Attention to Traffic Forecasting: Improving Predictions with Temporal Graph Attention Networks

Raúl Gadea and José L. Aznarte

***Abstract*—Dynamic traffic flow forecasting remains an open issue to this day. As other spatio-temporal problems, traffic prediction deals with both temporal and spatial nonlinear relationships, with the particularity that nearby points in the Euclidean space might be allocated in different roads, adding another layer of complexity. Traffic prediction has witnessed a revolution with the appearance of deep learning, with graph neural networks being prominently responsible for a steep increase in forecasting accuracy.**

In this paper, we consider the use of an automatic attention mechanism in order to improve the prediction capabilities of a traffic graph convolutional network. This model is based on the composition of gated recurrent units and graph convolution networks to model space and time simultaneously. To overcome the spatial modelling limitations of the original model, our proposal replaces the graph convolutional layer with a graph attention mechanism. Our aim is to model spatial relations in an automatic, more dynamic way.

In order to prove the validity and usefulness of our proposal, we have performed a thorough experimentation over two known traffic datasets used in previous research, plus a new, complex one which we have curated and published. Our results portray a clear and statistically significant advantage with the inclusion of spatial attention, surpassing the performance of a wide set of state-of-the-art models on every tested scenario.

I. Introduction

Dynamic traffic forecasting aims to predict future values of traffic-related variables such as speed or intensity, given historical and current values provided by sensors allocated along the road. It has many important applications, such as route selection, trip time estimation or better traffic flow control to reduce its impact on air quality.

It is also a particularly complex modelling problem given that it implies to model not only relations among points with different timestamps, but also spatial relations between adjacent points. To make it even more interesting, nearby points in Euclidean distance might be allocated in very different road segments, making their spatial relationship weak or even unexistent.

As an example, in Figure 1 both the spatial and temporal correlations are shown for four urban traffic sensors of the city of Madrid. Sensor 3598 appears in green, 6699 in red, 6700 in orange and 6770 in blue. All sensors present temporal correlation, presenting a clear daily seasonality. However, the spatial correlation is very particular. It can be seen how sensor

R. Gadea and J.L. Aznarte are with the Artificial Intelligence Department, Universidad Nacional de Educación a Distancia — UNED
Email: {rgadea | jlaznarte}@dia.uned.es

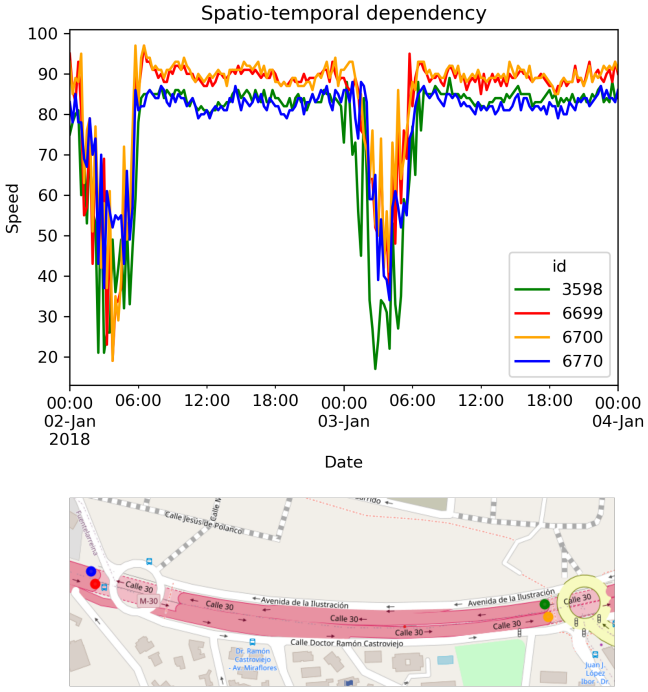

Fig. 1. Spatial and temporal traffic intensity dependency of for sensors in two different lanes

6699 presents a more similar curve to sensor 6700 than to 6770, even though they are farther in euclidean space. This occurs because sensors 6699 and 6700 are connected in the same lane, whereas 6770 is situated in a parallel lane pointing on opposite direction. This same phenomena can be seen between sensors 3598 and 6770.

Graph neural networks (GNN) [1] encode spatial relations in a mathematical structure that can represent the actual connections in the roads, and thus are in principle particularly well suited for problems such as this one (although some research questions this intuition [2]). These neural networks allow to model spatial relationships beyond Euclidean ones, making it a disruptive technique in the traffic forecasting context.

Many different approximations using GNN to forecast traffic-related variables have arisen over the last years. The temporal graph convolutional network (T-GCN) [3] is one of the simplest and more popular ones. It combines Gated Recurrent Units (GRU) [4] to model time with a standard graph convolutional network (GCN) [5] to model space.

However, experience from other fields shows that standard GCN is not always the best choice to capture inherent spatial relations, and that attention mechanisms are a better approach for this task. Accordingly, in this paper, an evolution of the T-GCN is proposed. Instead of modelling space with a GCN, we prove how modelling space with spatial attention mechanisms through the use of a graph attention network (GAT) [6] results in a better model, which we call temporal graph attention network (T-GAT).

Our work tries to give answers to some of the main challenges for traffic forecasting identified in [7]. That review paper tried to set a landmark on the field by identifying a set of needed future developments for a useful and feasible prediction of traffic. Among those challenges, we deal with increased data resolution, aggregation and quality (challenge 4), temporal characteristics and spatial dependencies (challenge 6), model selection and testing (challenge 7) and, especially, realizing the full potential of artificial intelligence (challenge 10).

The main contributions of this work are as follows:

- We prove the usefulness of attention mechanims for traffic forecasting, introducing T-GAT, a new deep learning model which uses spatial attention to improve over preexisting graph convolutional networks.
- We report on extensive experiments which show how this approach obtains significant improvements over other state-of-the-art baseline methods including the original T-GCN.
- We gather, prepare, process and publish a dataset of Madrid's M30 Highway traffic flow speed measures, thus creating a new graph forecasting dataset available for further experimentation. We carefully imputed all missing and erroneous data and created the adjacency matrix using driving distance instead of Euclidean distance.

The rest of the paper is organized as follows. In Section II a review of previous traffic flow forecasting methods is presented. Section III introduces the original T-GCN design as well as the modifications that lead to the novel T-GAT model. Section IV includes all data wrangling necessary to prepare the Madrid traffic intensity data for training and prediction. Section IV explains all experiments and presents and analyze their results. Section V presents our conclusions.

II. Related work

A. Traffic flow forecasting

Traffic flow forecasting is a classical problem in the transport engineering field and is gaining more and more attention every day. Solutions are categorized in two major groups: Parametric and non-parametric approaches. Some of the most conservative parametric models include historical average [8], lineal regression [9] or Kalman filtering models [10].

One of the oldest and most used time series prediction models is the autoregressive integrate moving average model (ARIMA) [8], which is still relevant to this day. Over the years ARIMA has been subject to modifications in order to improve its prediction precision, such as Kohonen ARIMA [11], subset ARIMA [12], seasonal ARIMA [13], and many more. Even though these methods give fairly good results in general, they depend on a stationarity hypothesis and therefore are not sufficient for a more general solution in traffic forecasting. Non-parametric methods can tackle this problem because of their strong learning abilities given the adequate data, allowing to learn non-stationary patterns. These methods include K-nearest [14], support vector regression (SVR) [15], fuzzy logic [16], Bayesian networks [17], and, finally, neural network models.

