Mesoscale Traffic Forecasting for Real-Time Bottleneck and Shockwave Prediction

Raphael Chekroun, Han Wang, Jonathan Lee, Marin Toromanoff, Sascha Hornauer, Fabien Moutarde, Maria Laura Delle Monache

Mesoscale Traffic Forecasting for Real-Time Bottleneck and Shockwave Prediction

Raphael Chekroun[a,b,c], Han Wang[a], Jonathan Lee[a], Marin Toromanoff[c], Sascha Hornauer[b], Fabien Moutarde[b], Maria Laura Delle Monache[a]

[a] *Department of Civil and Environmental Engineering, University of California, Berkeley, Berkeley, 94720, CA, USA*
[b] *Center for Robotics, Mines Paris, PSL University, 60 boulevard Saint-Michel, Paris, 75006, IDF, France*
[c] *Valeo Driving Assistant Research, 6 rue Daniel Costantini, Créteil, 94000, IDF, France*

Abstract

Accurate real-time traffic state forecasting plays a pivotal role in traffic control research. In particular, the CIRCLES consortium[1] project necessitates predictive techniques to mitigate the impact of data source delays. After the success of the MegaVanderTest experiment [1], this paper aims at overcoming the current system limitations and develop a more suited approach to improve the real-time traffic state estimation [2] for the next iterations of the experiment. In this paper, we introduce the SA-LSTM, a deep forecasting method integrating Self-Attention (SA) on the spatial dimension with Long Short-Term Memory (LSTM) yielding state-of-the-art results in real-time mesoscale traffic forecasting. We extend this approach to multi-step forecasting with the n-step SA-LSTM, which outperforms traditional multiforms-step forecasting methods in the trade-off between short-term and long-term predictions, all while operating in real-time.

1. Introduction & State of the Art

Traffic forecasting stands as a pivotal research challenge in contemporary industrial academia. With the impending advent of autonomous vehicular systems, the imperative of accurate traffic prediction is accentuated, primarily due to its potential ramifications on urban design, public safety, and the

[1] https://circles-consortium.github.io/

overarching efficacy of transportation infrastructures. Anticipating forthcoming traffic conditions enables stakeholders—ranging from policymakers to urban strategists—to allocate resources judiciously, institute infrastructural enhancements in a timely manner, and conceptualize efficacious traffic governance methodologies. Such a proactive stance not only ameliorates congestion but also mitigates accident risks, attenuates environmental ramifications, and culminates in both temporal and financial savings for commuters and the broader populace. Traffic information is relevant on several levels of granularity, on a scale from micro to macro, each presenting its own interest. Micro-scale traffic information captures detailed, vehicle-level data, such as individual speeds, positions, and behaviors, providing a high-resolution view of traffic conditions at specific locations. It is often used for fine-grained analyses, like understanding the dynamics of a single intersection, or collaborative planning to enable an energy-efficient driving [3, 4, 5]. On the other hand, macro-scale traffic information focuses on aggregated, high-level data that provides an overall picture of traffic flow across broader areas. This can include metrics like average speeds, traffic volumes, and congestion levels, and is generally employed for long-term planning and large-scale traffic management [6]. Mesoscale traffic information occupies the middle ground between micro-scale and macro-scale. Specifically in the studied use case, it focuses on how groups of vehicles interact with each others on segmented portions of a single highway and how it impact average speeds acrross these distincts sections. These three types of information offer valuable insights but differ in their level of detail and computational requirements.

Traffic forecasting has long been explored via rule-based methods. In particular, some research extended the Kalman Filter for traffic estimation via ensemble methods [7] or Kalman recursions in dynamic state-space [8]. Alternative modelizations, such as particle filters [9] or spatial copulas [10], have also been leveraged to this extent. However, these methods suffer from performance decays when unexpected events provoke nonpredictable changes or if the allocation to a traffic pattern is inaccurate.

The advent of deep learning has addressed several shortcomings of rule-based methods. By learning from data, these models can account for unpredictable yet regular behaviors. Laña et al. [11] employed Spiking Neural Networks to achieve long-term pattern forecasting, adapting these predictions to real-time situations. For short-term forecasting, the Graph Convolution Network (GCN) has emerged as a potent tool. Guo et al. [12] utilized a GCN for traffic forecasting, integrating it with a latent network to glean spatial-

temporal features. Mallick et al. [13] enhanced the capabilities of GCN by incorporating ensembling methods, leveraging Bayesian hyperparameter optimization and generative modeling. However, despite their efficiency, these deep models consist of computationally demanding operations, making them unsuitable for real-time forecasting.

Recurrent Neural Networks (RNN), and in particular Long Short Term Memory (LSTM) [14], are lighter deep-based methods for forecasting able to effectively to capture and model sequential data via a sophisticated memory mechanism. Key components of LSTM networks are represented in Figure 1. During training, the LSTM network learns to adjust the parameters of its gates and the cell state in a way that allows it to capture long-range dependencies and patterns in sequential data. This enables LSTMs to excel in time series prediction where understanding context and dependencies over time is crucial. However, LSTM remains limited to capturing both spatial and temporal patterns in series prediction.

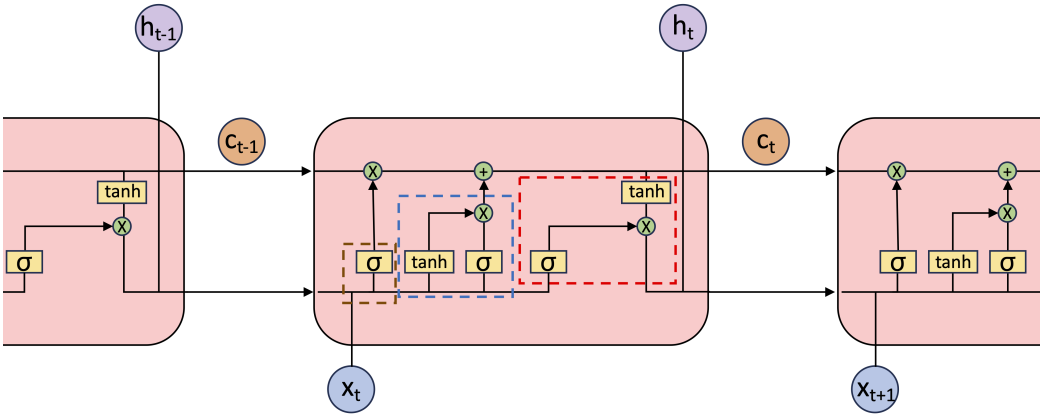


Figure 1 Representation of an LSTM cell. The Cell State (C_t, in orange) runs through the entire sequence. It stores and transmits information across time steps while selectively modifying or forgetting parts of it. The Hidden State (h_t, in purple) is the output of the LSTM cell at a specific time step. It carries information that is relevant to the current time step's prediction or output. It is also influenced by the cell state and the input at that time step. LSTMs employ three gate types (forget in brown, input in blue, and output in red) to regulate how information is managed within the cell state and the hidden state.

