Expert Systems With Applications 228 (2023) 120281

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Spatio-temporal graph mixformer for traffic forecasting

Mourad Lablack, Yanming Shen [*]

School of Electronic Information and Electrical Engineering, Dalian University of Technology, Dalian, Liaoning 116024, China

ARTICLE INFO

Keywords:
Traffic forecasting
Spatio-temporal multivariate time series
Graph neural network
Attention mechanism

ABSTRACT

Traffic forecasting is of great importance for intelligent transportation systems (ITS). Because of the intricacy implied in traffic behavior and the non-Euclidean nature of traffic data, it is challenging to give an accurate traffic prediction. Despite that previous studies considered the relationship between different nodes, the majority have relied on a static representation and failed to capture the dynamic node interactions over time. Additionally, prior studies employed RNN-based models to capture the temporal dependency. While RNNs are a popular choice for forecasting problems, they tend to be memory hungry and slow to train. Furthermore, recent studies start utilizing similarity algorithms to better express the implication of a node over the other. However, to our knowledge, none have explored the contribution of node i's past, over the future state of node j. In this paper, we propose a Spatio-Temporal Graph Mixformer (STGM) network, a highly optimized model with low memory footprint. We address the aforementioned limits by utilizing a novel attention mechanism to capture the correlation between temporal and spatial dependencies. Specifically, we use convolution layers with a variable fields of view for each head to capture long–short term temporal dependency. Additionally, we train an estimator model that express the contribution of a node over the desired prediction. The estimation is fed alongside a distance matrix to the attention mechanism. Meanwhile, we use a gated mechanism and a mixer layer to further select and incorporate the different perspectives. Extensive experiments show that the proposed model enjoys a performance gain compared to the baselines while maintaining the lowest parameter counts.

1. Introduction

Traffic forecasting aim to predict the future traffic behavior based on historical data, which is a spatio-temporal problem with complex spatial, temporal, and external dependencies (Chen et al., 2019). First in the spatial dependency, any node is more or less affected by its directly connected neighboring nodes, e.g., the distance between two nodes is a good indicator. Second in the temporal dependency, the traffic shows three main patterns, i.e., short-term, midterm, and long-term. Finally, there exist some external dependencies such as weather conditions, special events, and so on.

In the last decade, researchers start using neural network-based models to solve traffic forecasting problems. Liu, Zhen, Li, Zhan, and Lin (2019), Mourad, Qi, Shen, and Yin (2019) and Zhang, Zheng, and Qi (2017) consider the problem as an image prediction. The traffic records are first arranged into a grid map like an image, and each area contains the average of all the signals collected from stations that belong to the targeted area, the grids are then stacked together to represent different traffic signals such as speed, flow, and occupancy. Furthermore, the grids are processed using convolution layers to capture the spatial dependency. These techniques have improved considerably the prediction accuracy compared to the classical methods. However, CNNs and its variant are mainly good at extracting features in a Euclidean space and fail to capture the spatial features of the underline graph structure. The grid grouping also results in a loss of granularity and structural integrity of the road network.

More recent works have attempted to solve the problem in a non-Euclidean space by using graph neural networks, which are basically an extension to neural networks that take in consideration the graph structure. Therefore, instead of generating grids, (Chen et al., 2019; Zhao et al., 2020; Zhu, Song, Zhao, & Li, 2020) feed the traffic signals directly to the model with the adjacency matrix describing the links and distances between two nodes. On the other hand, (Bai, Yao, Li, Wang, & Wang, 2020; Wu et al., 2020) generate the graph representation from node embedding learned during training. However, these approaches result in a static graph representation used to propagate the information along neighbors. Traffic presents dynamic changes in the spatial behavior over time, thus a static representation is not well suited for capturing the spatial dynamics.

Concerning the temporal dependency, recurrent neural networks (RNNs) are widely used in the aforementioned researches. RNNs are

[*] Corresponding author.
E-mail addresses: mouradost@mail.dlut.edu.cn (M. Lablack), shen@dlut.edu.cn (Y. Shen).

https://doi.org/10.1016/j.eswa.2023.120281
Received 22 November 2022; Received in revised form 24 April 2023; Accepted 24 April 2023
Available online 29 April 2023

good for short-term dependency capture but sufferer from memory saturation in the long-term. Zhu et al. (2020) tried to alleviate this limitation by introducing attention mechanism. Although attention extends the temporal capture range of RNNs, they still suffer from the same limitation. Mourad et al. (2019) and Zhang et al. (2017) group the traffic data into time clusters, namely trend, period and closeness to capture the different cycles relative to traffic behavior, and therefore can capture long-range time changes. However, each group is handled by a different branch which increases the complexity and size of the model.

Motivated by the above limitations, we propose a Spatial–Temporal Graph Mixformer (STGM) architecture traffic forecasting. As described earlier, the traffic signal is affected by many external factors, and we observe an exhibition of hourly, daily and weekly periodicity while the sensors used to monitor the overall traffic are distinguished by a spatial location and the number of connections with other areas which we call centrality. As highlighted in Shao, Zhang, Wang, Wei, and Xu (2022), a simple network can successfully provide accurate predictions when given proper differentiable characteristics. In STGM, we explicitly capture the various characteristics with a series of data driven randomly initialized embedding that gradually constructs the spatial–temporal identities.

Traffic signals display long range periodic correlations that are acquired only with access to long temporal sequences. However, long sequences are inhibited in large graph manipulations given the physical memory constraints of the underline hardware. To overcome this limitation, we propose the similarity estimator model which can without the traditionally expensive computations approximate the broad view of long temporal correlations.

Furthermore, we propose improvements to the transformer architecture (Vaswani et al., 2017) by introducing temporal dilated causal convolution with variable fields of view that empowers each head with different temporal perspectives, and incorporate the global contribution from the estimator model alongside the distance matrix as biases to the attention score.

Finally, while local correlation can be easily captured by the dynamic message passing mechanism of STGA, the global correlation is only captured with multiple stacked STGA blocks which is inefficient, thus we propose to endow the graph with a super-node that is connected to all nodes which provides a bird eye view of the entire traffic behavior. Additionally, we design the CT-Mixer block - a simple yet effective approach to merge the encoded historical information and generate the features describing the future traffic behavior. The proposed CT-Mixer produces the desired features by applying a shared fully connected layer on each channel belonging to the same temporal group and repeating the same process along each feature for all the timestamps. Therefore, capturing the non-sequential temporal interaction and the underline inner-region correlation.

The contributions of this paper can be summarized as follows:

- We suggest a similarity estimator model that given a set of nodes with a segment of historical traffic, approximates the expected implication of each node over the future traffic signal.
- Furthermore, we designed a couple of improvements for the original transformer architecture named STGA that fully utilizes the multi-head attention and seamlessly capture both temporal and spatial dependencies.
- We propose the CT-Mixer module that works in coordination with STGA to further capture the temporal locality and internode channels information.
- Finally, we extensively experimented our model on three well-known real-world traffic datasets to verify the effectiveness of our architecture. The results show that STGM outperformed the baselines despite being one of the smallest models in terms of the number of parameters.

2. Related work

2.1. Spatio-temporal traffic forecasting

Early studies considered traffic forecasting as a simple time-series prediction problem such as Ahn, Eunjeong Ko, and Eun Yi Kim (2016), Lin (2016), Liu, Du, Yan, Chai, and Guo (2018), Mai, Ghosh, and Wilson (2014), Zhang, Liu, Yang, Wei, and Dong (2013) ARIMA, SVM, etc. Since these methods only infer the temporal dependency, they tend to give shallow predictions.

More recent studies consider both spatial and temporal dependencies with a grid like multivariate time series structure, and thanks to the advancement in deep learning methods, more capable and entangled models have been proposed such as Liang et al. (2019), Liao et al. (2018), Liu et al. (2019) and Zhang et al. (2017). By using stacks of RNNs and CNNs layers they have greatly improved the prediction accuracy. Mourad et al. (2019) and Yuan, Zhou, and Yang (2018) introduced convLSTM layers, a tight integration of CNNs into RNNs layers for a seamless capture of both dependencies.