Recently, neural networks and specially deep learning techniques have been rapidly evolving, obtaining outstanding results in areas such as computer vision or natural language problems. Given the sequential nature of traffic flow data, Recurrent Neural Netowrks (RNN) are the most extended technique for sequence modelling. The main problem with plain RNNs is their inability to capture long-term dependencies due to vanishing gradients. In order to correct this problem, newer proposals such as long-short term memory neural networks (LSTM) [18] and gated recurrent units (GRU) [4] were developed in order to capture long dependencies using memory mechanisms. In [19], the authors make a comparison between ARIMA models and deep learning approaches such as LSTM or GRU networks, resulting on a clear victory of the latter.

These networks obtain great results when working with temporal data, however they find it harder to address the spatial side of the traffic forecasting problem. In order to address this situation much of the work went to develop a combination of RNNs architectures for time modelling with convolutional neural networks (CNNs) for spatial modelling [20]. One interesting example of this combination is given in [21], where the authors develop a modular approach consisting in a spatial module, a time module and the later combination of both. Apart from being the state of art working with spatial data such as images, CNNs are also a disrupting methodology in time series prediction, being particularly suitable for spatio-temporal data such as traffic forecasting. In [22] researchers transform traffic data to an image representation in order to obtain a prediction.

B. GNNs

The main issue using CNNs to capture spatial relations among road points is that, in the standard representation where convolution can be applied, every "node" appears as equidistant and adjacently connected from one another. Needless to say, city roads hardly ever meet this property.

In recent years, graph neural networks (GNNs) [1] have regained attention, achieving exceptional results in many problems with a similar structure as traffic forecasting. This sort of networks allow the computation of spatial relations in a non-Euclidean space and therefore can create asymmetrical representations of roads.

Most taxonomies in this field classify GNNs depending on the technique utilized to share information between nodes: recurrent graph neural networks (RecGNNs) and convolutional graph neural networks (ConvGNNs) [23] [24].

1) Recurrent graph neural networks (RecGNNs): The first ever published graphical neural network was developed under the now common recurrent graph neural network label [1]. These networks represent nodes by using recurrent neural networks as a message passing device. RecGNNs update node states by exchanging neighbourhood information recurrently until a stable equilibrium is reached. The first versions where based on pure RNNs, but limits were found rapidly so more refined RecGNNs where developed such as the GRU based RecGNNs, known as gated graph neural network (GGNN) [25].

2) Convolutional graph neural networks (ConvGNNs): ConvGNNs are closely related to recurrent graph neural networks. However, ConvGNNs use a fixed number of layers to address the architectural mutual dependencies between nodes, which make them much faster and stable to train. They fall under two different categories depending if the method approached the problem in a spectral or spatial manner.

a) Spectral-based ConvGNNs: Spectral-based ConvGNNs are rooted in the mathematical foundations used in graph signal processing. Each ConvGNNs differ from each other on the filter used to de-noise the graph. Some of the most famous methods developed under this category are spectral CNNs or the ChebNet [26]. The most influential of them all has been GCN [5], which inspired other scientist to develop networks such as adaptive graph convolutional network (AGCN) [27] or dual graph convolutional network (DGCN) [28].

b) Spatial-based ConvGNNs: Spatial-based ConvGNNs use a similar approach to CNNs to obtain node relationships. However, instead of equally distributed and connected nodes, Graphs can adapt convolutions to broader spectrum of typologies. In recent years, these methods have been developing rapidly, and currently are without a doubt the most widely used graphical neural networks. This is partly due to efficiency, generality, and flexibility issues. Spatial-based models are more flexible when handling multi-source graph inputs such as edge inputs, directed graphs, signed graphs, and heterogeneous graphs, because these graph inputs can be incorporated into the aggregation function easily. Neural network for graphs (NN4G) [29], was the first work towards spatial-based ConvGNNs. Distinctively different from RecGNNs, NN4G learns graph mutual dependencies through a compositional neural architecture with independent parameters at each layer. After this, many other spatial-based ConvGNNs followed, such as: contextual graph Markov model (CGMM) [30], diffusion convolutional neural network (DCNN) [31] and GraphSAGE [32].

Another important milestone in the development of spatial-based convolutional GNNs was the addition of attention mechanisms to the graph, allowing the network to learn the importance of each edge while training. The Graph Attention Network (GAT) [6] was the first work to introduce attention into a graph neural network, later succeeded by GATv2 [33], which refined this concept. On the other hand, Gated attention networks (GaAN) [34] use a convolutional sub-network to control each attention head's importance, unlike the traditional multi-head attention mechanism, which consumes equally all attention heads.

C. Spatio-temporal Graph Neural Networks

In many real-world applications, graphs are dynamic both in terms of structure and inputs. Spatio-temporal graph neural networks (STGNNs) aim to learn hidden patterns from spatio-temporal graphs by capturing its dynamics. This group of methods are the state of art solving traffic flow forecasting problems. Most of these methods are combinations of ConvGNNs or RecGNNs, with more classical temporal approaches such as RNNs and CNNs.

1) RNN based approaches: Several RNN have been applied to traffic forecasting. For instance, a model called graph gated recurrent unit (GGRU) [34], with GaAN as a building block, was developed to address the traffic speed forecasting problem. Also the diffusion convolutional recurrent neural network (DCRNN) [35] captures spatial dependencies using bidirectional random walks on the graph, and the temporal dependency using the encoder-decoder architecture with scheduled sampling. The temporal graph convolutional network (T-GCN) [3] combines GNNs with a GRU architecture to simultaneously capture both spatial and temporal dependencies. Following this line of work, the attention temporal graph convolutional network (A3t-GCN) [36] added a temporal attention mechanism on top of the T-GCN architecture, achieving even better performance. Similarly as with plain RNNs, these networks are usually computationally expensive to train.

2) CNN based approaches: To tackle the RNNs issues, 1-dimensional CNNs were also used to capture the temporal dependencies. Spatio-temporal graph convolutional networks (STGCN) [37] effectively capture comprehensive spatio-temporal correlations by modelling multi-scale traffic networks. This model integrates 1D convolutional layers with ChebNet or GCN layers. On the other hand, the graph multi-attention network (GMAN) [38] and the attention-based spatial-temporal graph convolutional network (ASTGCN) [39] both incorporate an attention mechanism to better model spatial temporal correlation. Finally, the graph wavenet [40], uses a dilation convolution in order to capture long-term relationships tackling one of the bigger problems in temporal convolution, and introduces a self-learned adjacency matrix witch learns nodes relations while training.

III. Methodology

In the following section our proposed work is introduced and mathematically formulated. It consists in the update of the T-GCN presented in section II-C1 with a spatial attention layer GATv2 introduced in II-B2b. This change will presumably allow the network to better capture the spatial relation between nodes.