Incorporating an analysis of spatial dependencies has been explored through different methods. ConvLSTM replaces LSTM state-to-state and input-to-state transition with convolutions[15, 16] to bring LSTM the computational capability to analyze spatiotemporal series. Methods relying on attention are

able to learn how each data point interacts with each other at each timestep. In particular self-attention has been successfully leveraged with LSTM for diverse forecasting tasks [17, 18, 19], and Transformers [20, 21] recently showed promising results in data series [22]. Some methods leverages both convolution and self-attention to reach state-of-the-art results on some datasets [16]. Other methods combine LSTM and graph neural network [23, 24, 25] to forecast and quantify how roads and intersections impact one another through graph modelization. However, these methods are by design for macro-scale road systems and are unfit for meso modelizations.

In this study, our attention is directed toward mesoscale traffic forecasting, which occupies the middle ground between micro-scale and macro-scale analysis. Specifically, we examine how groups of vehicles interact on segmented portions of a single highway. We aim to determine the average speed of vehicles across these distinct sections. The data is limited to highway conditions and does not incorporate information from entry or exit ramps. The ultimate goal is to forecast in real-time the development of traffic bottlenecks and shockwaves as part of the *Congestion Impacts Reduction via Connected Autonomous Vehicles (CAV)-in-the-loop Lagrangian Energy Smoothing* (CIRCLES) project, which seeks to mitigate traffic congestion and energy waste by utilizing Connected Autonomous Vehicles (CAV) on highways.

This paper is organized as follows: In Section 2, we present the data source and how we modelize it as a data series problem. In Section 3 we present the methodology we developed for one-minute forecasting and multi-step forecasting. In Section 4, we present ablations studies and experimental results to justify our methods. Finally, Section 5 concludes this paper.

2. Data Collection and Forecasting Methodology

2.1. Data Acquisition

This research utilizes mesoscale data obtained from INRIX traffic services [26], which includes average speeds across multiple lanes on 21 segments of the I-24 interstate highway in Nashville, TN, as depicted in Figure 2.

This data spans mileposts MM66 to MM59, covering an 11.4 km road fraction divided into 21 segments, with a sampling rate of 3,600 data points per day. An example of typical morning traffic is shown in Figure 3.

While INRIX traffic data updates every minute, there can be a slight lag of up to three minutes in data generation. The focus of this study is to enhance the accuracy and timeliness of traffic forecasting, especially considering the

Figure 2 The Target Road Segment of CIRCLES: I-24 Westbound in Nashville, Tennessee, seen within the highlighted region.

brief delays in data updates. The objective is to develop a predictive model that effectively forecasts traffic patterns in three-minute intervals, leveraging the minute-by-minute data refreshment to anticipate and manage traffic conditions more efficiently.

2.2. Modelization as a data series problem

We modeled this data series problem in the following way. At every time-step t, we note v_t^i the average velocity over the lanes on the whole $i \in [0, 20]$ segment. Hence, the studied data series can be seen as follows:

$$\mathrm{V}_t = \begin{bmatrix} \mathrm{v}_t^0 \\ \mathrm{v}_t^1 \\ \vdots \\ \mathrm{v}_t^{20} \end{bmatrix},$$

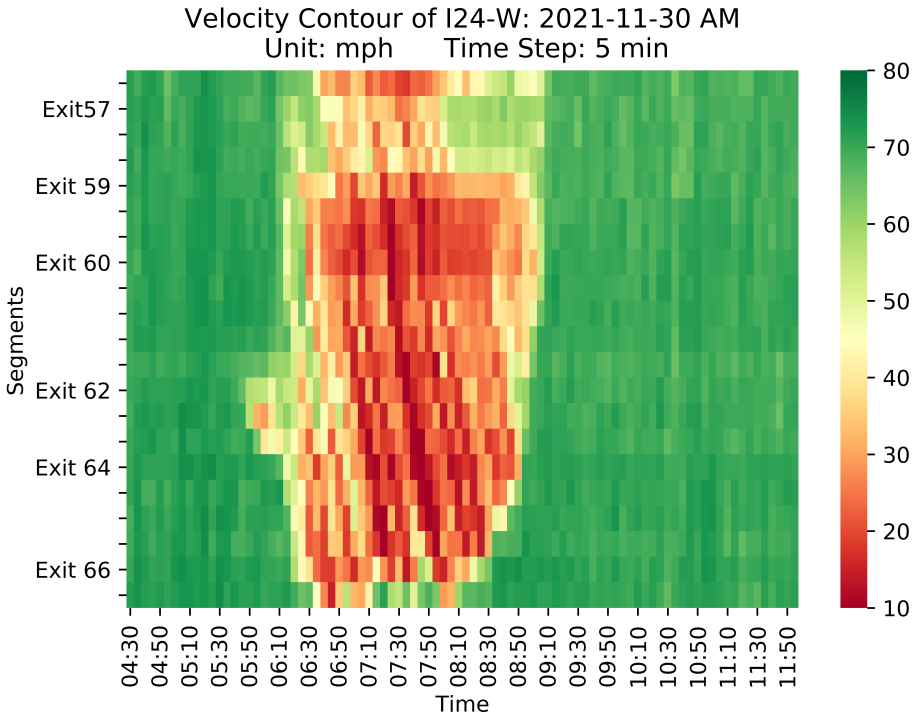


Figure 3 In the red contour of the figure, one observes the chronological progression of congestion on the specified segments. A notable persistent bottleneck is evident at Exit 59. This congestion initiates at approximately 6:00 a.m., likely attributable to the augmented commuting demand upstream, and it fully resolves by around 9:00 a.m.

We also define

$$\forall (t,k) \in \mathbb{N}^2, \mathrm{I}_t^k = V_t \oplus \cdots \oplus V_{t+k} = \begin{bmatrix} \mathrm{v}_t^0 & \cdots & \mathrm{v}_{t+k}^0 \\ \mathrm{v}_t^1 & \cdots & \mathrm{v}_{t+k}^1 \\ \vdots & & \vdots \\ \mathrm{v}_t^{20} & \cdots & \mathrm{v}_{t+k}^{20} \end{bmatrix}$$

The concatenation of k consecutive velocity vector starting at time t.

Therefore, our final model should be fed with I_{t-s}^s to output I_t^3, s being the chosen sequence length used as input.