More recently, the research community has shifted into using graph structures to describe traffic data in the non-Euclidean space. DCRNN (Li, Yu, Shahabi, & Liu, 2018) proposed to model the interaction between different segment of the road network as a diffusion process, they generate diffusion matrices from a random walk which approximate the spatial complexity of traffic, furthermore the integration of this process into GRU layer helps capture the temporal dependency. GraphSAGE (Hamilton, Ying, & Leskovec, 2018) enabled the use of convolution in graphs by proposing to simple a node state by aggregating the neighboring information which enabled (Zhao et al., 2020) to use GCNs and further improved the capture of spacial correlation. While (Li et al., 2021) generates dynamic graph representation for each time-step based on a series of node embedding joined with the hidden cell of a GRU layer, and the authors further combine the static distance matrix with each dynamic graph representation with a weighted sum. Shao, Zhang, Wei, et al. (2022) highlights the existence of two kinds of traffic signals, namely diffusion and inherent signals, and the authors proposed a decoupling system where the model first learn and filter the diffusion signal then the inherent block captures the remaining information. Deng, Chen, Jiang, Song, and Tsang (2021) proposes to enhance a gated dilated causal convolution with temporal and spatial normalization that helps better distinguish the high-frequency component and the local component underlying the raw data.

2.2. Attention mechanism

The attention mechanism came from the need of a more localized learning and surged in the language processing field, which was made popular with the work Vaswani et al. (2017). It soon was utilized in traffic forecasting to capture the temporal dependency. Zhu et al. (2020) used attention on the hidden cells to assign different weights to each time-step and improved considerably over previous work (Zhao et al., 2020). Chen et al. (2019) proposed a similar variant where in addition of using the attention on the cells, they also used it on a gated mechanism to control the residual connection. Sun, Zhao, Shi, and He (2021) and Yin et al. (2021) further improved the architecture by adopting an encoder–decoder style while using an attention mechanism on the transition phase from the encoder to the decoder, Shao, Zhang, Wang, and Xu (2022) learn long sequences by first pre-training a transformer based encoder using imputation technique applied to a patched sequence then using the decoder incorporate the long-range temporal context with the last temporal patch to provide a prediction with any arbitrary model. Other studies focused on the spacial dependency, by using attention in the message passing step to highlight the prominent neighbors for the targeted node, Veličković et al. (2018) proposed GAT and was soon applied to traffic forecasting by Wei and Sheng (2020) and Zhang et al. (2018) since it better suits the need for a dynamic spacial dependency characterized by traffic data. Bai et al. (2020) uses a different approach, instead of using the original adjacency matrix and modulating its weights with attention, they used an adaptation matrix that is fully learned with the input signals during training, this is a more flexible approach since the generated matrix is not bound to the graph physical structure. Similarly, Zhang et al. (2021) proposed a two level attentive graph propagation to express both local and global spatial correlation while capturing the temporal hierarchy with a multi-head attention. Given that the aforementioned studies heavily rely on multiple attention heads to capture different aspects of the data, we argue that to fully utilize the multiple perspectives of each head, the underline implementation should incorporate different field of views.

2.3. CNNs as an alternative to RNNs

Attention mechanisms have improved considerably previous architectures to better capture the Spatio-temporal dependencies. Shi, Qi, Shen, Wu, and Yin (2021) ensures the capture of daily and weekly periodicity with an attention mechanism hooked to the hidden cell of a temporal skip LSTM combined with a spatial attention. However, RNN and its variant are slow to train and sometimes fail to retain information in the long term despite being enhanced by attention mechanisms. Dilation convolution layers can be viewed as an alternative to RNNs, and they usually provide more robust results for long-range time dependency. This explains the shift observed in the research community from RNNs to CNNs based architectures, Kong et al. (2020) uses a gated temporal convolution network with different dilation factors to capture the long-range time dependency followed by a GAT layer for the spatial aspect, while Yu, Yin, and Zhu (2018) arrange two temporal convolution layers in a gated way that sandwich a GCN layer, Diao et al. (2019) improves it by adding an adjacency estimator that is fed to the GCN layer as a dynamic graph representation, Xie, Xiong, and Zhu (2020) adopt a transformer architecture to process spatial dependency. Similarly, Lan, Ma, Huang, Wang, Yang, and Li (2022) proposed a spatial attention based on Chebyshev polynomials to extract the dynamic spatial correlation and a multi-receptive field dynamic temporal gated convolution for long range capture, Fang, Long, Song, and Xie (2021) proposed an ODE based tensor coupled with dilation convolution to capture complex spatial semantic connection and long-term temporal dependency. However, Most previously cited studies achieves long-range temporal capture by utilizing a dilatation that grows with each layer, this means that the model ability to capture long-range is tightly related to the depth of the network, this become troublesome when working on large graph structure since the memory and computation needed for the message passing phase limits the possible depth of the network, thus limiting the temporal capture range of the network. In addition, most of the previous architectures rely on sequential stacking of CNN and underutilize the parallel computing enabled by modern GPUs training.

2.4. Similarity learning

Traffic behavior exposes a ubiquity of contemporaneous similarities between different nodes. Learning the similarity between each region and understanding the causality is of great importance for an accurate traffic prediction. Similarity and causality learning has long been a hot topic in studying multivariate time series and a plenty of powerful algorithms have been proposed. In the context of causality, Granger causality (Granger, 1969) is a well-known method, which is used in terms of predictability where X Granger causes Y if the historical data of X better predict Y than Y own past (Arnold, Liu, & Abe, 2007). The Granger causality assumes linear and stationary time series which becomes a big limitation for many real-world applications. Transfer Entropy was proposed as a non-linear extension to Granger causality (Barnett, Barrett, & Seth, 2009). While in the context of similarity, Dynamic Time Warping (DTW) (Bellman & Kalaba, 1959) is the prominent algorithm in the literature, initially used in speech recognition, it is currently used in many areas especially in traffic forecasting, mostly as an alternative or addition to the graph representation. Yu, Li, Zhang, and Zhu (2019) proposed a data driven graph representation based on similarity between nodes to expose the distant interaction in the traffic network that usually cannot be exposed with distance based adjacent matrices. Alternatively, DTW has successfully been used as a replacement loss function to Mean Squared Error in time series forecasting (Guen & Thome, 2019) and shows promising results.

3. Preliminaries

3.1. Traffic network

We define the traffic network as a directed graph $\mathbb{G} = (V, A; X^t)$, where V represents the vertices such as $|V| = N$. Let $(v_i, v_j) \in V$ be two adjacent vertices linked by a road segment. The weighted adjacency matrix $A \in \mathbb{R}^{N\times N}$ is then normalized using the equation described in Eq. (1), where σ is the standard deviation and $dist(i, j)$ is the distance between node i and node j. The exponential normalization generate values close to zero for the distances exceeding the standard deviation, a gradual separation for mid-range distances and high values for short distances. Finally, the values are kept between 0 and 1.

$$A_{i,j} = \begin{cases} \exp\left(-\left(\frac{\mathrm{dist}(i,j)}{\sigma}\right)^2\right) & \text{if } \mathrm{dist}(i,j) \neq 0 \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

$X^t \in \mathbb{R}^{N\times F}$ represents the signal collected from each vertex at a given time t. A signal can contain one or more information represented by F (e.g., volume, speed, occupancy, etc.).

3.2. Contribution learning

Let $n_i^H \in \mathbb{R}^H$ be the record of node i for the past H time steps, and $n_j^F \in \mathbb{R}^F$ be the records of node j for the future F time steps. We assume that any node n_i may contribute to any other node n_j regardless of the existence of an edge $e_{i,j}$. We define P as the set of all possible causal paths between n_i^H and n_j^F. Then, the total cost of a causal path can be determined by Eq. (2) where c_p is the total cost $c(n_i^h, n_j^f)$ assuming that $H = F$. The expected contribution of node n_i^H over n_j^F can then be calculated by Eq. (3). The total expected contribution $\mathbb{E}$ of the historical data over the future state of each node is described in Eq. (4).

$$c_p(n_i^H, n_j^F) = \sum c(n_i^h, n_j^f) \tag{2}$$

$$\xi_{i,j} = \exp\left(-\left(\frac{\mathrm{c}(n_i^H, n_j^F)}{\sigma}\right)^2\right) \tag{3}$$

$$\mathbb{E} = \begin{bmatrix} \xi_{1,1} & \cdots & \xi_{1,N} \\ \vdots & & \vdots \\ \cdots & \xi_{i,j} & \cdots \\ \vdots & & \vdots \\ \xi_{N,1} & \cdots & \xi_{N,N} \end{bmatrix} \tag{4}$$

3.3. Problem formulation

The traffic prediction can be formulated as follows, given H historical data $X = (X^1, X^2, \cdots, X^H)$, $X \in \mathbb{R}^{H\times N\times F}$, predict T future traffic data $Y = (Y^{H+1}, Y^{H+2}, \cdots, Y^{H+T})$, $Y \in \mathbb{R}^{T\times N\times F'}$, where F' is the targeted predicted features.