A. Problem definition

In this paper, the road network is represented as a weighted directed graph $G = (V; E)$ where V represents nodes on the graph and E are the connections between nodes. In this

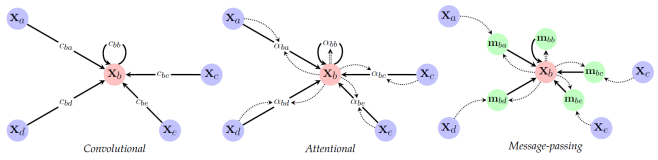

Fig. 2. The three different flavours of common GNN architectures.

particular scenario, each road sensor is represented as a node $V = \{v_1, v_2, ..., v_N\}$, where N is the number of nodes in the graph. Sensors are connected using a driving distance proximity measure by a set of edges represented with the weighted adjacency matrix $A \in \mathbb{R}^{N \times N}$. The adjacency matrix contains 0 when there is no connection between nodes and a real number between 0 and 1 otherwise, being 1 the strongest connection possible.

The traffic univariate information variable, such as vehicle speed, is represented with the matrix $X \in \mathbb{R}^{N \times P}$, where P denotes the length of the time series and $X_i \in \mathbb{R}^{N \times i}$ the traffic information of the whole graph at time i.

The model goal is to obtain a function f that given the road structure and the historic time series of traffic is able to predict the traffic variable for T points into the future.

$$[X_{t+1}, X_{t+2}...X_{t+T}] = f(G; (X_{t-n}, X_{t-n+1}, ..., X_{t-1})), \tag{1}$$

where n represents the number of historic values used as input for prediction and T the number of future output values to predict.

B. Temporal dependence modelling

Temporal dependence modelling is a key part in any forecasting problem. As stated above, there exists two main approaches among deep learning techniques for such task: the use of RNNs or CNNs. Each technique has its strengths and weaknesses, however RNNs are more extended for sequence modelling mainly due to their longer application history in this domain. To better capture long time dependencies a GRU-based network is proposed given that it has less gates and therefore less parameters to train than the LSTM counterpart, while usually achieving comparable results.

C. Spatial dependence modelling with attention

Spatial dependence modelling is the other main focus of this paper. Given the structural disposition of traffic forecasting problems, GNNs have risen as the main tool to perform spatial convolutions in this non-Euclidean domain. Following the literature [41], GNNs can be categorized into 3 different flavours: convolutional, attentional and message-passing. One important thing to note is that there is a representational containment between these approaches: convolutional $\subseteq$ attentional $\subseteq$ message-passing. Each one of them can be seen as a particular case of the following one, being message-passing the most general.

The aforementioned T-GCN model uses the convolutional flavour as its spatial dependence modelling unit:

$$h_u = \phi\Big(x_u, \bigoplus_{v \in N_u} c_{uv}\psi(x_v)\Big), \tag{2}$$

where ϕ and ψ are learnable affine transformations with activation functions, $\bigoplus$ is a permutation invariance non-parametric operation such as the sum, mean or maximum, N_u corresponds to the neighbourhood of the node u and c_{uv} specifies the importance of node v to node u's representation. Its value is the one presented in the weighted adjacency matrix A.

The T-GCN draws from [42] to define its convolutional layer. A two layered network would then be as follows:

$$f(X, A) = \sigma\left(\widehat{A}\ \mathsf{ReLU}\big(\widehat{A}XW_0\big)W_1\right), \tag{3}$$

where X represents the feature matrix, A represents the weighted adjacency matrix, $\widehat{A} = \tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}$ denotes the preprocessing step, $\tilde{A} = A + I$ is a matrix with the self-connection structure, $\tilde{D}$ is a degree matrix, with $\tilde{D} = \sum_j \tilde{A}_{ij}$. W_0 and W_1 represent the weight matrix in the first and second layer, and $\sigma()$ and $\mathsf{ReLU}()$ are activation functions.

On the other hand, in order to overcome some limitations of the T-GCN, our proposed model uses the attentional flavour as its spatial dependence modelling unit:

$$h_u = \phi\Big(x_u, \bigoplus_{v \in N_u} a(x_u, x_v)\psi(x_v)\Big). \tag{4}$$

As seen in equation 4 the main difference between the convolutional and attentional flavours is that in the convolutional case cuv has a fixed value, whereas in attention-based networks it is a trainable function $a(x_u, x_v)$. The attentional layer used for our model is the GATv2 [33], which is a refined actualization of the famous GAT [6], in which

$$f_u = \sigma\Big(\sum_{v \in N_u} a(x_u, x_v)Wxv\Big), \tag{5}$$

$$a(x_u, x_v) = \frac{exp(e(x_u, x_v))}{\sum_{w \in N_w} exp(e(x_u, x_w))}, \tag{6}$$

and

$$e(x_u, x_v) = a^\top \mathsf{LeakyReLU}(W \cdot [x_u || x_v]). \tag{7}$$

D. T-GAT

Elaborating over previous equations, the specific calculation process for the T-GAT is as follows. The update gate, u_t, is:

$$u_t = \sigma\Big(W_u f(A, [X_t, h_{t-1}]) + b_u\Big), \tag{8}$$

where f is the equation 7 explained above, A is the adjacency matrix, X_t the feature matrix for a specific timestamp, h_{t-1} the previous hidden state and W_u and b_u are trainable parameters. On the other hand, the memory gate is given by:

$$r_t = \sigma\Big(W_r f(A, [X_t, h_{t-1}]) + b_r\Big), \tag{9}$$

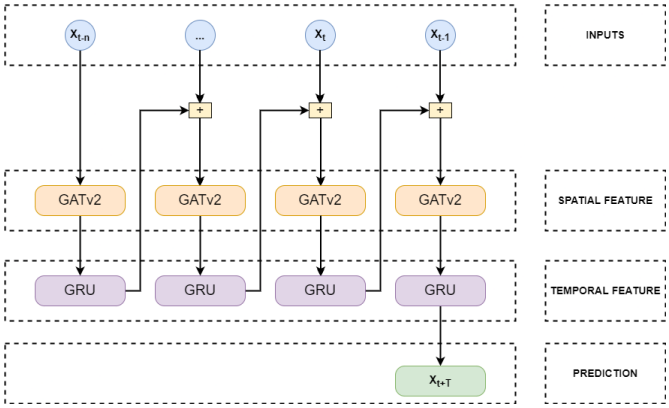

Fig. 3. Schematics of the T-GAT general architecture, with n historical input values, in blue, for a T horizon forecasting, in green. First, in yellow, a spatial attention convolution is applied to the concatenation of the historical value with the hidden state of the previous layer. Then, in purple, the gated recurrent unit calculates next hidden state to repeat the process all over again. The initial hidden state is set to zero.

where W_r and b_r are trainable parameters. The new hidden state candidate is:

$$c_t = tanh\Big(W_c f(A, [X_t, r * h_{t-1}]) + b_c\Big), \tag{10}$$

where W_c and b_c are also trainable parameters and the new hidden state is given by:

$$h_t = u_t * h_{t-1} + (1 - u_t) * c_t. \tag{11}$$

Its overall architecture can be seen in figure 3.

