LEARNING TO REMEMBER PATTERNS: PATTERN MATCHING MEMORY NETWORKS FOR TRAFFIC FORECASTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Traffic forecasting is a challenging problem due to complex road networks and sudden speed changes caused by various events on roads. A number of models have been proposed to solve this challenging problem with a focus on learning spatio-temporal dependencies of roads. In this work, we propose a new perspective of converting the forecasting problem into a pattern matching task, assuming that large data can be represented by a set of patterns. To evaluate the validness of the new perspective, we design a novel traffic forecasting model, called Pattern-Matching Memory Networks (PM-MemNet), which learns to match input data to the representative patterns with a key-value memory structure. We first extract and cluster representative traffic patterns, which serve as keys in the memory. Then via matching the extracted keys and inputs, PM-MemNet acquires necessary information of existing traffic patterns from the memory and uses it for forecasting. To model spatio-temporal correlation of traffic, we proposed novel memory architecture GCMem, which integrates attention and graph convolution for memory enhancement. The experiment results indicate that PM-MemNet is more accurate than state-of-the-art models, such as Graph WaveNet with higher responsiveness. We also present a qualitative analysis result, describing how PM-MemNet works and achieves its higher accuracy when road speed rapidly changes.

1 INTRODUCTION

Traffic forecasting is a challenging problem due to complex road networks, varying patterns in the data, and intertwined dependencies among models. This implies that prediction methods should not only find intrinsic spatio-temporal dependencies among many roads, but also quickly responds to irregular congestion and various traffic patterns (Lee et al., 2020) caused by external factors, such as accidents or weather conditions (Vlahogianni et al., 2014; Li & Shahabi, 2018; Xie et al., 2020; Jiang & Luo, 2021). To resolve the challenge and successfully predict traffic conditions, many deep learning models have been proposed. Examples include graph convolutional neural networks (GCNs) (Bruna et al., 2014) with recurrent neural networks (RNNs) (Siegelmann & Sontag, 1991), which outperform conventional statistical methods such as ARIMA (Vlahogianni et al., 2014; Li et al., 2018). Attention-based models, such as GMAN (Zheng et al., 2020) are also explored to better handle the complex spatio-temporal dependency. Graph WaveNet (Wu et al., 2019) adopts a diffusion process with self-learning adjacency matrix and dilated convolutional neural networks (CNNs), achieving state-of-the-art performance. Although effective, existing models have a weakness in that they do not accurately forecast when conditions are abruptly changed (e.g., rush hour and accidents).

In this work, we aim to design a novel method for modeling spatial-temporal dependencies of roads and to improve forecasting performance. In doing so, we first extract representative traffic patterns from historical data of roads, as we find that there are similar traffic patterns among roads and a set of traffic patterns can be generalized for the roads with similar spatio-temporal features. Figure 1 shows example speed patterns (left, 90-minute window) that we extract from many different roads and a represented traffic pattern (right time-series). With the represented patterns, we transform the conventional forecasting problem into a pattern matching task–given spatio-temporal features, which pattern would be best matched to refer for predicting future traffic conditions. Given insights

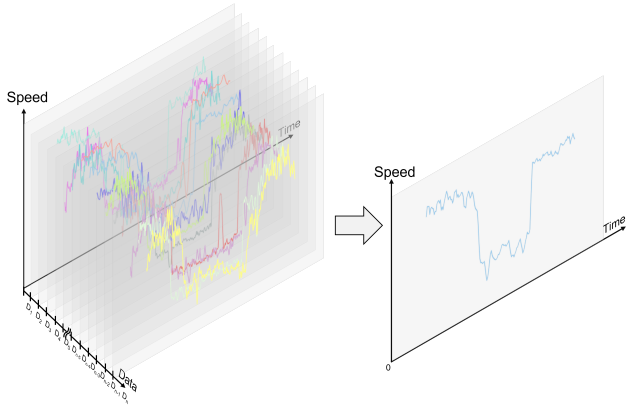

Figure 1: (left) Multiple traffic data with similar pattern, (right) extracted representative pattern