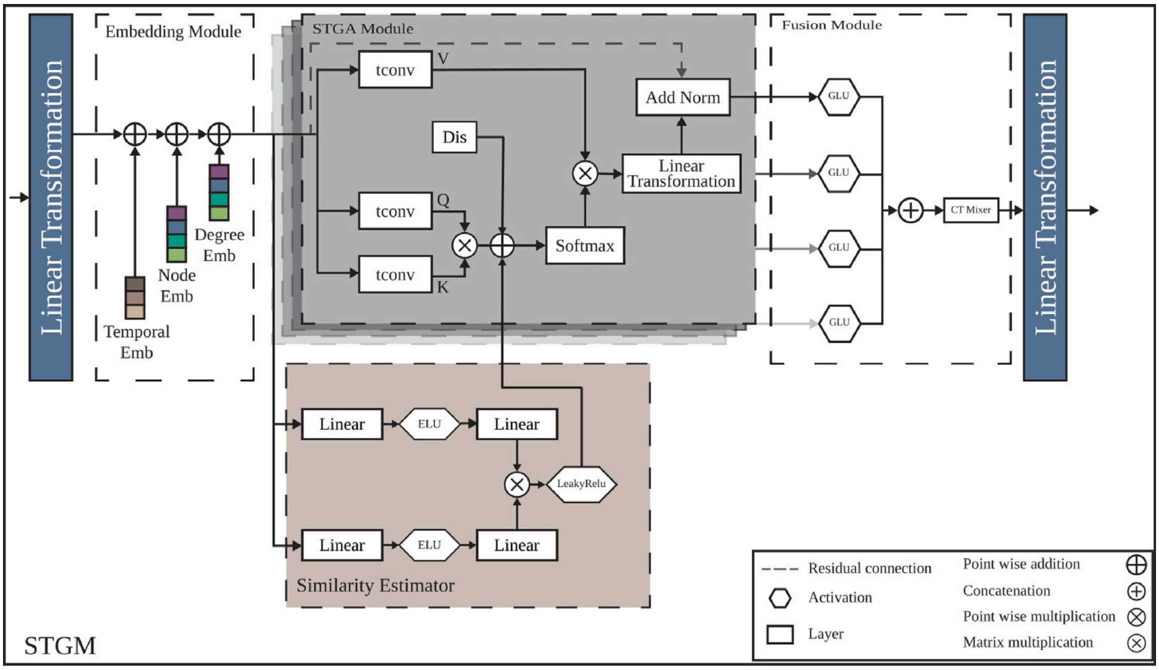

Fig. 1. The proposed STGM Architecture including the Estimator, Embedding, multi-heads STGA, and Fusion modules.

4. Proposed method

In this section, we introduce the Spatio-Temporal Graph Mixformer (STGM) framework for traffic forecasting shown in Fig. 1. We start with an overview of the entire architecture and explain how the information is processed through the network, and then we describe each component and their interaction.

4.1. Overview

We design three main modules i.e., embedding module, STGM module, and fusion module. The embedding module is used to incorporate the multiple additional information relative to the final prediction, the STGA module is used to generate a spatio-temporal score that represents a message-passing policy for each time-step. Finally, we use a fusion module that contains multiple gates to further restrict and regulate the amount of relative information used in the mixer layer. The mixer integrate all the information learned in each head and unifies it into a lower latent space. Alongside the main model we train a sub-model in a separate loop to estimate the expected contribution for each node toward the future state of the graph.

STGM first applies a linear transformation described in Eq. (5), which transforms each node features (F) to a higher latent space, and the transformed input is then fed to an embedding module that adds proxy information as described in Section 4.2. At this point the data is shared across the two sub-models, where a copy is used in the estimator model to generate the contribution matrix as described in Section 4.2, and the other copy is fed to four parallel STGA Modules described in Section 4.3. The generated contribution matrix alongside the distance matrix are used in each head to modulate the dynamic graph representation. STGA propagate each node information based on the dynamic graph representation. The output of each head is then fed to the fusion module that select and merge all results into a single output, additionally we add a residual connection coming from the embedding module, and we apply an L2 normalization. This process is repeated (N) times as shown in Fig. 1. Finally, the output is fed to a linear transformation to produce the desired prediction features (F') with a temporal sequence length (T).

$$h_t = \sigma(X_t W_{in} + B_{in}) \tag{5}$$

Where X_t is the input, h_t is the output of the input layer, σ is a non-linear activation and $W_{in} \in \mathbb{R}^{F\times C}$, $B_{in} \in \mathbb{R}^{1\times C}$ are learnable parameters. F is the feature size of each node and $C = F \times \omega$.

$$Y_T = \sigma(h'_t W_{out} + B_{out}) \tag{6}$$

h'_t is the output from the previous layers, Y_T is the prediction with a sequence length T, σ is a non-linear activation and $W_{out} \in \mathbb{R}^{C\times F'}$, $B_{out} \in \mathbb{R}^{1\times F'}$ are learnable parameters. F' is the feature size of each predicted node and $C = F \times \omega$.

4.2. Embedding module

The embedding module takes $h \in \mathbb{R}^{T\times N\times C}$ and performs three element-wise additions. Note that all the added embeddings are learnable parameters. First, a temporal positional embedding $E_\tau \in \mathbb{R}^{T\times C}$ and $E_\Delta \in \mathbb{R}^{T\times C}$ is added. This embedding is a lookup table that contains 12×24 vectors (corresponding to 24 hours of a day on which 1 hour represents 12×5 minutes) for E_τ and 7 vectors representing each day of the week for E_Δ. We index T vectors that reflect the time at which the measurements have been taken, which allows the model to keep a day long chronological order, thus keeping track of the day cycle.

Second, a node embedding $E_\eta \in \mathbb{R}^{N\times C}$ is added, which is a node identifier that is intended to be captured by the STGA for the creation of the attention score. This works by accentuating the correlation between nodes that are most likely to influence each other. It can be combined with external information about each node such as weather or road conditions, otherwise, it will only act as an identifier.

Finally, a degree embedding $E_\delta \in \mathbb{R}^{D\times C}$ is added, where D is the highest possible degree in the graph $(N+1)$. It contains information relative to the number of connections that a node has and highlights the centrality of a node in the graph.

4.3. STGA module

In our work, we view the key K_{j_t} and query Q_{i_t} as a message passing from node i to node j at each time interval $[t, \cdots, t+\tau]$, therefore by calculating the correlation between Q and K we highlight the links that are more likely to impact the targeted node. In addition, we leverage the graph structural information and the hypothetical node

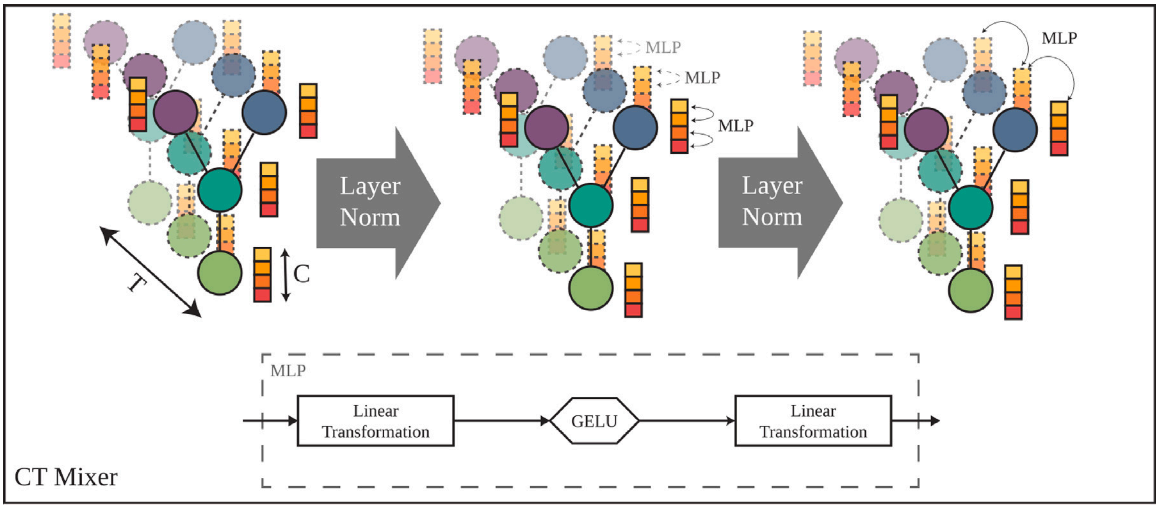

Fig. 2. Channel temporal mixer layer.