IV. Data

All models have been tested over two previously published datasets plus a new, unpublished, one. In this section, the two standardized datasets as well as all data wrangling to create the M30 Dataset are detailed.

A. Standardized datasets

These datasets are the same ones tested in the original T-GCN paper:

- **SZ-traffic dataset**: This dataset corresponds to traffic speed measures of the city of Shenzhen from 01/01/2015 to 31/01/2015 for a selection of 156 sensors, one sensor per street.This dataset contains an adjacency matrix of 156×156 containing the relationship among nodes and a historic feature matrix for all traffic speed measures with a resolution of 15 minutes. The mean of connections between nodes is 3.41 with a standard deviation of 1.19.
- **Los-loop dataset**: This dataset corresponds to traffic speed measures of the highway in Los Angeles county from 01/03/2012 to 07/01/2012 for a selection of 207 sensors along the road. This dataset contains a weighted adjacency matrix of 207×207 containing the relationship among nodes based on euclidean distance and a historic feature matrix for all traffic speed measures with a resolution of 5 minutes. The mean of connections between nodes is 13.69 with a standard deviation of 5.01. Data has been imputed exactly like in the original paper.

B. M30 Dataset

1) Data analysis: The M30 dataset is obtained as a subset of two main data sources:

- **Madrid Traffic Flow Dataset:** This is the main data source of the project. It is provided by the Municipality of Madrid through its open data portal [43]. This dataset contains historical measurements of traffic intensity, in number of cars per hour, and speed in meters per second for over 4.000 sensors of the city of Madrid, with a 15 minute resolution for at least 8 years of history at the time of writing. Every sensor is located by their coordinates (longitude and latitude).
- **Open Street Map**: The Open Street Map Portal [44] has been used to create the graph in order to calculate driving distances between points rather than just using Euclidean distance.

The remote sensing network of the city of Madrid has been improved over the years, adding sensors and measures. The M30 is a circular highway with a length of 32.5 Km and an average radius of 5.17 Km, supporting an average traffic intensity of around 300.000 vehicles per day. The published dataset comprises 1 month of historical data of sensors corresponding to the M30 highway. For the selected period, from 01/01/2018 to 31/01/2018, only sensors that are consistent in position and time are used. After applying this filter, the dataset consists on 349 sensors.

2) Data imputation: Before doing any modelling, it is crucial to assert that there exists a measure point for every timestamp. Unfortunately, sensors regularly provide faulty measures, in the form of `NaN` values or long periods of the same constant value, see Figure 4.

In order to solve this problem it is necessary to both identify and correct this behaviour. Two thresholds have been developed to locate faulty repeated measures, one for the zero measure and another for every other value. This is done due to the fact that repeated zero values are not always erroneous and therefore do not need imputation, however repeated measures of values different than zero are highly unlikely given the measure precision. This thresholds are fixed as 24 repeated values for zero and 5 repeated values otherwise. After identifying faulty measures they are deleted and prepared for imputation.

The imputation method selected is based on the additive Facebook's prophet library [45], which is able to capture three main model components: trend, seasonality and holidays. Contrary to RNNs, Prophet can predict with `NaN` values in its historic, and therefore can be used as an imputation method based on trendy and seasonal interpolation. An example of this imputation can be seen in figures 4 and 5. Nevertheless, only data with 95% informed points are used for modelling as data quality minimum criterion.

3) Driving distance: Another important task is to obtain driving distances between nodes rather than Euclidean distance to construct the graph, since two sensors close in space might be further away in driving distance. In figure 6 it can be seen how two near sensors in different driving lanes are not connected. Furthermore, despite Euclidean distance

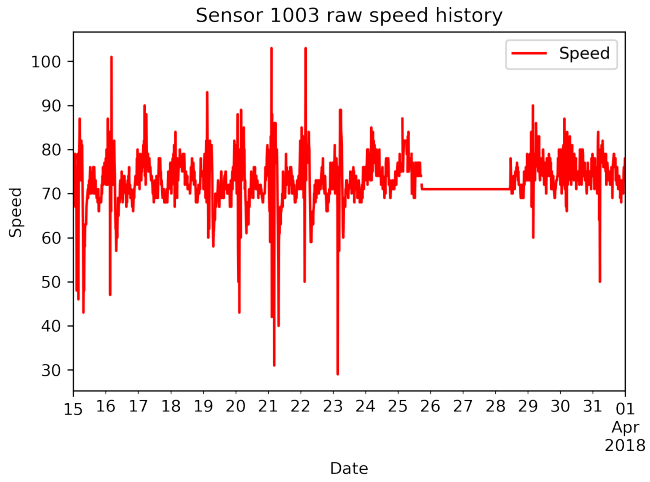

Fig. 4. Raw data example.

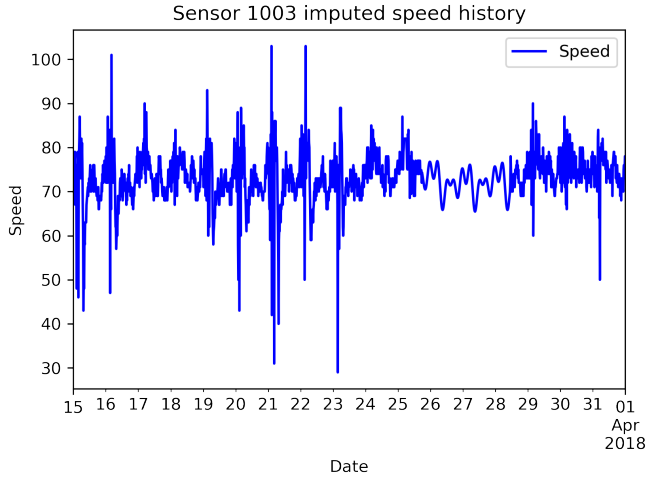

Fig. 5. Imputed data example.

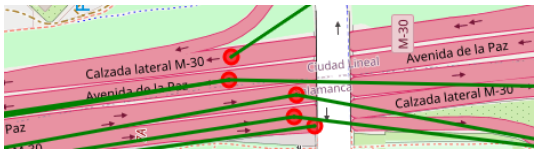

Fig. 6. Nearby sensors are not connected when allocated in different lanes.

is commutative, driving distance is not. The trajectory from point A to point B might be longer than from point B to point A. This property allows to work with a directed graph rather than with an undirected one. The Open Street Map API [44] has been used for this task. However, obtaining all driving distance for every node among each other is too computationally expensive without better performance. Thus, the API call is only performed for the 10 nodes closer to each other in Euclidean distance.