from huge success of neural memory networks in natural language processing and machine translation (Weston et al., 2015; Sukhbaatar et al., 2015; Kaiser et al., 2017; Madotto et al., 2018), we design graph convolutional memory networks for managing representative patterns in spatial-temporal manners, called GCMem. Lastly, we design PM-MemNet, which utilizes the representative patterns from GCMem for traffic forecasting. PM-MemNet consists of an encoder and decoder. The encoder consists of temporal embedding with stacked GCMem, which generates meaningful representations via memorization and the decoder is composed with gated recurrent unit (GRU) with GCMem. We compare PM-MemNet to existing state-of-the-art models and find that PM-MemNet outperforms existing models. We also present a qualitative analysis where we further investigate strengths of PM-MemNet with a traffic patterns, where speed abruptly changes and high responsiveness of a model is desired for accurate forecasting.

The experimental results indicate that PM-MemNet achieves state-of-the-art performance, especially in the long-term prediction results, compared to existing deep learning models. To further investigate characteristics PM-MemNet, we conduct an ablation study with various decoder architectures and find that PM-MemNet shows the best performance. Lastly, we discuss limitations of this work and future directions of neural memory networks in the traffic forecasting domain.

The contributions of this work include: (1) computing representative traffic patterns of roads, (2) design of GCMem to manage the representative patterns, (3) design of PM-MemNet that matches and uses most appropriate patterns from GCMem for traffic forecasting, (4) evaluation of PM-MemNet compared to state-of-the-art models, (5) qualitative analysis for identifying strengths of PM-MemNet, and (6) discussion of limitations and future research directions.

2 Related Work

2.1 Traffic Forecasting

Deep learning models achieve a huge success by effectively capturing spatio-temporal features in the traffic forecasting task. First, much work shows that RNN-based models overwhelm conventional temporal modeling approaches, such as ARIMA and SVR (Vlahogianni et al., 2014; Li et al., 2018). More recently, many studies demonstrate that attention (Zheng et al., 2020; Park et al., 2020) and CNNs (Yu et al., 2018; Wu et al., 2019) record better performance in the long-term period prediction task, compared to the RNN-based models. In terms of spatial modeling, Zhang et al. (2016) propose a CNN-based spatial modeling method in the Euclidean space. As another line of modeling methods, using graph structures for managing complex road networks become popular, such as GCNs. However, there are difficulties in using GCNs in the modeling process, such as building an adjacency matrix and dependence of GCNs on invariant connectivity in the adjacency matrix. To overcome the difficulties, a set of approaches are proposed, such as graph attention models (GATs) make it possible to calculate edge importance dynamically (Park et al., 2020). GWNet (Wu et al., 2019) adopts the self-adaptive adjacency matrix to capture hidden spatial dependencies in training.

Although effective, forecasting models still suffer from inaccurate prediction with abruptly changing speeds and instability with lagging patterns on long-term periods. To address these challenges, we build, save, retrieve representative traffic patterns for predicting speed rather than for forecasting with an input sequence directly.

2.2 Neural Memory Networks

Neural Memory Networks are widely used for sequence-to-sequence modeling in the natural language processing and machine translation domain. Memory networks are first proposed by Weston et al. (2015) to answer the query more precisely even when datasets are large with long-term memory. Memory networks perform read and write operations given input queries. Sukhbaatar et al. (2015) introduce end-to-end memory networks, which can updates memory in an end-to-end manner and thus remove the need to memory with queries directly. By end-to-end learning of memory, models can be easily applicable to realistic settings. Further, by using adjacent weight tying, they can achieve recurrent characteristics that can enhance memory context learning. Also, Kaiser et al. (2017) propose novel memory networks that can be utilized in various domains, where life-long one-shot learning is needed. Also, Madotto et al. (2018) introduce Mem2Seq, which integrates the multi-hop attention mechanism with memory networks. In our work, we utilize memory networks for the traffic pattern modeling due to similarity of the tasks in nature and develop novel graph convolutional memory networks, GCMem to better model spatio-temporal correlation of given traffic patterns.

3 Proposed Approach

In this section, we define the traffic forecasting problem, describe how we extract key patterns in the traffic data which serve as keys, and introduce our model, PM-MemNet.