future contribution by introducing two biases to represent the edge information A and the expected future contribution $\mathbb{E}$ as described in Eq. (8). The key and query are calculated based on a temporal convolution operation ($\circledast$) with different kernel sizes for each head, where each kernel acts as a different field of views $(1, \tau)$ over the historical data, each head contains (P) kernels which match the desired prediction sequence length. The attention score can then be formulated as follows:

$$att_{(1,\tau_h)}(X_i) = softmax(m_t^{i\to j})V_j^t, j \subset \eta_{(i)} \tag{7}$$

$$m_t^{i\to j} = \frac{Q_i^t K_j^{t^T}}{\sqrt{d_{K^t}}} + A_{i\to j} + \mathbb{E}_{i\to j} \tag{8}$$

$$K = X \circledast W_k,\ Q = X \circledast W_q,\ V = X \circledast W_v \tag{9}$$

$$X \in \mathbb{R}^{C\times N\times T},\ W_{k,q,v} \in \mathbb{R}^{C\times P\times 1\times \tau}$$

We further use a residual connection followed by an L2 normalization. Since the temporal convolution performs a weighted average, the channels size is reduced to C' that can be calculated by Eq. (10), and then we perform a linear transformation to match the input channels.

$$C' = T - \tau + 1 \tag{10}$$

4.4. Estimator model

In traffic, a node exhibit fluctuations that are proportional to its neighboring nodes. Understanding the amount of contribution and the time that it takes to affect each other node is crucial for an accurate forecasting. Calculating this contribution with traditional algorithms is expensive. Furthermore, accurate calculation implies having prior knowledge of the traffic future state which defies the purpose of forecasting. We propose to estimate this contribution with a model that tries to approximate this process. Let $X^i = \{x_1^i, x_2^i, \cdots, x_P^i\}$ and $Y^j = \{y_1^j, y_2^j, \cdots, y_F^j\}$ be two sequences of traffic measurements, where X is the past P values of a source node i and Y is the future F values of a destination node j. $f(X^i, Y^j)$ is a function that calculate the contribution of node i over node j given the two time series. Let g be a function that approximate the contribution score given only X such as $g(X^i) \approx f(X^i, Y^j)$.

Scoring the contribution solely based on historical data is a difficult task, therefore, we decided to enrich the input X with the same embeddings as the main input. The model first split and transforms the features into a lower latent space, an ELU activation function is then applied to smoothly reduce negative values, followed by another linear transforms. We then perform a matrix multiplication followed by a leakyRelu since negative values are meaningless in this case. Finally, the model is scored based on a pre-calculated contribution scores as shown in Fig. 3 based on DTW algorithm. We choose to use MSE as a loss function since we are not interested in a perfect match but rather in the same direction.

Note that the model is trained alongside STGM main loop, and the estimation is incorporated into the main model through all its attention heads as a modulator which influences the propagation amount between each node.

4.5. Fusion module

Having multiple perspectives based on variable time window is a powerful attribute of our model. However, not all perspective are similarly important to the final prediction. Designing a proper mechanism to integrate the different views is crucial to the final result.

The fusion module can be split down into two steps, a selection step and a mixing step, it takes as input the four $h'_{(1,\tau_i)}$ and applies a gated linear unit described in Eq. (12), this act as a gate that selects the information flowing through the network. It is then concatenated along the channel axis and passed through a linear transformation to match the desired number of channels (C) as shown in Eq. (11). The CT Mixer then takes the outputs and shares the information belonging to the same group as shown in Fig. 2, the Mixer layer (CT Mixer) is inspired from Tolstikhin et al. (2021) and consists of two MLP blocks that are shared across the two axis respectively (temporal axis and channel axis) for each node. The idea here is that for each n_i we share the different channels across each time-step for the first part, then we share its timestamps across each channel. This sharing is necessary because the different STGA heads generate the prediction time sequence from different kernel sizes that can be interpreted as a weighted sum over the field of view and to achieve long-range without using a dilation factor that grows with the network depth we use this sharing technique, which allows us to capture the global temporal information at a minimum cost. Additionally, we use a residual connection between each sharing so that the sequential temporal information is not lost.

$$h'' = [\mathop{\Big|\Big|}_{\tau\in K} \sigma(h'_{(1,\tau)})]W_f; K = [2, 3, 6, 7] \tag{11}$$

$W_f \in \mathbb{R}^{4C\times C}$ and σ represent the gated linear unit, and τ is the kernel size used in STGA.

$$GLU(a, b) = a \otimes \frac{1}{1 + e^{-b}} \tag{12}$$

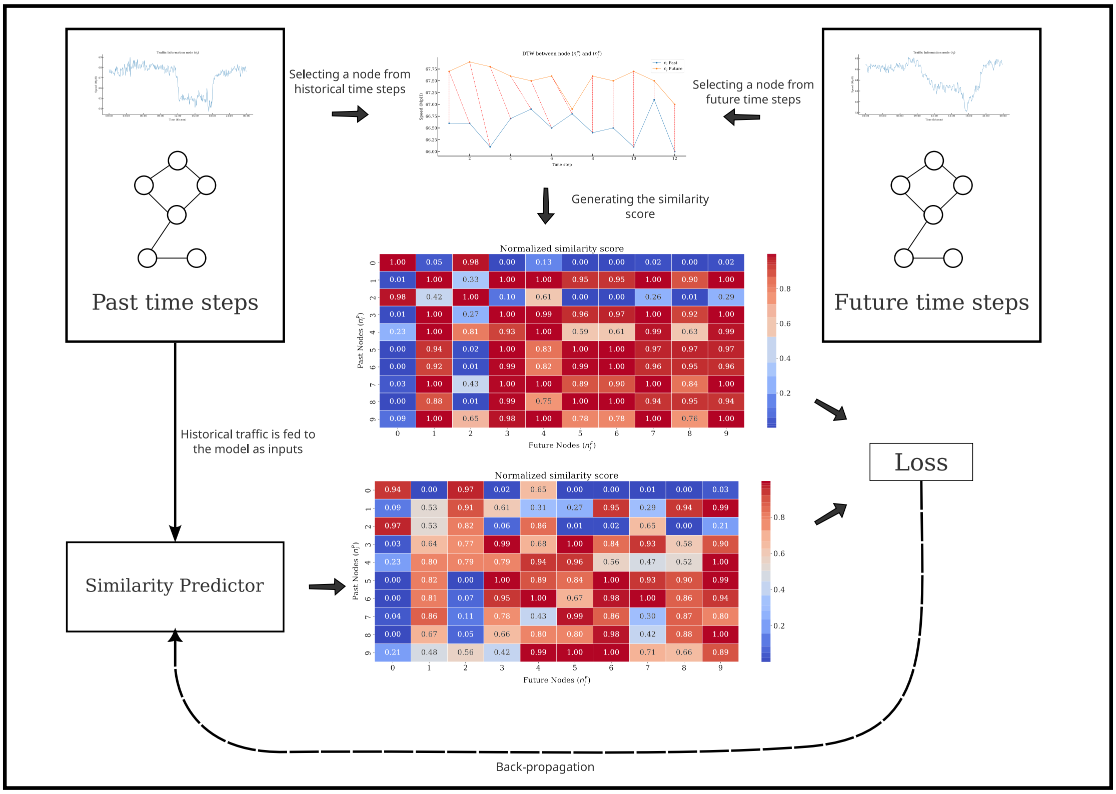
Selecting a node from historical time steps
DTW between node (n_i^p) and (n_j^f)
Selecting a node from future time steps
Past time steps
Future time steps
Generating the similarity score
Normalized similarity score
Past Nodes (n_i^p)
Future Nodes (n_j^f)
Historical traffic is fed to the model as inputs
Loss
Similarity Predictor
Back-propagation

Fig. 3. Estimator training process.

5. Experiments

In this section, we evaluate our model performance against various popular models on three widely used public datasets. For reproducibility, we follow the template benchmark presented in Jiang et al. (2021). We first describe the datasets and their post-processing. Afterward, we give a brief description for each baseline followed by the metrics used for comparison. Finally, we present and discuss the results.

5.1. Datasets

We used three datasets in our experiments:

- **PeMS-Bay:** A group of highways traffic speed sensors manage by the state of California. This dataset was taken from the Bay Area of California and was published by Jiang et al. (2021).
- **PeMSD7M:** The measurements from a subset of the Californian district 7 made available by Yu et al. (2018).
- **METR-LA:** This dataset was made available by Li et al. (2018), which contains measurements taken from the state of Los Angles, the sensors and signals collection are managed in collaboration between The Los Angeles Metropolitan Transportation Authority and the University of Southern California.