4) Temporal Graph creation: The graph is structured as a static temporal graph, meaning that all connections and nodes are static through time. In order to create the connections for each node, as well as its weight, we compute the pairwise road

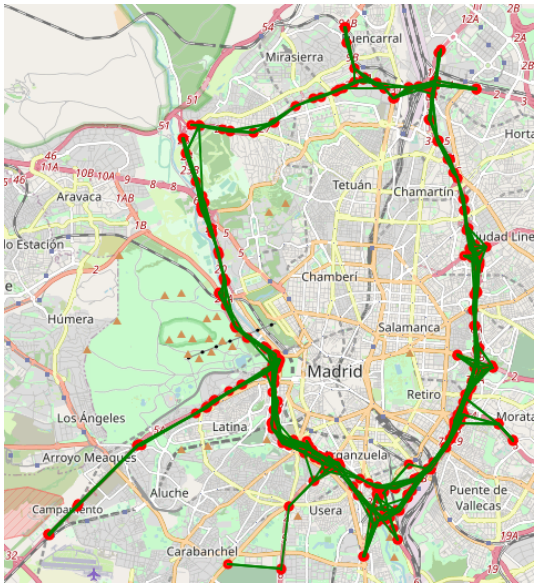

Fig. 7. Final M30 computed graph.

network distances between sensors and build the adjacency matrix using the thresholded Gaussian kernel [46] used in other traffic forecasting related papers [31]. In order to avoid the graph to establish connections between nodes that already have another node between them, a maximum connections parameters is set to 3. After this, a weighted adjacency matrix of 349×349 is produced, with a node connection mean of 2.88 and with a standard deviation of 0.39. In Figure 7 the M-30 Highway graph built with this methodology can be seen.

V. Experiments

The data and the code used for experimentation in this paper can be accessed in https://github.com/raulgadea/T-GAT.

A. Evaluation metrics and benchmark models

In order to compare and contrast the proposed modification capabilities with the original T-GCN, the same standard metrics have been used:

1) Root Mean Squared Error (RMSE):

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(Y_t - \hat{Y}_t^{\,2})} \tag{12}$$

2) Mean Absolute Error (MAE):

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|Y_t - \hat{Y}_t| \tag{13}$$

Likewise, to keep our comparison comprehensive and in order to validate our results with a variety of different approaches, we have kept the benchmarking models used by the original proposal, while also considering the aforementioned GAT:

1) **Historic Average (HA):** Uses the average traffic values of each node as its prediction.

2) **Support Vector Regressor (SVR)**: A regression model based on support vector machines. This implementation uses de linear kernel and the penalty term 0.001.
3) **Graph Convolutional Network Model (GCN)**: A graph convolutional model with historic values encoded as variables. [5]
4) **Graph Attentional Network Model (GAT)**: A graph attentional model with historic values encoded as variables. [33]
5) **Gated Recurrent Unit model (GRU)**: A gated recurrent neural network treating each node as a disconnected entity. [4]
6) **Temporal Graph Convolutional Neural Network (T-GCN)**: A combination of Graph convolutions with a gated recurrent unit. [3]

B. Experimental design

T-GCN experimental design has been replicated to better capture the impact of the inclusion of the graph attention layer in the architecture. Four forecasting horizons have been tested: 15, 30, 45 and 60 minutes.

Regarding training, data has been normalized between [0, 1] and split into a 80% for training and 20% for testing. Hyper-parameters are the same as in T-GCN original paper: Adam optimizer with a learning rate of 0.001, batch size of 64, a L2 regularization with $\lambda = 1.5 \cdot 10^{-3}$ and hidden dimensions of 100, 64 and 128 for Shenzhen, Losloop and M30 respectively. The only exception occurs with the application of the T-GAT model in the Losloop dataset, which required a more delicate tuning.[1]

C. Results

Tables I, II and III present RMSE and MAE performance of the Temporal Graph Attention Network against the baselines for the Shenzen, Losloop and M30 datasets respectively. All deep neural networks include a standard deviation parameter given that results are forecast in batches. Additional information about the error distribution can be seen in figures 8, 9 and 10. In figure 11 the temporal evaluation of the RMSE is also provided.

It can be seen how the application of graph attention networks improve the performance in every scenario. The comparison of plain graph convolutions with and without spatial attention, results in a exceptional improvement. GAT not only surpasses GCN performance but also competes with other, more sophisticated solutions involving recurrent neural networks. Moreover, the temporal graphical convolution architecture also improves in every scenario with the addition of spatial attention.

In order to provide statistical evidence of the results obtained some non-parametric tests have been performed. Firstly, a Friedman rank test has been applied to determine if there exists any different performance error between the GCN, GAT, GRU, T-GCN and T-GAT models among all datasets and forecasting horizons. A Friedman statistic of $F = 41.6$ for RMSE and $F = 35.4$ for MAE has been obtained, resulting on a $P_{value} = 2.02 \cdot 10^{-8}$ and $P_{value} = 3.84 \cdot 10^{-7}$ respectively. These values are much smaller than the usual threshold $\alpha = 0.05$, meaning there is statistical evidence that there exists different performances between at least two models of the collection.

Given that Friedman's null hypothesis was rejected, two post-hoc pairwise non-parametric-based comparison were carried out to check the differences between the proposed T-GAT model and the baselines. These procedures are the Conover [47] and the Nemenyi [48] post-hoc tests. Statistical significance for $\alpha = 0.05$ is achieved in every scenario, with the exception of the Nemenyi test against T-GCN. This is due to the fact that these non-parametric tests work with the rank value, so the more models that are tested the more difficult it is to obtain statistical significance. When GCN and GRU are dropped from the test, the null hypothesis is rejected in every scenario for both tests.

Another testing approach is to perform the Wilcoxon test [49] for individual comparison between models, which is the non-parametric equivalent of the paired t-test. With this technique, statistical significance below $\alpha = 0.05$ is achieved against every model. All hypothesis test values can be seen in Table IV.

After applying different non-parametric tests it can be concluded that the addition of spatial attention to the original T-GCN network leads to a performance improvement with more than 95% of confidence.

VI. Conclusions

Through this work, the impact of the addition of a spatial attention mechanism to a deep learning model designed to forecast traffic intensity has been measured. The resulting T-GAT model outperforms a wide set of preexisting state-of-the-art models in all tested scenarios with a great degree of confidence. This evidence suggests that the general use of attention mechanisms could be beneficial in the traffic forecasting problem.

Future work will be oriented to further refinements of the inclusion of attention mechanisms, both in space and in time, which will allow to improve predictions over longer horizons.

VII. Acknowledgment

This research was partially funded by the Empresa Municipal de Transportes (EMT) of Madrid under the program "Aula Universitaria EMT/UNED de Calidad del Aire y Movilidad Sostenible".

References

[1] A. Sperduti and A. Starita, "Supervised neural networks for the classification of structures," *IEEE Transactions on Neural Networks*, vol. 8, no. 3, pp. 714–735, 1997.
[2] R. de Medrano and J. L. Aznarte, "On the inclusion of spatial information for spatio-temporal neural networks," *Neural Comput. Appl.*, vol. 33, no. 21, pp. 14 723–14 740, 2021. [Online]. Available: https://doi.org/10.1007/s00521-021-06111-6
[3] L. Zhao, Y. Song, C. Zhang, Y. Liu, P. Wang, T. Lin, M. Deng, and H. Li, "T-gcn: A temporal graph convolutional network for traffic prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 9, pp. 3848–3858, 2019.

[1] $lr = 0.0002$ and Weight decay regularization $wd = 10^{-6}$

TABLE I
RESULTS OF ALL CONSIDERED MODELS FOR THE SHENZHEN DATASET.