3.1 Problem Setting

To handle spatial relationships of roads, we utilize a road network graph. We define a road network graph as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{A})$, where $\mathcal{V}$ is a set of all different nodes with $|\mathcal{V}| = N$, $\mathcal{E}$ is a set of the edges representing the connectivity between nodes, and $\mathcal{A} \in \mathbb{R}^{N \times N}$ is a weighted adjacency matrix that contains the connectivity and edge weight information. An edge weight is calculated based on the distance and the direction of the edge between two connected nodes. As used in the previous approaches (Li et al., 2018; Wu et al., 2019; Zheng et al., 2020; Park et al., 2020), we calculate edge weights via Gaussian kernel as follows: $A_{i,j} = \exp -\frac{\text{dist}_{ij}^2}{\sigma^2}$, where dist_{ij} is a distance between node i and j and σ is the standard deviation of the distances.

Prior work formulates a traffic forecasting problem as a simple spatiotemporal data prediction problem (Li et al., 2018; Wu et al., 2019; Zheng et al., 2020; Park et al., 2020), aiming to predict values in the next T time steps with previous T' historical traffic data input and an adjacency matrix. Traffic data at time t is represented by a graph signal matrix $X_{\mathcal{G}}^t \in \mathbb{R}^{N \times d_{in}}$, where d_{in} is the number of features, such as speed, flow, and time of the day. In summary, the goal of previous work is to learn a mapping function $f(\cdot)$ to directly predict future T graph signals from T' historical input graph signals:

$$\left[{X_{\mathcal{G}}}^{(t-T'+1)}, \cdots, {X_{\mathcal{G}}}^{(t)}\right] \xrightarrow{f(\cdot)} \left[{X_{\mathcal{G}}}^{(t+1)}, \cdots, {X_{\mathcal{G}}}^{(t+T)}\right]$$

The goal in this work is different the previous work in that we aim at predicting future traffic speeds from patterned data, instead of utilizing input $X_{\mathcal{G}}$ directly. We denote $p \in \mathbb{P}$ as a set of given traffic patterns and $d : \mathbb{X} \times \mathbb{P} \rightarrow [0, \infty)$ as a distance function for pattern matching. Then, our problem is to train mapping function $f(\cdot)$ as follows:

$$\left[{X_{\mathcal{G}}}^{(t-T'+1)}, \cdots, {X_{\mathcal{G}}}^{(t)}\right] \xrightarrow{d(\cdot), k-NN} \left[P_1^t, \ldots, P_N^t\right] \xrightarrow{f(\cdot)} \left[{X_{\mathcal{G}}}^{(t+1)}, \cdots, {X_{\mathcal{G}}}^{(t+T)}\right],$$

where $P_i^t = \{p_1, \ldots, p_k\}$ is k-nearest neighboring traffic patterns of node i in time t, with a distance function d. Note that p_j is the j-th nearest neighbor pattern.

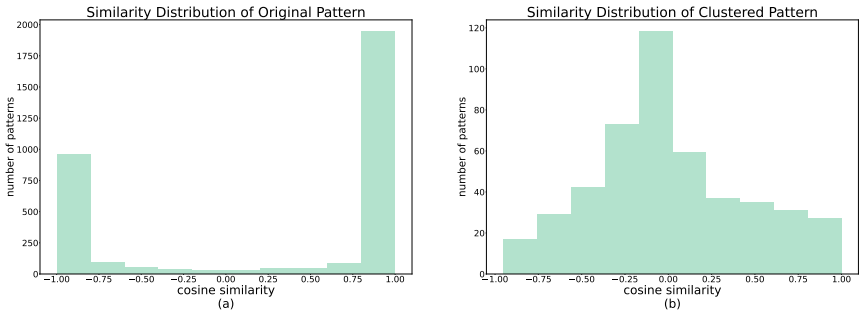

Figure 2: (a) Cosine similarity distribution of an arbitrary pattern in original patterns and (b) clustered patterns