All the raw data is collected from sensors that take measurements each 30 seconds, they are then aggregated into 5 minutes intervals. Missing data are filled with a simple linear interpolation. We then apply a Z-score normalization described in Eq. (13). Each dataset includes information about the distances between each sensor, we use Eq. (1) to construct the graph representation. Furthermore, for each dataset we calculate the contribution matrix for all timestamps which are used to train the estimator sub-model. Following Jiang et al. (2021) we use a split ratio of 7:1:2 for all the datasets. Further information about the datasets are presented in Table 1.

$$z = \frac{x - \mu}{\sigma} \tag{13}$$

Table 1
Datasets details.

Dataset	PEMSD7M	PEMS-BAY	METR-LA
Area	North Central	San Francisco Bay	LA
Target	Speed	Speed	Speed
Channels	Speed	Speed	Speed
Nodes	228	325	207
Time steps	12672	52116	34272
Time interval	5mn	5mn	5mn
Start time	2012–05–01	2017-01-01	2012–03–01
End time	2012–06–29	2017-06-30	2012–06–27

where μ is the mean, and σ is the standard deviation of the dataset.

5.2. Evaluation metrics

To evaluate our model, we choose to use three widely used metrics which are described as follows:

- RMSE: Root Mean Squared Error

$$RMSE = \sqrt{\frac{1}{\epsilon}\sum_{i=1}^{\epsilon}\left(\hat{Y}^i - Y^i\right)^2} \tag{14}$$

- MAE: Mean Absolute Error

$$MAE = \frac{1}{\epsilon}\sum_{i=1}^{\epsilon}\left|\hat{Y}^i - Y^i\right| \tag{15}$$

- MAPE: Mean Absolute Percent Error

$$MAPE = \frac{1}{\epsilon}\sum_{i=1}^{\epsilon}\frac{\left|\hat{Y}^i - Y^i\right|}{Y^i} \times 100\% \tag{16}$$

Where ϵ refers to the number of test samples, $\hat{Y}^i$ and Y^i represent the prediction and ground truth for ith element, respectively.

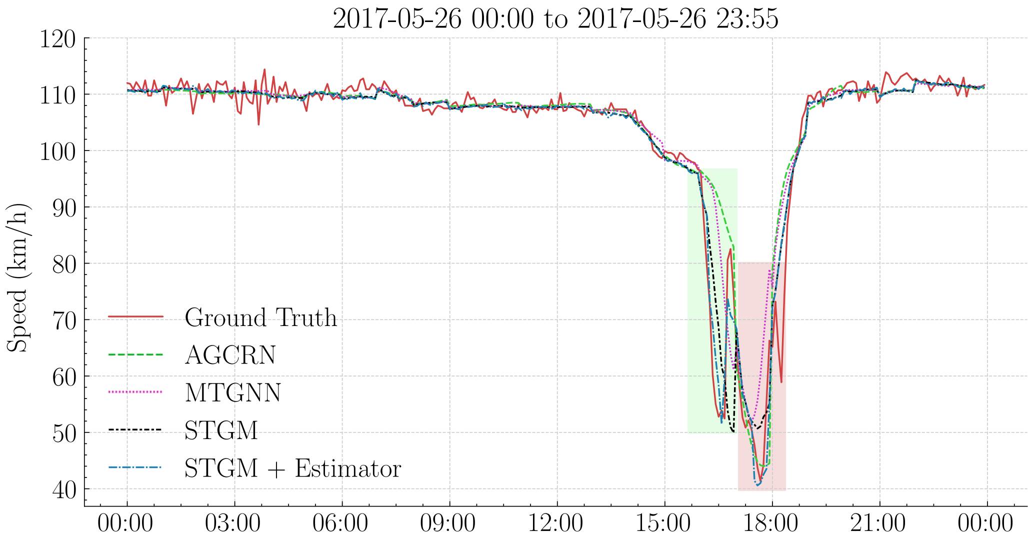

Fig. 4. 1 hour ahead prediction on PeMS-Bay dataset.

5.3. Baselines

We compared the proposed STGM method with the following 9 baselines:

- **HA Historical Average:** We average h previous historical steps, where $h = 7$ and use it as a prediction value.
- **LSTNet Long- and Short-term Time-series network** (Lai, Chang, Yang, & Liu, 2018): Uses LSTM combined with convolution layers to capture both dependencies. An additional autoregressive model is used to capture the non-periodic temporal dependency.
- **DCRNN Diffusion Convolution Recurrent Neural Network** (Li et al., 2018): Model the spatial dependency using a diffusion process characterized by a random walk, which propagate information in a convolution way. The convolution is embedded into a GRU layer to capture the temporal dependency as well.
- **STGCN Spatio-Temporal Graph Convolution Networks** (Yu et al., 2018): A model that uses the Chebyshev polynomials approximation as a kernel in a convolution layer sandwiched between two gated convolutions which are used to capture the spatial–temporal dependency.
- **GWN Graph WaveNet** (Wu, Pan, Long, Jiang, & Zhang, 2019): A GCN based model with an adaptive adjacency matrix learned from node embedding, combined with dilated 1D convolution layers to capture the temporal and spatial dependencies.
- **GMAN Graph Multi Attention Network** (Zheng, Fan, Wang, & Qi, 2019): A Spatial–Temporal graph attention model based on an encoder–decoder architecture.
- **ASTGCN Attention Based Spatial–Temporal Graph Convolution Networks** (Guo, Lin, Feng, Song, & Wan, 2019): An attention based model that combines GCNs to capture the spatial information, and CNNs to capture the temporal information.
- **MTGNN Multivariate Temporal Graph Neural Network** (Wu et al., 2020): An inception inspired model using dilated convolution layers to capture the temporal dependency, and a mix-hop propagation layer based on dynamic graph representation obtained from node embeddings to capture the spatial dependency.
- **AGCRN Adaptive Graph Convolution Recurrent Network** (Bai et al., 2020): A model based on a GCN enhanced layer with a Node Adaptive Parameter Learning Module (NAPL) which is embedded into multiple GRU blocks.
- **DGCRN Dynamic Graph Convolutional Recurrent Network** (Li et al., 2021): Which uses a GRU layer to capture the temporal correlation in addition to an attention mechanism to generate dynamic graph representation for each time-step combined with the distance matrix.

5.4. Implementation and parameters setting

Our model was implemented using the Pytorch 1.13.1 framework and trained on Nvidia RTX 2070 GPU with 8G memory. We use a batch size of 64 and two Adam optimizers with a start learning rate of $1e^{-3}$ that is reduced by a $1e^{-1}$ factor on a plateau. An early stopping based on the validation loss is used to save the best weights. Otherwise, the model is trained for 200 epochs. For both models, we use 32 units in all layers. For each STGA head, the temporal convolution kernel is set to ((1, 2), (1, 3), (1, 6), (1, 7)) respectively with a dilation starting from 1 and increasing by 1 in each layer. The model uses $H = 12$ historical data as input and a prediction length $T = 12$. Finally, we use four layers which provide us with the best performance. We repeated the experiment 5 times and showed the average validation metrics.

For each baseline, we used the same batch size of 64 and an Adam optimizer with a learning rate of $1e^{-3}$ with the same reduction on plateau used to train STGM. We trained each model multiple times until they reached convergence. The number of layers and hidden sizes are set to the values described either in their corresponding papers or in the benchmark provided by Jiang et al. (2021).

5.5. Results

Table 2 shows the prediction performance on the test set for the three datasets, where the best results are shown in bold and the second best are underlined. We have the following observations: (1) The results shows that the historical average (HA) performs poorly and is far from neural networks based models, which proves that classical statistical methods are not suited for such complex and dynamic datasets. (2) We observe that GNN based models are well above the LSTNet which proves the effectiveness of explicitly modeling the spatial dependency. (3) The models that generate dynamically the graph representation such as Graph WaveNet and MTGNN seem to be more flexible and leads to better performance. (4) Overall, our base model performs relatively well with competitive results in small steps and outperforms the previous state-of-art model in the long run. With the addition of the estimator model the error drops considerably, and the model achieves the best performance on all time-steps. While our base model experience less performance drop for larger time-steps which is more evident when utilizing the estimator output. To investigate the benefits of training the two models conjointly and better appreciate the prediction quality over the baselines, we compared STGM to the best performing models on

Table 2
Comparison with the baselines on PEMSBAY, PEMSD7M and METRLA.