2.3. Training and validation datasets

The training set is composed of 504,000 data points (every minute for 350 days). We also built two validation sets, also represented in Figure 4:

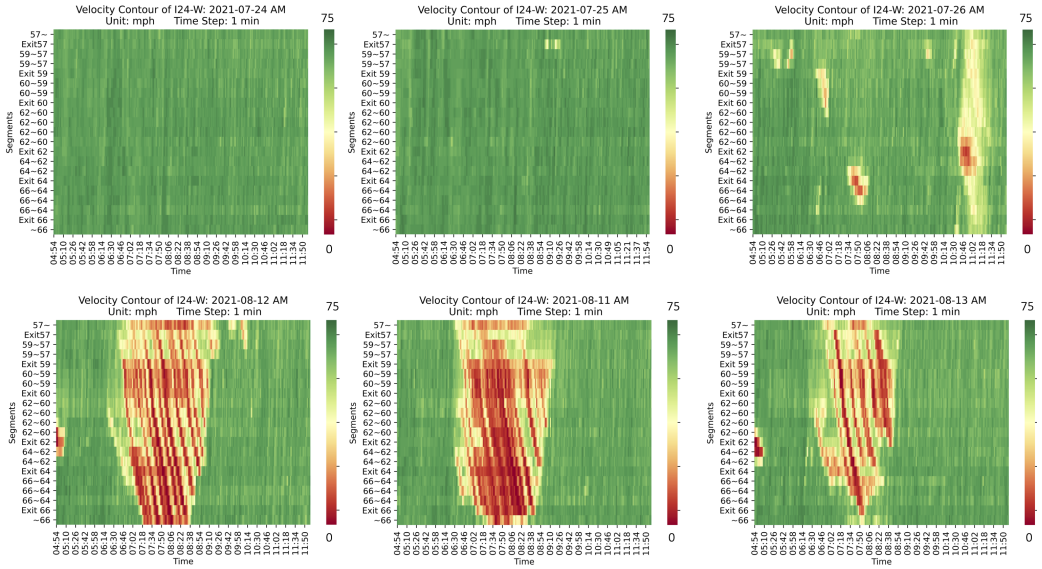


Figure 4 Illustrative representation of the validation datasets. **Top Row:** *Three representative snapshots from the Easy Validation set, showcasing common traffic patterns with periodic congestion and the prominence of temporal dependencies.* **Bottom Row:** *Three exemplar visuals from the Hard Validation set, highlighting moments of intense congestion, significant vehicle interactions, and the emphasis on spatial dependencies.*

- The Easy Validation set: made of 86,400 data points (every minute for 60 days), it mostly represents common traffic as most of it is smooth, with some discrete congestion and traffic shockwaves. There is usually at least one bigger traffic bottleneck every day between 6 am and 10 am. Traffic is mostly fluid in this dataset, interactions between vehicles are almost negligible, and metrics mostly represent a model capability to capture temporal dependencies.

- The Hard Validation set: the composition of four 440 minutes of highly congested traffic bottlenecks (1,760 total data points). Metrics are evaluated independently and averaged on those three sets to obtain the Hard metric. Traffic being highly congested, interactions between vehicles are consequent, and validations metrics on this dataset represent a model capability to capture spatial dependencies.

3. Real-Time Mesoscale Traffic Forecasting

Our primary emphasis is on single-step traffic prediction, which involves forecasting traffic conditions just one minute ahead. Subsequently, we expand our approach to solve the problem of multi-step traffic forecasting, which involves predicting traffic conditions several minutes into the future.

3.1. One-minute INRIX Prediction

While LSTM is a correct baseline for both accuracy and inference time, they do not qualify as an optimal solution for our data. Indeed, at a given time t, a traffic bottleneck at position k will impact both the short and long-term v_t^k, but also the neighboring segments $v_t^{k-\epsilon}$ and $v_t^{k+\epsilon}$. Hence, studied data presents spatio-temporal relationships, while standard LSTM mostly focuses on temporal relationships. To overcome this limitations, self-attention can be a powerful tool. Indeed, self-attention can intuitively capture the dynamic dependencies between different segments of the road network, recognizing how traffic conditions on one segment affect others. By attending to relevant spatial and temporal patterns, self-attention enables traffic forecasting models to adapt and predict congestion, flow changes, and bottlenecks. This intuitive capacity to capture inter-dependencies makes self-attention a valuable asset in improving the accuracy and reliability of mesoscale traffic forecasting, ultimately contributing to more effective traffic management strategies and reduced congestion. Mathematically, self-attention update the tokens via a weighted by X sum, with X computed via Equation 1.

$$X = \text{softmax}(\frac{Q.K^T}{\sqrt{d}})V \tag{1}$$

with the queries Q, keys K, and values V being three tensors created through linear projection from the input tensor, and d the feature size.

Therefore, we designed a Self-Attention LSTM (SA-LSTM) whose output gate is augmented with a self-attention layer on the spatial dimension. Our SA-LSTM is represented in Figure 5.

To train the network to focus on more fine-grained spatial information without further increasing the computational time of operations at inference, we leveraged the Laplacian Pyramid loss [27] mathematically defined in Equation 2.

$$\text{Lap}_n\left(x, x'\right) = \sum_{j=0}^{n} 2^{2j} \left| L^j(x) - L^j\left(x'\right) \right|_1 \tag{2}$$

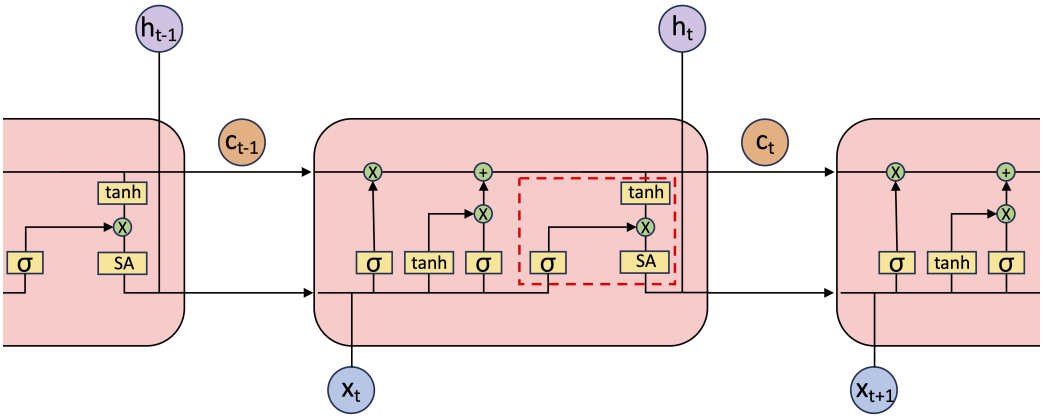


Figure 5 A single cell from an SA-LSTM network. The SA-LSTM is an LSTM in which the output gate, in red, is augmented with self-attention.

where $L^j(x)$ is the j-th level of the Laplacian pyramid representation of x [28]. It is a convolution-based loss able to weights the details at fine scales by capturing multi-scale information. It is used in addition to the MSE loss generally used for LSTMs.