T	Metric	Shenzhen Dataset						
		HA	SVR	GCN	GAT	GRU	T-GCN	T-GAT
15 min	RMSE	4.2951	4.1301	5.3480 ± 0.0808	4.2544 ± 0.0492	4.2490 ± 0.1042	4.0809 ± 0.0397	**4.0402 ± 0.0877**
	MAE	2.7815	2.7200	3.9134 ± 0.0425	2.8873 ± 0.0372	2.8099 ± 0.0598	2.7419 ± 0.0262	**2.6673 ± 0.0594**
30 min	RMSE	4.3740	4.1670	5.3757 ± 0.0770	4.3142 ± 0.0620	4.2772 ± 0.1159	4.0924 ± 0.0475	**4.0306 ± 0.0669**
	MAE	2.8309	2.7883	3.9351 ± 0.0503	2.9567 ± 0.0344	2.8260 ± 0.0670	2.7514 ± 0.0302	**2.6642 ± 0.0361**
45 min	RMSE	4.4393	4.2000	5.4101 ± 0.0851	4.3543 ± 0.0518	4.1649 ± 0.0762	4.1118 ± 0.0555	**4.0374 ± 0.0831**
	MAE	2.8723	2.8324	3.9592 ± 0.0511	3.0054 ± 0.0250	2.8374 ± 0.0508	2.7863 ± 0.0403	**2.6708 ± 0.0488**
60 min	RMSE	4.4917	4.2300	5.4261 ± 0.0761	4.3846 ± 0.0788	4.2731 ± 0.0901	4.1250 ± 0.0510	**4.0685 ± 0.0911**
	MAE	2.9056	2.8654	3.9731 ± 0.0535	2.9892 ± 0.0488	2.9568 ± 0.0572	2.7761 ± 0.0357	**2.6968 ± 0.0532**

TABLE II
RESULTS OF ALL CONSIDERED MODELS FOR THE LOSLOOP DATASET.

T	Metric	Losloop Dataset						
		HA	SVR	GCN	GAT	GRU	T-GCN	T-GAT
15 min	RMSE	7.3067	5.3686	8.9705 ± 0.4020	5.0170 ± 0.1750	6.1178 ± 0.0871	5.1856 ± 0.1629	**4.9699 ± 0.1650**
	MAE	3.8782	2.9861	6.6155 ± 0.2997	2.9906 ± 0.0988	3.6245 ± 0.0569	3.3036 ± 0.0960	**2.9435 ± 0.0958**
30 min	RMSE	7.9575	6.4302	9.3767 ± 0.2416	5.9831 ± 0.2718	7.1445 ± 0.3377	6.2011 ± 0.2669	**5.9786 ± 0.3848**
	MAE	4.1699	3.4439	6.8963 ± 0.2120	**3.3891 ± 0.1315**	4.2791 ± 0.2166	3.8510 ± 0.1745	3.4696 ± 0.2399
45 min	RMSE	8.5986	7.2640	9.7082 ± 0.1359	6.6667 ± 0.2530	7.6506 ± 0.1098	6.9785 ± 0.2088	**6.6363 ± 0.2415**
	MAE	4.4824	3.8465	6.9968 ± 0.1456	**3.7295 ± 0.1312**	4.4183 ± 0.0994	4.4304 ± 0.1720	4.0514 ± 0.1572
60 min	RMSE	9.2619	7.9821	9.9953 ± 0.3105	7.3156 ± 0.3142	8.0614 ± 0.3417	7.7854 ± 0.2636	**7.2232 ± 0.4502**
	MAE	4.8280	4.2274	6.5963 ± 0.2111	**4.1465 ± 0.1877**	4.7718 ± 0.2310	4.8024 ± 0.1488	4.3509 ± 0.1982

TABLE III
RESULTS OF ALL CONSIDERED MODELS FOR THE M30 DATASET.

T	Metric	M30 Dataset						
		HA	SVR	GCN	GAT	GRU	T-GCN	T-GAT
15 min	RMSE	9.2291	8.1580	12.6491 ± 0.5986	8.1384 ± 0.5003	8.0407 ± 0.4687	7.6845 ± 0.4076	**7.6057 ± 0.3867**
	MAE	5.1680	4.5552	9.1352 ± 0.4207	4.6469 ± 0.3591	4.7430 ± 0.2959	4.6146 ± 0.3019	**4.6140 ± 0.2681**
30 min	RMSE	9.8126	8.8255	12.9819 ± 0.3912	8.8680 ± 0.4466	8.5089 ± 0.4205	8.2679 ± 0.2173	**7.9930 ± 0.3796**
	MAE	5.4684	4.9414	9.3617 ± 0.2679	5.1212 ± 0.3083	5.2424 ± 0.3031	5.2294 ± 0.1818	**4.8255 ± 0.2741**
45 min	RMSE	10.3915	9.4176	13.4071 ± 0.3957	9.5175 ± 0.5790	8.8205 ± 0.5129	8.4113 ± 0.5337	**8.2102 ± 0.3617**
	MAE	5.7741	5.2928	9.6616 ± 0.2581	5.6425 ± 0.4167	5.4609 ± 0.3555	5.2247 ± 0.3823	**4.9142 ± 0.2359**
60 min	RMSE	10.9869	9.9505	13.7151 ± 0.6154	9.9975 ± 0.4549	9.0805 ± 0.4325	8.7436 ± 0.4203	**8.3670 ± 0.4057**
	MAE	6.0984	5.6238	9.8641 ± 0.4275	5.9363 ± 0.3729	5.6070 ± 0.3131	5.4944 ± 0.3039	**5.0862 ± 0.2668**

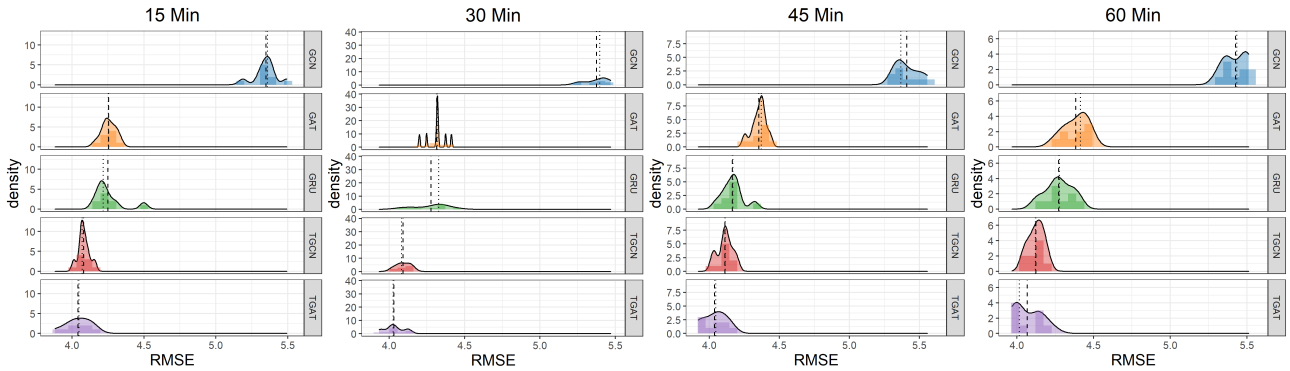

Fig. 8. RMSE distribution of all considered models for the Shenzen dataset.