Dataset	Model	3 steps (15 min)			6 steps (30 min)			12 steps (60 min)		
		RMSE	MAE	MAPE	RMSE	MAE	MAPE	RMSE	MAE	MAPE
PEMSBAY	HA	6.687	3.333	8.10%	6.686	3.333	8.10%	6.685	3.332	8.10%
	LSTNet	3.224	1.643	3.47%	4.375	2.383	5.04%	5.515	2.974	6.86%
	STGCN	2.827	1.327	2.79%	3.887	1.698	3.81%	4.748	2.055	5.02%
	DCRNN	2.867	1.377	2.96%	3.905	1.726	3.97%	4.798	2.091	4.99%
	GWN	2.759	1.322	2.78%	3.737	1.660	3.75%	4.562	1.991	4.75%
	ASTGCN	3.057	1.435	3.25%	4.066	1.795	4.40%	4.770	2.103	5.30%
	GMAN	4.219	1.802	4.47%	4.143	1.794	4.40%	5.034	2.186	5.29%
	MTGNN	2.849	1.334	2.84%	3.800	1.658	3.77%	4.491	1.950	4.59%
	AGCRN	2.856	1.354	2.94%	3.818	1.670	3.84%	4.570	1.964	4.69%
	DGCRN	2.69	1.28	**2.66** %	**3.63**	1.65	**3.55** %	4.42	1.89	4.43 %
	STGM	2.892	1.355	2.891%	3.754	1.657	3.76%	4.474	1.934	4.54%
	STGM + E	**2.623**	**1.254**	2.699 %	3.687	**1.584**	3.70 %	**4.369**	**1.857**	**4.34** %
PEMSD7M	HA	7.077	3.917	9.90%	7.083	3.920	9.92%	7.095	3.925	9.95%
	LSTNet	4.308	2.423	5.73%	8.951	5.132	12.22%	10.881	6.624	16.72%
	STGCN	4.051	2.124	5.02%	5.532	2.783	6.96%	6.695	3.374	8.74%
	DCRNN	4.143	2.213	5.33%	5.679	2.907	7.41%	7.138	3.670	9.81%
	GWN	3.992	2.130	5.00 %	5.332	2.715	6.75%	6.431	3.266	8.47%
	ASTGCN	4.257	2.340	5.83%	5.506	2.992	7.69%	6.587	3.572	9.48%
	GMAN	5.711	2.877	7.25%	6.171	3.084	7.77%	7.897	3.988	10.02%
	MTGNN	4.032	2.120	5.02%	5.373	2.687	6.70 %	6.496	3.204	8.24%
	AGCRN	4.073	2.167	5.19%	5.479	2.769	6.89%	6.733	3.358	8.55%
	DGCRN	–	–	–	–	–	–	–	–	–
	STGM	4.137	2.166	5.18%	5.449	2.681	6.75%	6.413	3.153	8.12 %
	STGM + E	**3.859**	**2.002**	**4.96** %	**5.248**	**2.502**	**6.62** %	**6.331**	**3.002**	**8.01** %
METR-LA	HA	12.061	5.811	14.94%	12.162	5.940	15.52%	12.152	5.915	15.17%
	LSTNet	7.002	4.082	8.22%	7.124	4.944	9.23%	9.215	5.113	12.56%
	STGCN	5.205	2.988	7.03%	6.101	3.299	8.52%	7.922	3.872	10.05%
	DCRNN	4.986	2.899	6.98%	6.044	3.211	8.36%	7.865	3.854	10.01%
	GWN	5.213	2.814	6.92%	5.999	3.101	8.25%	7.656	3.538	9.98%
	ASTGCN	5.214	3.015	7.11%	6.173	3.376	8.58%	7.997	3.942	10.12%
	GMAN	6.259	3.916	7.76%	6.921	4.227	8.98%	8.322	4.055	10.91%
	MTGNN	5.854	3.304	7.17%	6.544	3.854	8.76%	8.156	3.997	10.28%
	AGCRN	5.712	3.391	7.22%	6.452	3.921	8.78%	8.221	4.021	10.53%
	DGCRN	5.01	2.62	6.63 %	6.05	2.99	8.02%	7.19	3.44	9.73%
	STGM	4.999	2.821	7.02%	5.992	3.032	7.98 %	7.404	3.392	9.62 %
	STGM + E	**4.891**	**2.569**	**6.52** %	**5.759**	**2.857**	**7.80** %	**7.099**	**3.229**	**9.39** %

each time-step using PeMS-Bey dataset. The results presented in Fig. 5 shows that after 6 steps STGM start outperforming the other models and the gap between STGM and the baselines keep increasing, which proves the robustness and stability that our model provides especially for longer temporal sequences. STGM also performs the best regardless of the dataset that is trained on.

To show the effectiveness of our model and its variant, we compare STGM and STGM+E to the best performing models on a randomly selected day (24 hours) and node (sensor) on the PeMS-Bay dataset as shown in Fig. 4. Both STGM and STGM+E captures quite well the different cycles of the day without the need for additional preprocessing such as ASTGCN. Additionally, while all models show a similar performance in smooth traffic, the models express considerable performance difference in rush hours. The green box highlights the first peak, and we observe that most models did not capture the sudden drop in speed at the exception to our model with STGM+E the best match. On the other hand the red area shows that the other models have eventually catch-up, which means that STGM and its variant provide a more sensitive and reactive to rapid changes in the temporal dependency.

Fig. 6 shows the correlation between the parameters count of each model and their performance at step 12 (1 hour ahead prediction). We observe that usually, good-performing models are usually complex and heavy, this is back-up by the top 2 models having over 550k parameters. Meanwhile, STGM parameters count is one of the lowest and yet performs the best compared to bigger models. Which proves that our proposed method is very efficient and can be upscaled to bigger graph structures.

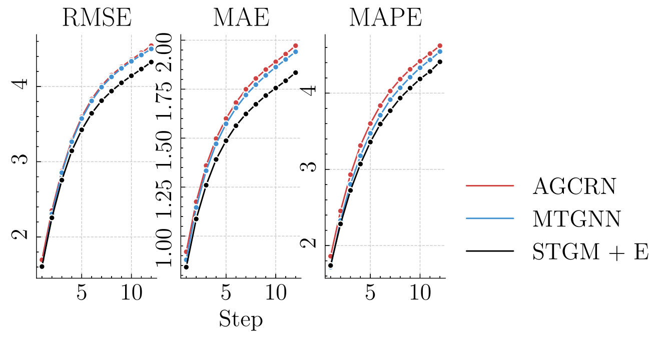

Fig. 5. PeMS-Bay prediction error over time-steps.

5.6. Ablation analysis

To further investigate the importance of each STGM module we design multiple variants and compare them to the original model on the PeMS-Bay dataset at step 12.

1. **STGM:** The full model contains all three modules (Embedding, STGA, Gated linear unit, the mixer layer and the estimator model).
2. **STGM (w/o estimator):** The estimator model is removed from the training and no expected contribution is calculated.
3. **STGM (w/o embedding):** this model has no additional embedding. The input is directly fed to the STGA module (Note that we use a commonly used positional encoding instead).

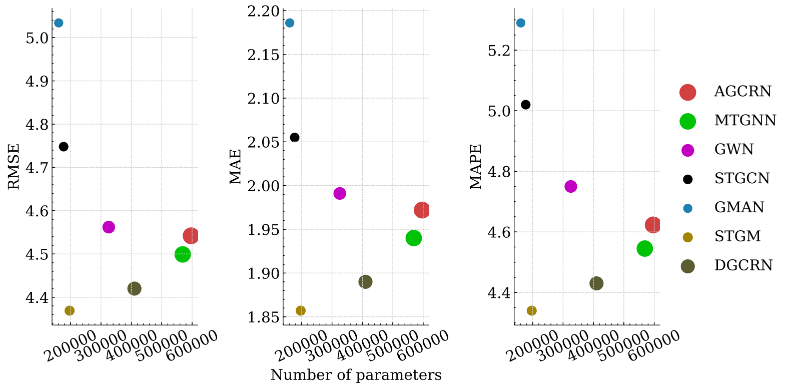

Fig. 6. Model parameters count vs performance.

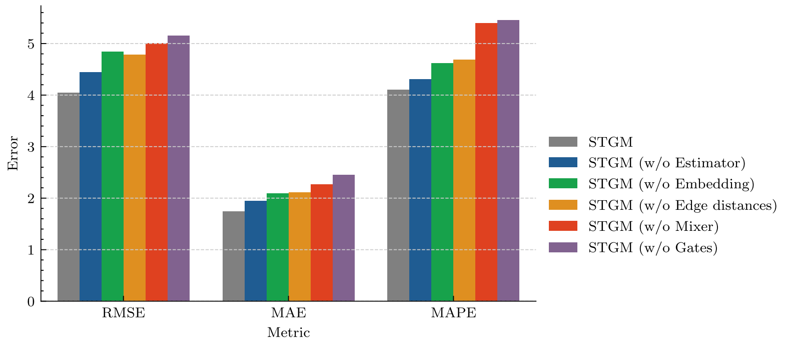

Fig. 7. Model analysis.

