924v1. doi:10.48550/arxiv.2004.11924.

[36] S. Y. Shah, D. Patel, L. Vu, X. H. Dang, B. Chen, P. Kirchner, H. Samulowitz, D. Wood, G. Bramble, W. M. Gifford, G. Ganapavarapu, R. Vaculin, P. Zerfos, Autoai-ts: Autoai for time series forecasting, Proceedings of the ACM SIGMOD International Conference on Management of Data 17 (2021) 2584–2596. URL: https://arxiv.org/abs/2102.12347v2. doi:10.48550/arxiv.2102.12347.

[37] L. Tran, M. Y. Mun, M. Lim, J. Yamato, N. Huh, C. Shahabi, Deeptrans: A deep learning system for public bus travel time estimation using traffic forecasting, PVLDB 13 (2020) 2957–2960. URL: https://doi.org/10.14778/3415478.3415518. doi:10.14778/3415478.3415518.

[38] J. Ma, J. Chan, G. Ristanoski, S. Rajasegarar, C. Leckie, Bus travel time prediction with real-time traffic information, Transportation Research Part C: Emerging Technologies 105 (2019) 536–549. doi:10.1016/J.TRC.2019.06.008.

[39] X. Chen, J. Wang, K. Xie, Trafficstream: A streaming traffic flow forecasting framework based on graph neural networks and continual learning, IJCAI International Joint Conference on Artificial Intelligence (2021) 3620–3626. URL: https://arxiv.org/abs/2106.06273v1. doi:10.48550/arxiv.2106.06273.

[40] H. V. Jagadish, J. Gehrke, A. Labrinidis, Y. Papakonstantinou, J. M. Patel, R. Ramakrishnan, C. Shahabi, Big data and its technical challenges, Communications of the ACM 57 (2014) 86–94. URL: https://dl.acm.org/doi/10.1145/2611567. doi:10.1145/2611567.

[41] M. Milkovits, E. Huang, C. Antoniou, M. Ben-Akiva, J. A. Lopes, Dynamit 2.0: The next generation real-time dynamic traffic assignment system, Proceedings - 2nd International Conference on Advances in System Simulation, SIMUL 2010 (2010) 45–51. doi:10.1109/SIMUL.2010.28.