3.2. n-step SA-LSTM

Section 3.1 studied one-minute forecasting. In practice however, we aim to forecast up to three-minutes, *i.e.* have access to $\mathrm{V}_{t+1}, \mathrm{V}_{t+2}, \mathrm{V}_{t+3}$. n-step forecasting is classically done via recursive inference over the data. Therefore, inferring I_t^3 is made in three successive inferences. First, the network is fed with I_{t-s}^s and outputs $\tilde{v}^t$. Then, the network is fed with $I_{t-s+1}^s \oplus \tilde{v}^t$ to infer $\tilde{v}^{t+1}$, and so on. However, such methods suffer from accumulation error, as inaccuracies in each inference will weigh on the next ones. Also, total inference time is at least n times the inference time of a single network inference. This method is therefore unfit for real-time inference. Another method is the all-at-once technique, in which a single LSTM is fed I_{t-s}^s and trained to output I_t^{t+n}. While significantly faster and offering better long-term forecasting, this method can lead to underperformance on short term forecasting compared to 1-step LSTM which is not desirable in our case of study.

We designed the n-step SA-LSTM, a highly supervised multi-layer SA-LSTM represented in Figure 6, to take the best of both world: a fast method resilient to accumulation error offering good short term and long term forecasting.

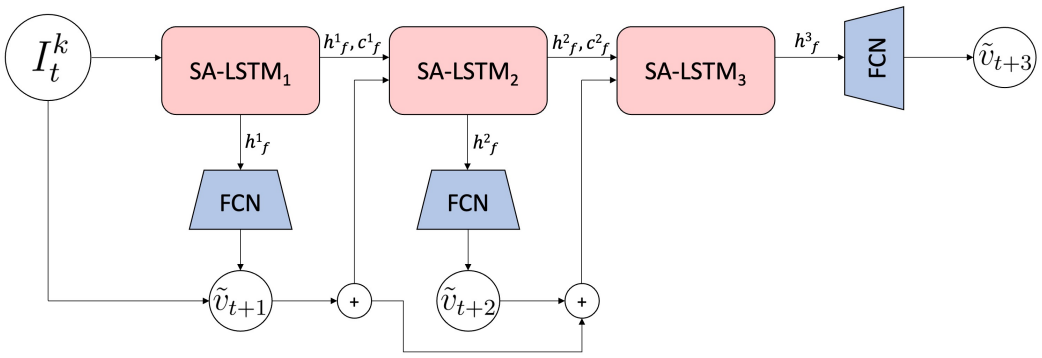


Figure 6 Each i-th layer of the 3-step SA-LSTM is trained to infer the forecasting at time $t+i$ through a shared weight fully connected network (FCN). We note h^i_f and c^i_f the outputs of the last cell of the i-th SA-LSTM.

An n-step SA-LSTM is a n layers SA-LSTM where:

- Each layer output is constrained via a loss to converge toward V_{t+i};
- Each i-th layer input is the concatenation of the network input $v_{t-k-1}...v_t$ concatenated with previous layer output $(\tilde{v}_{t+j})_{j\leq i}$. Layer i also takes h_{i-1}, c_{i-1} as input.

Therefore, each layer have the same input and output dimensions but contains a different number of cells - which is equal to the input sequence length. Indeed, if input sequence length is 8, first layer will have 8 cells, second layer $8+1$ cells as we add the previous layer output, and so on. The training of a n-step LSTM is sequential layer-wise as each layer is trained independently until convergence.

- Epochs $0 \rightarrow N$: layer$_1$ is trained, other layers are frozen and loss is only on $\tilde{v}_{t+1}$
- Epochs $N \rightarrow 2 \times N$: layer$_2$ is trained, other layers are frozen and loss is only on $\tilde{v}_{t+1}$

$\vdots$

- Epochs $(n-1) \times N \rightarrow n \times N$: all layers are unfrozen and the network is fine-tuned.

4. Experimental results

Unless specified otherwise, all presented models have been trained with an AdamW [29] optimizer set with a learning rate of 0.01 and a scheduler to make the learning rate decrease by a factor of 10 when validation metrics stagnate or increase over 3 consecutive validations. Training aims to minimize the Mean Square Error (MSE) between the prediction $\tilde{y}$ and the ground truth y *i.e.*, the value $\frac{1}{n}\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2$. For ablation studies, training seeds are fixed, and gradient descent is not stochastic: every batch contains the whole dataset and hence is an epoch.

4.1. One-minute forecasting

Ablation study over self-attention. Comparison between LSTM and SA-LSTM is presented in Table 1. We observe that LSTM and SA-LSTM are on-par on the Easy validation set, and SA-LSTM is significantly better on the Hard dataset. Hence, adding a self-attention layer to an LSTM allows for enhancing the quality of spatial dependencies predictions with no degradation of temporal dependencies.

		Validation set		
Method	Self-Attention	Easy	Hard	Time (ms)
LSTM	-	0.66	5.71	0.2
	✓	0.64	**4.53**	0.5

Table 1 Ablation study of LSTM and SA-LSTM on INRIX data for traffic forecasting. Metric is MSE scaled by $\times 10^3$. Time is inference time measured as the mean over 50,000 inferences after a warmup of 1,000 inferences.

Ablation study over Laplacian Pyramid loss. Experimental results are presented in Table 2. Training with this loss gave better results on one-minute predictions, particularly on the Hard dataset with a significant observed improvement. Indeed, this loss allows the model to focus the training on high-frequency details, which are more important in the Hard set. We observed the optimal depth to be 3 for our chosen hyper-parameters setting. Accuracy degrades for deeper depth than 3 because of the required pre-processing on tensors used for the Laplacian Pyramid loss during training only: we need the dimension of the inputs of this loss to be a multiple of 2^{depth}, and going to deep adds substantial empty padding. Therefore most

of the input becomes 0, which improves training metrics but significantly decreases validation metrics.

Method	Validation	Pyramid depth				
		0	1	2	3	4
SA-LSTM	Easy	0.64	0.64	**0.63**	**0.63**	0.65
	Hard	4.48	4.31	3.94	**3.59**	4.12

Table 2 Ablation study of SA-LSTM trained with Laplacian Pyramid Loss using several depths on INRIX data for traffic forecasting. Metric is MSE and scaled by $\times 10^{-3}$. Inference time is unchanged compared to SA-LSTM, as the core network is the same. Next experiments will fix the Laplacian Pyramid loss depth at 3.

Comparison with state-of-the-art methods. To validate our method, we compare inference time and validations accuracy with state-of-the-art spatio-temporal forecasting methods in Table 3. ConvLSTM [15] is a type of LSTM in which state-to-state and input-to-state transitions are replaced with convolutions. Self-Attention ConvLSTM [16] is a ConvLSTM whose transitions have been augmented with self-attention layers. Transformers [20] leverage attention to transform an input sequence into an output one by weighting how each elements of the input sequence interact one with each other. Interestingly, convolution-based methods trained with the Lap_3 loss led to a drop in accuracy on the easy validation set, while self-attention only methods experimentally benefit from it. SA-LSTM yields the best metrics on the Hard validation set and is comparable with the best method on the Easy one. Moreover, inference time is significantly lower than other methods designed for spatio-temporal forecasting and stays well under the intended millisecond.