4. **STGM (w/o edge distance):** this model does not include the edge features to the STGA attention score, thus producing a fully dynamic graph representation without any knowledge of the actual traffic network structure.
5. **STGM (w/o mixer):** this model does not perform the sharing between the different timestamps and channels.
6. **STGM (w/o gates):** this model does not contain the GLU that acts as a gate instead the outputs of the different STGA heads are directly concatenated and fed to the mixer layer.

As Fig. 7 illustrates, suppressing the expected contribution provided by the estimator model leads to a considerable drop in performance which indicate that the estimator indeed is able to predict a partial causality solely based on historical data. A bigger drop is observed when removing the embedding module due to multiple reasons. First, the node embedding is essential for the STGA module since it constructs the dynamic graph representation based on two crucial components, the traffic features which describe the current traffic state and the node embedding which provides unique identification. Second, the degree embedding provides the STGA with additional information about the importance of each node. Thirdly, the STGA module distinguishes the sequence order based on the temporal embedding. Finally, since the input is shared between the base model and the estimator the provided estimated contribution matrix is no longer accurate due to the lack of information provided to the estimator. In addition, the temporal embedding carries the information relative to the different temporal cycles for the long-range time dependency. On the other hand, removing the external edge information (distance information) does not penalize that much the model since it is additional information designed to help construct the dynamic graph representation and is mitigated by the estimator output, this design choice allows our method to be used even when there is no provided information about the underline graph structure. When not using GLU gate, the model performs poorly, this might be due to the information flowing from the different heads being more of a nuisance and creating interferences. Finally, when it comes to the mixer layer, we see a significant loss in performance because STGA mainly constructs its timestamps based on different kernels with variable fields of view, thus each head gives temporally miss-aligned features. The mixer layer is indispensable since it shuffles the aggregated information into a unified output.

6. Conclusion

In this paper, we proposed an efficient and accurate traffic forecasting model entitled STGM, which enjoys low memory consumption and high precision. Unlike existing studies that use a static graph representation, we proposed STGA module, a dynamic temporal aware adjacency matrix generator. The module uses a custom convolution based multi-heads attention, with multiple fields of view for different temporal perspectives. Additionally, we use the normalized distance matrix coupled with a node agnostic contribution estimator model to modulate the generated graph representations. Furthermore, STGM filters the relevant information using multiple GLU gates and captures the spatio-temporal correlation between nodes through the CT-Mixer module. The CT-Mixer ensures a granular localized temporal integration and global cohesive spatial propagation using multiple shared linear transformations along two axes. The experiments conducted on the three large-scale datasets show that despite the small size of our model, the performance is exceeding those of larger models on long-range predictions. The implementation of our model can be found at: https://github.com/mouradost/stgm.

CRediT authorship contribution statement

Mourad Lablack: Writing – original draft, Methodology, Software, Investigation, Visualization. **Yanming Shen:** Writing – review & editing, Methodology.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grant 62276044, the Innovation Foundation of Science and Technology of Dalian under Grant 2022JJ12SN052, and CAAI-Huawei MindSpore Open Fund.

References

Ahn, J., Eunjeong Ko, & Eun Yi Kim (2016). Highway traffic flow prediction using support vector regression and Bayesian classifier. In *2016 international conference on big data and smart computing* (pp. 239–244). http://dx.doi.org/10.1109/BIGCOMP.2016.7425919, ISSN: 2375-9356.

Arnold, A., Liu, Y., & Abe, N. (2007). Temporal causal modeling with graphical granger methods. In *Proceedings of the 13th ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 66–75). New York, NY, USA: Association for Computing Machinery, http://dx.doi.org/10.1145/1281192.1281203.

Bai, L., Yao, L., Li, C., Wang, X., & Wang, C. (2020). Adaptive graph convolutional recurrent network for traffic forecasting. arXiv preprint arXiv:2007.02842.

Barnett, L., Barrett, A. B., & Seth, A. K. (2009). Granger causality and transfer entropy are equivalent for Gaussian variables. *Physical Review Letters*, *103*, Article 238701. http://dx.doi.org/10.1103/PhysRevLett.103.238701, URL: https://link.aps.org/doi/10.1103/PhysRevLett.103.238701.

Bellman, R., & Kalaba, R. (1959). On adaptive control processes. *IRE Transactions on Automatic Control*, *4*(2), 1–9. http://dx.doi.org/10.1109/TAC.1959.1104847.

Chen, C., Li, K., Teo, S. G., Zou, X., Wang, K., Wang, J., et al. (2019). Gated residual recurrent graph neural networks for traffic prediction. *Proceedings of the AAAI Conference on Artificial Intelligence*, *33*(01), 485–492. http://dx.doi.org/10.1609/aaai.v33i01.3301485, URL: https://ojs.aaai.org/index.php/AAAI/article/view/3821, Number: 01.

Deng, J., Chen, X., Jiang, R., Song, X., & Tsang, I. W. (2021). ST-Norm: Spatial and temporal normalization for multi-variate time series forecasting. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining* (pp. 269–278). Virtual Event Singapore: ACM, http://dx.doi.org/10.1145/3447548.3467330.

Diao, Z., Wang, X., Zhang, D., Liu, Y., Xie, K., & He, S. (2019). Dynamic spatial-temporal graph convolutional neural networks for traffic forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence*, *33*(01), 890–897. http://dx.doi.org/10.1609/aaai.v33i01.3301890, URL: https://ojs.aaai.org/index.php/AAAI/article/view/3877, Number: 01.

Fang, Z., Long, Q., Song, G., & Xie, K. (2021). Spatial-temporal graph ODE networks for traffic flow forecasting. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining* (pp. 364–373). Virtual Event Singapore: ACM, http://dx.doi.org/10.1145/3447548.3467430.

Granger, C. W. J. (1969). Investigating causal relations by econometric models and cross-spectral methods. *Econometrica*, *37*(3), 424–438. http://dx.doi.org/10.2307/1912791.

Guen, V. L., & Thome, N. (2019). Shape and time distortion loss for training deep time series forecasting models. (p. 13).

Guo, S., Lin, Y., Feng, N., Song, C., & Wan, H. (2019). Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence*, *33*(01), 922–929. http://dx.doi.org/10.1609/aaai.v33i01.3301922, URL: https://ojs.aaai.org/index.php/AAAI/article/view/3881, Number: 01.

Hamilton, W. L., Ying, R., & Leskovec, J. (2018). Inductive representation learning on large graphs. arXiv:1706.02216 [Cs, Stat], URL: http://arxiv.org/abs/1706.02216.

Jiang, R., Yin, D., Wang, Z., Wang, Y., Deng, J., Liu, H., et al. (2021). DL-traff: Survey and benchmark of deep learning models for urban traffic prediction. CoRR abs/2108.09091, URL: https://arxiv.org/abs/2108.09091.

Kong, X., Xing, W., Wei, X., Bao, P., Zhang, J., & Lu, W. (2020). STGAT: Spatial-Temporal graph attention networks for traffic flow forecasting. *IEEE Access*, *8*, 134363–134372. http://dx.doi.org/10.1109/ACCESS.2020.3011186, Conference Name: IEEE Access.

Lai, G., Chang, W.-C., Yang, Y., & Liu, H. (2018). Modeling long- and short-term temporal patterns with deep neural networks. In *The 41st international ACM SIGIR conference on research and development in information retrieval* (pp. 95–104). New York, NY, USA: Association for Computing Machinery, http://dx.doi.org/10.1145/3209978.3210006.

Lan, S., Ma, Y., Huang, W., Wang, W., Yang, H., & Li, P. (2022). DSTAGNN: Dynamic spatial-temporal aware graph neural network for traffic flow forecasting, no. 162. (pp. 11906–11917).

Li, F., Feng, J., Yan, H., Jin, G., Jin, D., & Li, Y. (2021). Dynamic graph convolutional recurrent network for traffic prediction: Benchmark and solution. arXiv:2104.14917, arXiv:arXiv:2104.14917.

Li, Y., Yu, R., Shahabi, C., & Liu, Y. (2018). Diffusion convolutional recurrent neural network: Data- driven traffic forecasting. arXiv:1707.01926 [Cs, Stat], URL: http://arxiv.org/abs/1707.01926.

Liang, Y., Ouyang, K., Jing, L., Ruan, S., Liu, Y., Zhang, J., et al. (2019). Urbanfm: Inferring fine-grained urban flows. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining* (pp. 3132–3142). http://dx.doi.org/10.1145/3292500.3330646, URL: http://arxiv.org/abs/1902.05377.