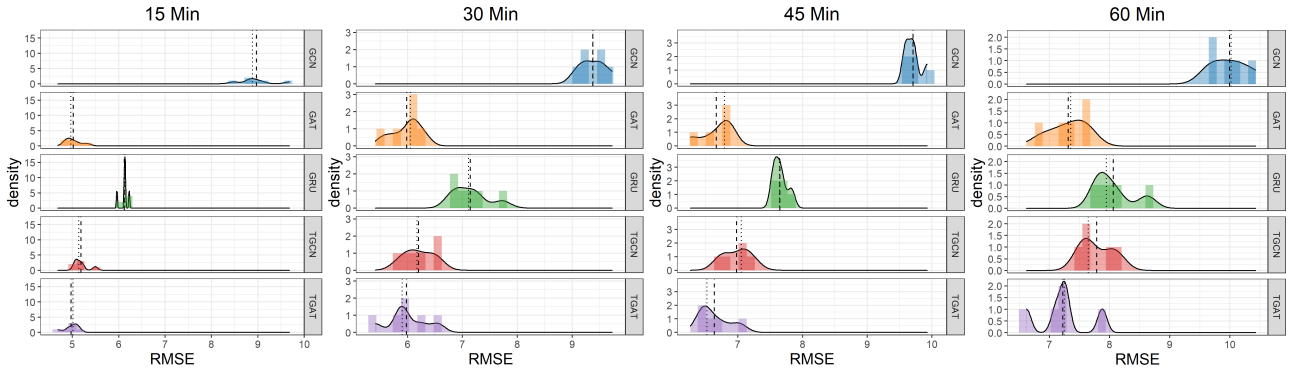

Fig. 9. RMSE distribution of all considered models for the Losloop dataset.

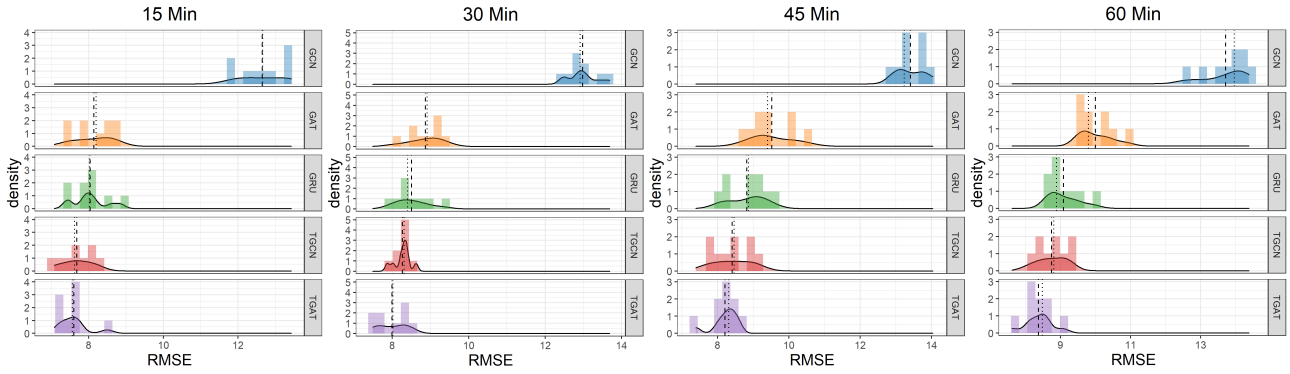

Fig. 10. RMSE distribution of all considered models for the M30 dataset.

TABLE IV
STATISTICAL SIGNIFICANCE OF THE RESULTS

Test	Model	Metric	P-value
Conover	GCN	RMSE	0
Conover	GAT	RMSE	0.0009
Conover	GRU	RMSE	0.0009
Conover	T-GCN	RMSE	0.0485
Conover	GCN	MAE	0
Conover	GAT	MAE	0.0202
Conover	GRU	MAE	0.0028
Conover	T-GCN	MAE	0.0485
Nemenyi	GCN	RMSE	0.001
Nemenyi	GAT	RMSE	0.0028
Nemenyi	GRU	RMSE	0.0028
Nemenyi	T-GCN	RMSE	0.2352
Nemenyi	GCN	MAE	0.001
Nemenyi	GAT	MAE	0.0028
Nemenyi	GRU	MAE	0.0028
Nemenyi	T-GCN	MAE	0.2352
Wilcoxon	GCN	RMSE	0.0002
Wilcoxon	GAT	RMSE	0.0002
Wilcoxon	GRU	RMSE	0.0002
Wilcoxon	T-GCN	RMSE	0.0002
Wilcoxon	GCN	MAE	0.0002
Wilcoxon	GAT	MAE	0.0386
Wilcoxon	GRU	MAE	0.0002
Wilcoxon	T-GCN	MAE	0.0002

[4] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[5] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[6] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *arXiv preprint arXiv:1710.10903*, 2017.

[7] E. I. Vlahogianni, M. G. Karlaftis, and J. C. Golias, "Short-term traffic forecasting: Where we are and where we're going," *Transportation Research Part C: Emerging Technologies*, vol. 43, pp. 3–19, 2014.

[8] M. S. Ahmed and A. R. Cook, *Analysis of freeway traffic time-series data by using Box-Jenkins techniques*, 1979, no. 722.

[9] H. Sun, C. Zhang, and B. Ran, "Interval prediction for traffic time series using local linear predictor," in *Proceedings. The 7th International IEEE Conference on Intelligent Transportation Systems (IEEE Cat. No. 04TH8749)*. IEEE, 2004, pp. 410–415.

[10] I. Okutani and Y. J. Stephanedes, "Dynamic prediction of traffic volume through kalman filtering theory," *Transportation Research Part B: Methodological*, vol. 18, no. 1, pp. 1–11, 1984.

[11] M. Van Der Voort, M. Dougherty, and S. Watson, "Combining kohonen maps with arima time series models to forecast traffic flow," *Transportation Research Part C: Emerging Technologies*, vol. 4, no. 5, pp. 307–318, 1996.

[12] S. Lee and D. B. Fambro, "Application of subset autoregressive integrated moving average model for short-term freeway traffic volume forecasting," *Transportation Research Record*, vol. 1678, no. 1, pp. 179–188, 1999.

[13] B. M. Williams and L. A. Hoel, "Modeling and forecasting vehicular traffic flow as a seasonal arima process: Theoretical basis and empirical results," *Journal of transportation engineering*, vol. 129, no. 6, pp. 664–672, 2003.

[14] N. S. Altman, "An introduction to kernel and nearest-neighbor nonparametric regression," *The American Statistician*, vol. 46, no. 3, pp. 175–185, 1992.