Notably, both ConvLSTM and SA-ConvLSTM results on the Easy dataset degrade when training with a Laplacian Pyramid Loss but improve on the Hard dataset, as seen in the ablation in Table 3. More generally, we observed the Laplacian Pyramid Loss to improve all methods on the Hard validation dataset, however, SA-LSTM trained with Laplacian Pyramid Loss still outperforms other variations. Our intuition is that ConvLSTM-based models are by design highly focused on spatial dependencies and less on temporal ones than regular LSTM-based methods. Training with this loss worsens the spatial-dependency/ temporal-dependency analysis trade-off and over-advantages the analysis of spatial predictions over temporal ones.

Method	Lap_3 Loss	Validation Set		Time (ms)
		Easy	Hard	
LSTM	✗	0.66	5.71	**<u>0.2</u>**
	✓	**0.61**	4.09	
ConvLSTM	✗	0.63	5.15	3.7
	✓	0.68	5.13	
SA-ConvLSTM	✗	0.72	4.19	4.1
	✓	0.76	3.94	
Transformers	✗	0.65	5.03	1.8
	✓	0.64	4.71	
SA-LSTM	✗	0.64	4.52	<u>0.5</u>
	✓	0.63	**3.58**	

Table 3 Comparison of different forecasting methods and ablation study over the Lap_3 loss on INRIX data for traffic forecasting. Metric is MSE scaled by $\times 10^3$.

Overall, the model offering the best results on both the Easy and Hard datasets, so on temporal-focused and spatial-focused prediction, is the SA-LSTM. An example heatmap prediction from this network and corresponding traffic curve in different scenarios are represented in Figure 7 and Figure 10, in Section 5.

4.2. n-step forecasting

We compared different multi step forecasting methods and compared metrics on $t+1$, $t+2$ and $t+3$. We also compare running time, as we want our solution to run under the millisecond.

The most optimal results for $t+1$ are achieved using the recursive and 3-step methods. This is expected since the LSTM weights dedicated to this inference were trained specifically for 1-step forecasting using real INRIX data. In contrast, the underperforming all-at-once method is not as finely tuned for 1-step predictions. However, from $t+2$ onwards, the accumulation errors begin to impact the recursive method, which then gets surpassed by both the all-at-once and 3-step approaches, leading to similar performance metrics. This gap becomes even more pronounced at $t+3$, where both the all-at-once and 3-step methods significantly outpace the recursive method.

Method	Validation set	$t+1$	$t+2$	$t+3$	Total time (ms)
Recursive	Easy	**0.63**	0.83	1.24	1.8
	Hard	**3.58**	6.51	10.67	
All-at-once	Easy	0.70	**0.82**	**0.96**	<u>0.5</u>
	Hard	4.31	5.58	**7.43**	
n-step	Easy	**0.63**	0.83	1.03	<u>0.9</u>
	Hard	**3.58**	**5.41**	7.56	

Table 4 Comparison of different multi step forecasting methods. Metrics are MSE scaled by 10^3. Underlined running times are the one acceptable for our application case.

It's worth noting that the 3-step LSTM offers the best overall results for both single-step and multi-step predictions while maintaining sub-millisecond inference times. This highlights the method's proficiency in 1-step forecasting and its resistance to cumulative errors.

For our use case, n-step SA-LSTM appears as the best trade-off between inference time and both single and multi-step inference: $t+1$, $t+2$ and $t+3$ predictions are on-par with our best results overall while being faster than any other forecasting method except Vanilla LSTM. An example heatmap prediction from this method and corresponding traffic curve in different scenarios are represented in Figure 8 and Figure 11.

5. Qualitative observations

This section presents qualitative observations and analysis of some of our results. Presented qualitative representation comes in two forms of different granularity.

5.1. Heatmaps

We first present a comparison of ground truth and inferred speed profile plotted has heatmaps for different forecasting methods and for both single step forecasting and multi-step forecasting. These heatmaps bear 3D information and represent mean velocity of all the vehicles on each segment of the studied part of the I-24 highway at each time step. While this kind of representation can give an overall insight on the quality of the inference, it is in practice hardly analyzable with the naked eyes. This subsection present heatmaps for single step Lap_3 SA-LSTM, 3-step Lap_3 SA-LSTM, and for the

example of a failing case a 3 minute inference via the recursive method with the Lap_3 SA-LSTM.

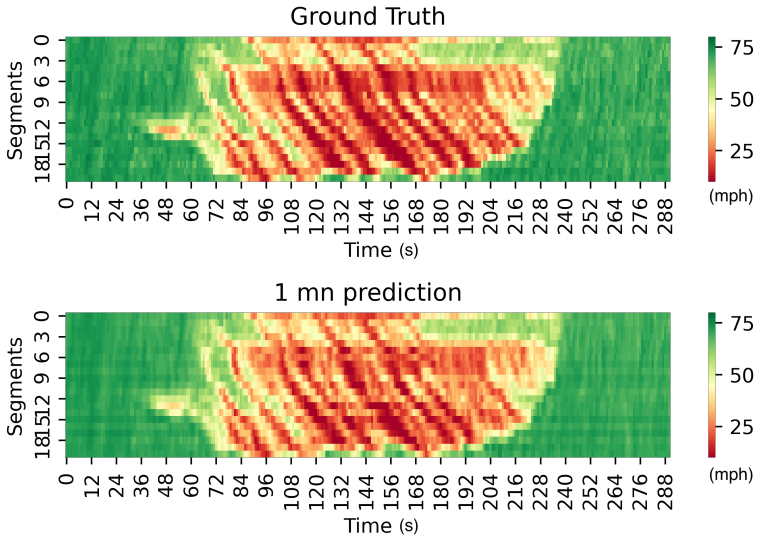


Figure 7 Comparison of heatmap generated from Lap_3 SA-LSTM one-minute traffic prediction with an heatmap generated from ground truth data. We observe inference to be as expected in both fluid and congested setup. Speeds are in miles/hour, time is in minute.