Liao, B., Zhang, J., Wu, C., McIlwraith, D., Chen, T., Yang, S., et al. (2018). Deep sequence learning with auxiliary information for traffic prediction. arXiv:1806.07380 [Cs], URL: http://arxiv.org/abs/1806.07380.

Lin, J. (2016). Study on the prediction of urban traffic flow based on ARIMA model. *DEStech Transactions on Engineering and Technology Research*, (iceta), http://dx.doi.org/10.12783/dtetr/iceta2016/7033, URL: http://dpi-proceedings.com/index.php/dtetr/article/view/7033, Number: iceta.

Liu, Z., Du, W., Yan, D.-m., Chai, G., & Guo, J.-h. (2018). Short-term traffic flow forecasting based on combination of K-nearest neighbor and support vector regression. *Journal of Highway and Transportation Research and Development (English Edition)*, *12*(1), 89–96. http://dx.doi.org/10.1061/JHTRCQ.0000615, URL: https://ascelibrary.org/doi/abs/10.1061/JHTRCQ.0000615, Publisher: Research Institute of Highway, Ministry of Transport, Beijing (RIOH).

Liu, L., Zhen, J., Li, G., Zhan, G., & Lin, L. (2019). ACFM: A dynamic spatial-temporal network for traffic prediction. *IEEE Transactions on Intelligent Transportation Systems*, 14.

Mai, T., Ghosh, B., & Wilson, S. (2014). Short-term traffic-flow forecasting with auto-regressive moving average models. *Proceedings of the ICE - Transport*, *167*, 232–239. http://dx.doi.org/10.1680/tran.12.00012.

Mourad, L., Qi, H., Shen, Y., & Yin, B. (2019). ASTIR: Spatio-temporal data mining for crowd flow prediction. *IEEE Access*, *7*, 175159–175165. http://dx.doi.org/10.1109/ACCESS.2019.2950956, Conference Name: IEEE Access.

Shao, Z., Zhang, Z., Wang, F., Wei, W., & Xu, Y. (2022). Spatial-temporal identity: A simple yet effective baseline for multivariate time series forecasting. arXiv:2208.05233, arXiv:arXiv:2208.05233.

Shao, Z., Zhang, Z., Wang, F., & Xu, Y. (2022). Pre-training enhanced spatial-temporal graph neural network for multivariate time series forecasting. In *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining* (pp. 1567–1577). http://dx.doi.org/10.1145/3534678.3539396, arXiv:2206.09113.

Shao, Z., Zhang, Z., Wei, W., Wang, F., Xu, Y., Cao, X., et al. (2022). Decoupled dynamic spatial-Temporal graph neural network for traffic forecasting. *Proceedings of the VLDB Endowment*, *15*(11), 2733–2746. http://dx.doi.org/10.14778/3551793.3551827.

Shi, X., Qi, H., Shen, Y., Wu, G., & Yin, B. (2021). A spatial–temporal attention approach for traffic prediction. *IEEE Transactions on Intelligent Transportation Systems*, *22*(8), 4909–4918. http://dx.doi.org/10.1109/TITS.2020.2983651.

Sun, B., Zhao, D., Shi, X., & He, Y. (2021). Modeling global spatial–Temporal graph attention network for traffic prediction. *IEEE Access*, *9*, 8581–8594. http://dx.doi.org/10.1109/ACCESS.2021.3049556, Conference Name: IEEE Access.

Tolstikhin, I. O., Houlsby, N., Kolesnikov, A., Beyer, L., Zhai, X., Unterthiner, T., et al. (2021). MLP-mixer: An all-MLP architecture for vision. CoRR abs/2105.01601, arXiv:2105.01601, URL: https://arxiv.org/abs/2105.01601.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., et al. (2017). Attention is all you need. In *Advances in neural information processing systems* (pp. 5998–6008).

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., & Bengio, Y. (2018). Graph attention networks. arXiv:1710.10903 [Cs, Stat], URL: http://arxiv.org/abs/1710.10903.

Wei, C., & Sheng, J. (2020). Spatial-temporal graph attention networks for traffic flow forecasting. *IOP Conference Series: Earth and Environmental Science*, *587*, Article 012065. http://dx.doi.org/10.1088/1755-1315/587/1/012065, URL: https://iopscience.iop.org/article/10.1088/1755-1315/587/1/012065.

Wu, Z., Pan, S., Long, G., Jiang, J., Chang, X., & Zhang, C. (2020). Connecting the dots: Multivariate time series forecasting with graph neural networks. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 753–763). New York, NY, USA: Association for Computing Machinery, http://dx.doi.org/10.1145/3394486.3403118.

Wu, Z., Pan, S., Long, G., Jiang, J., & Zhang, C. (2019). Graph WaveNet for deep spatial-temporal graph modeling. (p. 7). http://dx.doi.org/10.48550/arXiv.1906.00121, arXiv preprint arXiv:1906.00121.

Xie, Y., Xiong, Y., & Zhu, Y. (2020). SAST-GNN: A self-attention based spatio-temporal graph neural network for traffic prediction. In Y. Nah, B. Cui, S.-W. Lee, J. X. Yu, Y.-S. Moon, & S. E. Whang (Eds.), *Lecture notes in computer science*, *Database systems for advanced applications* (pp. 707–714). Cham: Springer International Publishing, http://dx.doi.org/10.1007/978-3-030-59410-7_49.

Yin, X., Wu, G., Wei, J., Shen, Y., Qi, H., & Yin, B. (2021). Multi-stage attention spatial-temporal graph networks for traffic prediction. *Neurocomputing*, *428*, 42–53. http://dx.doi.org/10.1016/j.neucom.2020.11.038, URL: https://linkinghub.elsevier.com/retrieve/pii/S0925231220318312.

Yu, B., Li, M., Zhang, J., & Zhu, Z. (2019). 3D graph convolutional networks with temporal graphs: A spatial information free framework for traffic forecasting. arXiv, arXiv:1903.00919.

Yu, B., Yin, H., & Zhu, Z. (2018). Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. In *Proceedings of the twenty-seventh international joint conference on artificial intelligence* (pp. 3634–3640). Stockholm, Sweden: International Joint Conferences on Artificial Intelligence Organization, http://dx.doi.org/10.24963/ijcai.2018/505, URL: https://www.ijcai.org/proceedings/2018/505.

Yuan, Z., Zhou, X., & Yang, T. (2018). Hetero-convlstm: A deep learning approach to traffic accident prediction on heterogeneous spatio-temporal data. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining* (pp. 984–992). London United Kingdom: ACM, http://dx.doi.org/10.1145/3219819.3219922, URL: https://dl.acm.org/doi/10.1145/3219819.3219922.

Zhang, X., Huang, C., Xu, Y., Xia, L., Dai, P., Bo, L., et al. (2021). Traffic flow forecasting with spatial-temporal graph diffusion network. arXiv, arXiv:2110.04038.

Zhang, L., Liu, Q., Yang, W., Wei, N., & Dong, D. (2013). An improved K-nearest neighbor model for short-term traffic flow prediction. *Procedia - Social and Behavioral Sciences*, *96*, 653–662. http://dx.doi.org/10.1016/j.sbspro.2013.08.076, URL: https://www.sciencedirect.com/science/article/pii/S1877042813022027.

Zhang, J., Shi, X., Xie, J., Ma, H., King, I., & Yeung, D.-Y. (2018). GaAN: Gated attention networks for learning on large and spatiotemporal graphs. arXiv:1803.07294 [Cs], URL: http://arxiv.org/abs/1803.07294.

Zhang, J., Zheng, Y., & Qi, D. (2017). Deep spatio-temporal residual networks for citywide crowd flows prediction. In *Thirty-First AAAI conference on artificial intelligence* (p. 7).

Zhao, L., Song, Y., Zhang, C., Liu, Y., Wang, P., Lin, T., et al. (2020). T-GCN: A temporal graph convolutional network for traffic prediction. *IEEE Transactions on Intelligent Transportation Systems*, *21*(9), 3848–3858. http://dx.doi.org/10.1109/TITS.2019.2935152, Conference Name: IEEE Transactions on Intelligent Transportation Systems.

Zheng, C., Fan, X., Wang, C., & Qi, J. (2019). GMAN: A graph multi-attention network for traffic prediction. arXiv:1911.08415 [Cs, Eess], URL: http://arxiv.org/abs/1911.08415.

Zhu, J., Song, Y., Zhao, L., & Li, H. (2020). A3T-GCN: Attention temporal graph convolutional network for traffic forecasting. arXiv:2006.11583 [Cs, Stat], URL: http://arxiv.org/abs/2006.11583.