[15] C.-H. Wu, J.-M. Ho, and D.-T. Lee, "Travel-time prediction with sup-

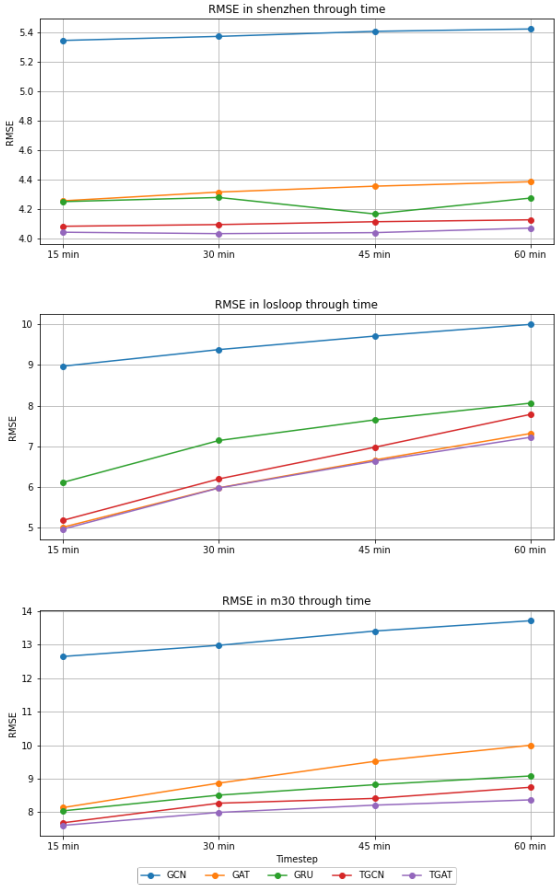

Fig. 11. Distribution of the RMSE for different forecasting horizons.

port vector regression," *IEEE transactions on intelligent transportation systems*, vol. 5, no. 4, pp. 276–281, 2004.

[16] H. Yin, S. Wong, J. Xu, and C. Wong, "Urban traffic flow prediction using a fuzzy-neural approach," *Transportation Research Part C: Emerging Technologies*, vol. 10, no. 2, pp. 85–98, 2002.

[17] S. Sun, C. Zhang, and G. Yu, "A bayesian network approach to traffic flow forecasting," *IEEE Transactions on intelligent transportation systems*, vol. 7, no. 1, pp. 124–132, 2006.

[18] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[19] R. Fu, Z. Zhang, and L. Li, "Using lstm and gru neural network methods for traffic flow prediction," in *2016 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC)*. IEEE, 2016, pp. 324–328.

[20] Y. Wu and H. Tan, "Short-term traffic flow forecasting with spatial-temporal correlation in a hybrid deep learning framework," *arXiv preprint arXiv:1612.01022*, 2016.

[21] R. de Medrano and J. L. Aznarte, "A spatio-temporal attention-based spot-forecasting framework for urban traffic prediction," *Applied Soft Computing*, vol. 96, p. 106615, 2020.

[22] X. Ma, Z. Dai, Z. He, J. Ma, Y. Wang, and Y. Wang, "Learning traffic as images: a deep convolutional neural network for large-scale transportation network speed prediction," *Sensors*, vol. 17, no. 4, p. 818, 2017.

[23] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, "A comprehensive survey on graph neural networks," *IEEE transactions on neural networks and learning systems*, vol. 32, no. 1, pp. 4–24, 2020.

[24] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun, "Graph neural networks: A review of methods and applications," *AI Open*, vol. 1, pp. 57–81, 2020.

[25] Y. Li, D. Tarlow, M. Brockschmidt, and R. Zemel, "Gated graph sequence neural networks," *arXiv preprint arXiv:1511.05493*, 2015.

[26] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," *Advances in neural information processing systems*, vol. 29, pp. 3844–3852, 2016.

[27] R. Li, S. Wang, F. Zhu, and J. Huang, "Adaptive graph convolutional neural networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[28] C. Zhuang and Q. Ma, "Dual graph convolutional networks for graph-based semi-supervised classification," in *Proceedings of the 2018 World Wide Web Conference*, 2018, pp. 499–508.

[29] A. Micheli, "Neural network for graphs: A contextual constructive approach," *IEEE Transactions on Neural Networks*, vol. 20, no. 3, pp. 498–511, 2009.

[30] D. Bacciu, F. Errica, and A. Micheli, "Contextual graph markov model: A deep and generative approach to graph processing," in *International Conference on Machine Learning*. PMLR, 2018, pp. 294–303.

[31] J. Atwood and D. Towsley, "Diffusion-convolutional neural networks," in *Advances in neural information processing systems*, 2016, pp. 1993–2001.

[32] T. Hamaguchi, H. Oiwa, M. Shimbo, and Y. Matsumoto, "Knowledge transfer for out-of-knowledge-base entities: A graph neural network approach," *arXiv preprint arXiv:1706.05674*, 2017.

[33] S. Brody, U. Alon, and E. Yahav, "How attentive are graph attention networks?" *arXiv preprint arXiv:2105.14491*, 2021.

[34] J. Zhang, X. Shi, J. Xie, H. Ma, I. King, and D.-Y. Yeung, "Gaan: Gated attention networks for learning on large and spatiotemporal graphs," *arXiv preprint arXiv:1803.07294*, 2018.

[35] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," *arXiv preprint arXiv:1707.01926*, 2017.

[36] J. Bai, J. Zhu, Y. Song, L. Zhao, Z. Hou, R. Du, and H. Li, "A3t-gcn: Attention temporal graph convolutional network for traffic forecasting," *ISPRS International Journal of Geo-Information*, vol. 10, no. 7, p. 485, 2021.

[37] C. Song, Y. Lin, S. Guo, and H. Wan, "Spatial-temporal synchronous graph convolutional networks: A new framework for spatial-temporal network data forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 01, 2020, pp. 914–921.

[38] C. Zheng, X. Fan, C. Wang, and J. Qi, "Gman: A graph multi-attention network for traffic prediction," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 01, 2020, pp. 1234–1241.

[39] S. Guo, Y. Lin, N. Feng, C. Song, and H. Wan, "Attention based spatial-temporal graph convolutional networks for traffic flow forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 922–929.

[40] Z. Wu, S. Pan, G. Long, J. Jiang, and C. Zhang, "Graph wavenet for deep spatial-temporal graph modeling," *arXiv preprint arXiv:1906.00121*, 2019.

[41] M. M. Bronstein, J. Bruna, T. Cohen, and P. Veličković, "Geometric deep learning: Grids, groups, graphs, geodesics, and gauges," *arXiv preprint arXiv:2104.13478*, 2021.

[42] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," *arXiv preprint arXiv:1312.6203*, 2013.

[43] Municipality of madrid portal. [Online]. Available: https://datos.madrid.es/portal/site/egob/

[44] OpenStreetMap contributors, "Planet dump retrieved from https://planet.osm.org ," https://www.openstreetmap.org , 2017.

[45] S. J. Taylor and B. Letham, "Forecasting at scale," *The American Statistician*, vol. 72, no. 1, pp. 37–45, 2018.

[46] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE signal processing magazine*, vol. 30, no. 3, pp. 83–98, 2013.

[47] W. J. Conover and R. L. Iman, "Multiple-comparisons procedures. informal report," Los Alamos National Lab.(LANL), Los Alamos, NM (United States), Tech. Rep., 1979.

[48] P. B. Nemenyi, *Distribution-free multiple comparisons*. Princeton University, 1963.

[49] F. Wilcoxon, "Individual comparisons by ranking methods," in *Breakthroughs in statistics*. Springer, 1992, pp. 196–202.