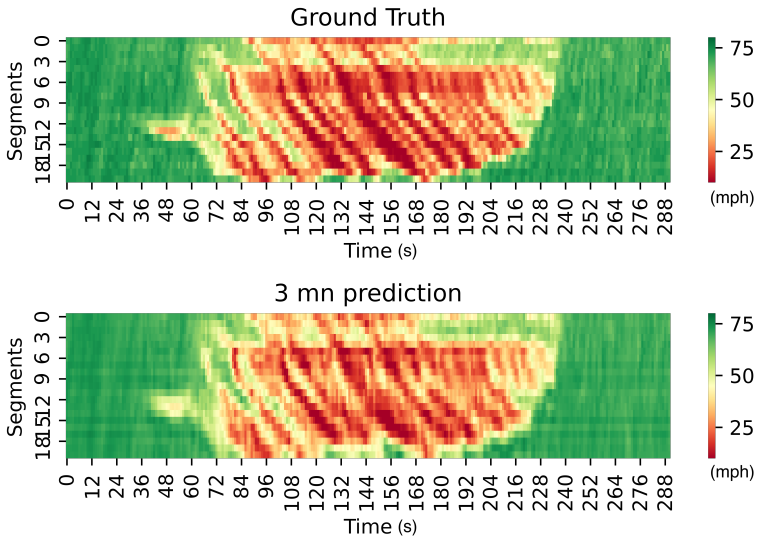


Figure 8 Comparison of heatmap generated from 3-step Lap_3 SA-LSTM three-minute traffic prediction with an heatmap generated from ground truth data. We observe inference to be as expected in both fluid and congested setup. Speeds are in miles/hour, time is in minute.

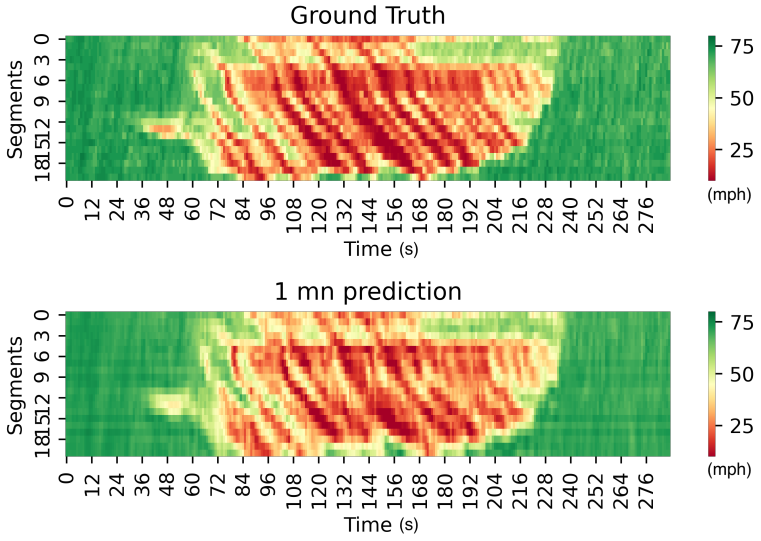
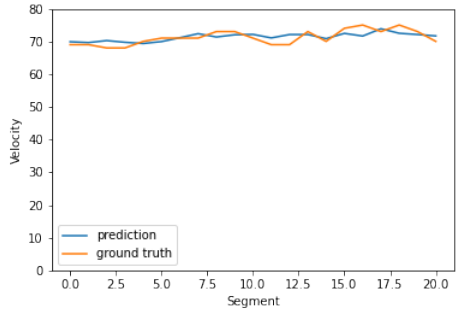


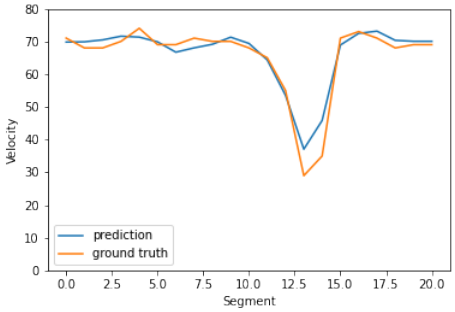
Figure 9 Comparison of heatmap generated from recursive Lap_3 SA-LSTM three-minute traffic prediction with a heatmap generated from ground truth data. We observe some blurr in the figure. This is due to the loss of accuracy caused by accumulation error.

5.2. Velocity curves in diverse stages of traffic

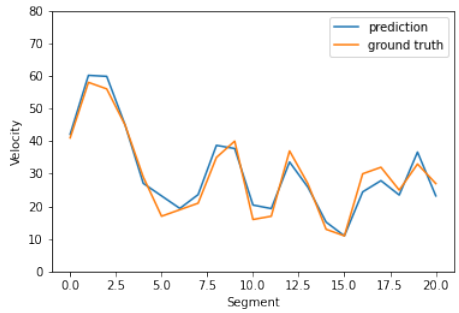
A more granular and easier to analyse type of representation is the plot of velocity curves in different stages of traffic. Contrarily to heatmaps, a velocity curve focuses on a single timestep and represents the relation between segment and mean velocity of the vehicles in it. This subsection present velocity curves in four representative stages of traffic (free flow of timestep 24, bottleneck of time 48, fully congested on time 180, and dissipation stage of timestep 216) for single step Lap_3 SA-LSTM, 3-step Lap_3 SA-LSTM, and for the example of a failing case a 3 minute inference via the recursive method with the Lap_3 SA-LSTM.



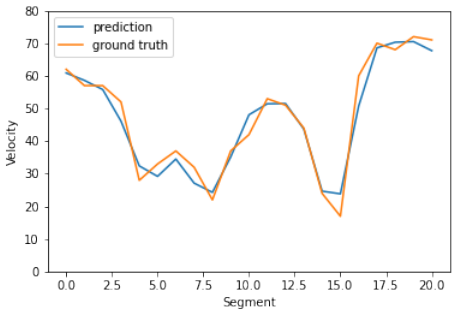
(a) Timestep 24: During the free flow stage, the prediction is able to generate the results around the free flow speed, 70 mile/hr.



(b) Timestep 48: A bottleneck start to form between segments 11 - 15. The prediction presents the same velocity change pattern with an accurate spatial location of the bottleneck as ground truth.

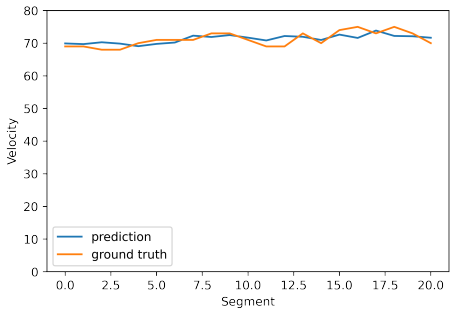


(c) Timestep 180: During the fully congested stage, the model is able to predict the propagation of the upstream shockwaves.

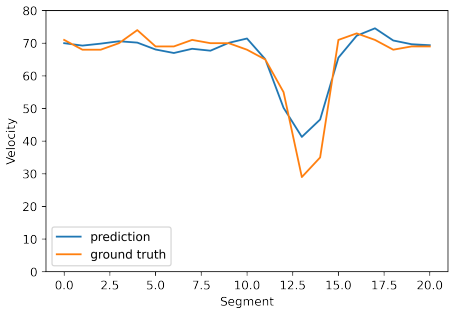


(d) Timestep 216: During the dissipation stage of the congestion, the prediction is able to capture the speed recovery at the bottleneck and upstream.

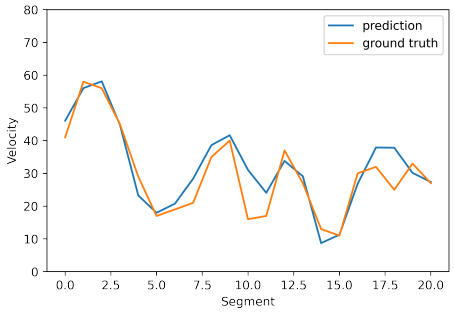
Figure 10 Comparison between the ground truth and the one-minute predictions from the Lap_3 SA-LSTM during different stages of the congestion lifecycle. Velocities are in mph.



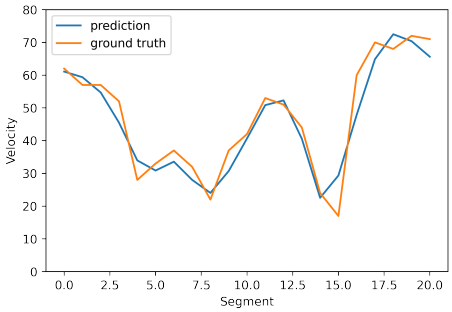
(a) Timestep 24: Prediction of free flow stage align with the ground truth around free flow speed.



(b) Timestep 48: A bottleneck start to form between segments 11 - 15. The prediction presents the same velocity change pattern with an accurate spatial location of the bottleneck as ground truth.

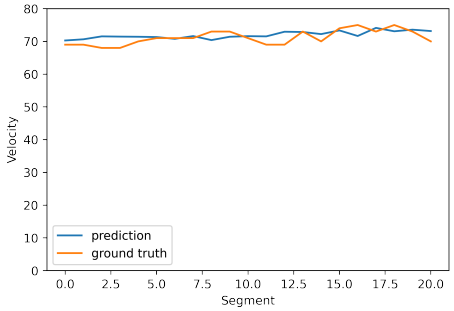


(c) Timestep 180: During the fully congested stage, the prediction captured the pattern of shockwave, while the prediction of absolute speed value has diversion from the ground truth. The predicted locations of the bottleneck and shockwaves are reliable.

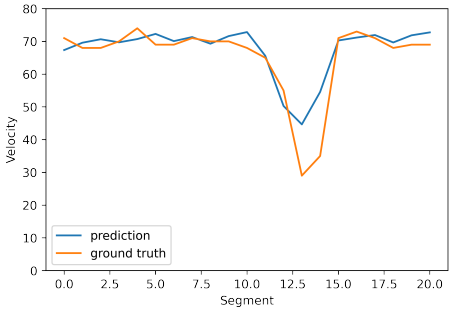


(d) Timestep 216: During the dissipation stage of the congestion, the prediction is able to capture the speed recovery at the bottleneck and upstream.

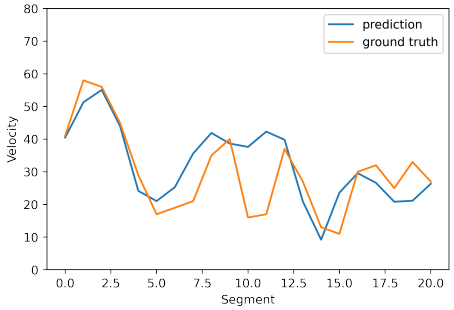
Figure 11 Comparison between the ground truth and the three-minute predictions from the 3-step Lap_3 SA-LSTM during different stages of the congestion lifecycle. Velocities are in mph.



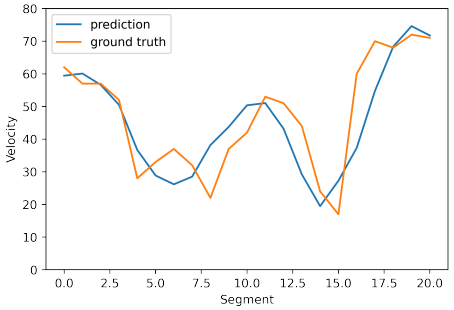
(a) Timestep 24: Prediction of free flow stage align with the ground truth around free flow speed.



(b) Timestep 48: A bottleneck start to form between segments 11 - 15. The prediction presents the same velocity change pattern with an accurate spatial location of the bottleneck as ground truth. The lowest speed at the bottleneck is underestimated.



(c) Timestep 180: During the fully congested stage, the prediction captured the pattern of shockwave, while the prediction of absolute speed value has diversion from the ground truth.



(d) Timestep 216: During the dissipation stage of the congestion, the prediction is able to capture the speed recovery at the bottleneck and upstream.

Figure 12 Comparison between the ground truth and the three-minute predictions from the recursive Lap_3 SA-LSTM during different stages of the congestion lifecycle. Velocities are in mph.

6. Conclusion

This paper tackles the problem of real-time mesoscale traffic forecasting, and presents a fast and accurate method able to extract and analyze both temporal and spatial dependencies in traffic data series. This approach has been analyzed through an extensive ablation study of its components, and compared with state-of-the-art methods for spatio-temporal forecasting to highlight how adapted it is for the studied task. Lastly, we introduced a novel technique for generalization of one-step forecasting method to multi-step forecasting. This method showed to provide the best trade-off inference time on both short-term and long-term forecasting for our considered use case.

References

[1] J. W. Lee, H. Wang, K. Jang, A. Hayat, M. Bunting, A. Alanqary, W. Barbour, Z. Fu, X. Gong, G. Gunter, S. Hornstein, A. R. Kreidieh, N. Lichtlé, M. W. Nice, W. A. Richardson, A. Shah, E. Vinitsky, F. Wu, S. Xiang, S. Almatrudi, F. Althukair, R. Bhadani, J. Carpio, R. Chekroun, E. Cheng, M. T. Chiri, F.-C. Chou, R. Delorenzo, M. Gibson, D. Gloudemans, A. Gollakota, J. Ji, A. Keimer, N. Khoudari, M. Mahmood, M. Mahmood, H. N. Z. Matin, S. Mcquade, R. Ramadan, D. Urieli, X. Wang, Y. Wang, R. Xu, M. Yao, Y. You, G. Zachár, Y. Zhao, M. Ameli, M. N. Baig, S. Bhaskaran, K. Butts, M. Gowda, C. Janssen, J. Lee, L. Pedersen, R. Wagner, Z. Zhang, C. Zhou, D. B. Work, B. Seibold, J. Sprinkle, B. Piccoli, M. L. D. Monache, A. M. Bayen, Traffic control via connected and automated vehicles: An open-road field experiment with 100 cavs (2024). arXiv:2402.17043.

[2] H. Wang, Z. Fu, J. Lee, H. N. Z. Matin, A. Alanqary, D. Urieli, S. Hornstein, A. R. Kreidieh, R. Chekroun, W. Barbour, W. A. Richardson, D. Work, B. Piccoli, B. Seibold, J. Sprinkle, A. M. Bayen, M. L. D. Monache, Hierarchical speed planner for automated vehicles: A framework for lagrangian variable speed limit in mixed autonomy traffic (2024). arXiv:2402.16993.

[3] G.-P. Antonio, C. Maria-Dolores, Multi-agent deep reinforcement learning to manage connected autonomous vehicles at tomorrow's intersections, IEEE Transactions on Vehicular Technology 71 (7) (2022) 7033–7043. doi:10.1109/TVT.2022.3169907.

[4] M. L. Delle Monache, T. Liard, A. Rat, R. Stern, R. Bhadani, B. Seibold, J. Sprinkle, D. B. Work, B. Piccoli, Feedback control algorithms for the dissipation of traffic waves with autonomous vehicles, Computational Intelligence and Optimization Methods for Control Engineering (2019) 275–299.

[5] R. E. Stern, S. Cui, M. L. Delle Monache, R. Bhadani, M. Bunting, M. Churchill, N. Hamilton, H. Pohlmann, F. Wu, B. Piccoli, et al., Dissipation of stop-and-go waves via control of autonomous vehicles: Field experiments, Transportation Research Part C: Emerging Technologies 89 (2018) 205–221.

[6] A. Derrow-Pinion, J. She, D. Wong, O. Lange, T. Hester, L. Perez, M. Nunkesser, S. Lee, X. Guo, B. Wiltshire, P. W. Battaglia, V. Gupta, A. Li, Z. Xu, A. Sanchez-Gonzalez, Y. Li, P. Velickovic, Eta prediction with graph neural networks in google maps, in: Proceedings of the 30th ACM International Conference on Information, CIKM '21, Association for Computing Machinery, New York, NY, USA, 2021, p. 3767–3776. doi:10.1145/3459637.3481916.
URL https://doi.org/10.1145/3459637.3481916

[7] Y. Yuan, F. Scholten, J. W. C. van Lint, Efficient traffic state estimation and prediction based on the ensemble kalman filter with a fast implementation and localized deterministic scheme, in: 2015 IEEE 18th International Conference on Intelligent Transportation Systems, 2015.

[8] B. Portugais, M. Khanal, Adaptive traffic speed estimation (2014).

[9] S. Ren, J. Bi, Y. F. Fung, X. I. Li, T. K. Ho, Freeway traffic estimation in beijing based on particle filter, in: 2010 Sixth International Conference on Natural Computation, 2010.

[10] X. Ma, S. Luan, C. Ding, H. Liu, Y. Wang, Spatial interpolation of missing annual average daily traffic data using copula-based model, IEEE Intelligent Transportation Systems MagazineIF: 3 (2019).

[11] I. Laña, J. L. Lobo, E. Capecci, J. Del Ser, N. Kasabov, Adaptive long-term traffic state estimation with evolving spiking neural networks, Transportation Research Part C: Emerging TechnologiesIF: 3 (2019).

[12] K. Guo, Y. Hu, Z. Qian, Y. Sun, J. Gao, B. Yin, Dynamic graph convolution network for traffic forecasting based on latent network of laplace matrix estimation, IEEE Transactions on Intelligent Transportation SystemsIF: 3 (2020).

[13] T. Mallick, P. Balaprakash, J. Macfarlane, Deep-ensemble-based uncertainty quantification in spatiotemporal graph neural networks for traffic forecasting (2022).

[14] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural computation 9 (1997) 1735–80. doi:10.1162/neco.1997.9.8.1735.

[15] X. Shi, Z. Chen, H. Wang, D. Yeung, W. Wong, W. Woo, Convolutional LSTM network: A machine learning approach for precipitation nowcasting, CoRR abs/1506.04214 (2015). arXiv:1506.04214.
URL http://arxiv.org/abs/1506.04214

[16] Z. Lin, M. Li, Z. Zheng, Y. Cheng, C. Yuan, Self-attention convlstm for spatiotemporal prediction, in: The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020, AAAI Press, 2020, pp. 11531–11538.
URL https://ojs.aaai.org/index.php/AAAI/article/view/6819

[17] W. Li, F. Qi, M. Tang, Z. Yu, Bidirectional lstm with self-attention mechanism and multi-channel features for sentiment classification, Neurocomputing 387 (2020) 63–77.

[18] D. Deng, L. Jing, J. Yu, S. Sun, Sparse self-attention lstm for sentiment lexicon construction, IEEE/ACM transactions on audio, speech, and language processing 27 (11) (2019) 1777–1790.

[19] R. Jing, A self-attention based lstm network for text classification, in: Journal of Physics: Conference Series, Vol. 1207, IOP Publishing, 2019, p. 012008.

[20] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, I. Polosukhin, Attention is all you need, in: I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (Eds.), Advances in Neural Information Processing Systems, Vol. 30, Curran Associates, Inc., 2017.
URL https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[21] Q. Wen, T. Zhou, C. Zhang, W. Chen, Z. Ma, J. Yan, L. Sun, Transformers in time series: A survey (2022). doi:10.48550/ARXIV.2202.07125.
URL https://arxiv.org/abs/2202.07125

[22] N. Wu, B. Green, X. Ben, S. O'Banion, Deep transformer models for time series forecasting: The influenza prevalence case, CoRR

abs/2001.08317 (2020). arXiv:2001.08317.
URL https://arxiv.org/abs/2001.08317

[23] W. Jiang, J. Luo, Graph neural network for traffic forecasting: A survey, CoRR abs/2101.11174 (2021). arXiv:2101.11174.
URL https://arxiv.org/abs/2101.11174

[24] L. Zhao, Y. Song, C. Zhang, Y. Liu, P. Wang, T. Lin, M. Deng, H. Li, T-gcn: A temporal graph convolutional network for traffic prediction, IEEE Transactions on Intelligent Transportation Systems 21 (9) (2020) 3848–3858. doi:10.1109/TITS.2019.2935152.

[25] T. Bogaerts, A. D. Masegosa, J. S. Angarita-Zapata, E. Onieva, P. Hellinckx, A graph cnn-lstm neural network for short and long-term traffic forecasting based on trajectory data, Transportation Research Part C: Emerging Technologies 112 (2020) 62–77. doi:https://doi.org/10.1016/j.trc.2020.01.010.
URL https://www.sciencedirect.com/science/article/pii/S0968090X19309349

[26] T. Reed, Inrix global traffic scorecard (2019).

[27] E. L. Denton, S. Chintala, A. Szlam, R. Fergus, Deep generative image models using a laplacian pyramid of adversarial networks, CoRR abs/1506.05751 (2015). arXiv:1506.05751.
URL http://arxiv.org/abs/1506.05751

[28] H. Ling, K. Okada, Diffusion distance for histogram comparison, in: 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06), Vol. 1, 2006, pp. 246–253. doi:10.1109/CVPR.2006.99.

[29] I. Loshchilov, F. Hutter, Fixing weight decay regularization in adam, CoRR abs/1711.05101 (2017). arXiv:1711.05101.
URL http://arxiv.org/abs/1711.05101