3.2 Key Extraction from Traffic Pattern

Analyzing the traffic data, we find that the data has repeating patterns. Besides, in the traffic data, the leading and trailing patterns have high correlation even in short-term periods. To employ the revelation, we extract daily patterns from historical data by analyzing time-oriented average speed on each road in the road network. Then, from the daily patterns, with window size T', we sample a pattern set $\mathbb{P}$ without any duplication in range. After sampling, we investigate similarity distribution among all patterns via cosine similarity measurement. For cosine similarity, all patterns are normalized into zero-based signals. As shown in Figure 2 (a), we certify that pattern set $\mathbb{P}$ has class imbalance or imbalanced distribution, which means items are not equally positioned in different classes. In our pattern matching approach, class imbalance in pattern data causes ineffectiveness in memory learning and makes training biased. For example, some patterns p have too many similar candidates for the k-NN. To cope with the issue, we use the technique for correcting class imbalance which Lin et al. (2017) have proposed, where we undersample abundant patterns through K-means clustering to make balanced distribution, as shown in Figure 2 (b). In the end, the clustered patterns are used as keys for memory access. In the remains of our paper, the extracted pattern set is marked as $\mathbb{P}$.

3.3 Neural Memory Architecture

Conventionally, memory networks have used the attention mechanism for memory units to enhance memory reference performance, but this attention-only approach cannot effectively capture spatial dependencies among roads. To address this issue, we design a new memory architecture, GCMem (Figure 3 b), which integrates multi-layer memory with the attention mechanism (Madotto et al., 2018) and graph convolution (Bruna et al., 2014). By using GCMem, a model can capture both pattern-level attention and graph-aware information sharing via GCNs.

To effectively handle representative patterns in spatio-temporal manners, we utilize several techniques. First, we use the adjacent weight modification and tying technique in MemNN (Sukhbaatar et al., 2015) that has been used for effective memorization of sentences and connection search between query and memorized sentences. As Sukhbaatar et al. (2015); Madotto et al. (2018) propose the technique to capture information from memorized sentences, they make use of sentence-level attention. However, their memory only learns pattern similarity and cannot handle spatial dependency. As such, using the same method for traffic forecasting is insufficient, it is not able to handle a graph structure which is essential for building spatial dependencies of roads. In order to consider the graph structure while maintaining the original sentence-level attention score, we further use a adjacency matrix, learnable adaptive matrix, and attention scores for the GCNs. As a result, adjacent memory cells can effectively retain attention mechanisms while considering a graph structure.

Figure 3 (b) and (c) shows our proposed graph convolution memory architecture. For memory reference, we utilize pattern set $\mathbb{P}$, which contains the extracted traffic patterns (in Section 3.2) and k-Nearest Neighbor (k-NN) with distance function, $d(\cdot)$. For each input traffic data, $X_i \in \mathbb{R}^{T' \times d_{in}}$ in node i, based on k-nearest patterns and the distance function, we build a representative memory

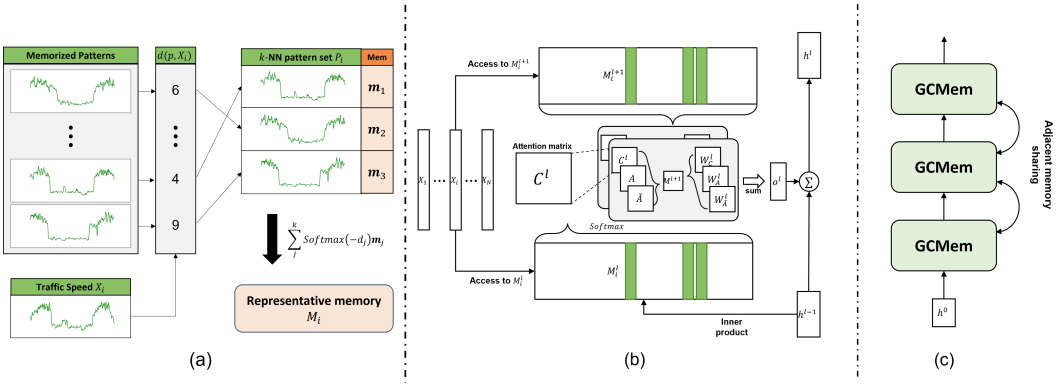

Figure 3: (a) Representative memory selection among k-nearest patterns for X_i with k=3, $d_j = d(p_j, X_i)$, (b) GCMem architecture with GCNs (gray blocks), (c) 3-layered GCMem example.

M_i^l as follows:

$$M_i^l = \sum_j^k \text{Softmax}(-d_j)\mathbf{m}_j, \tag{1}$$

where $\mathbf{m}_j$ is the memory context for p_j, $d_j = g(X_i, p_j)$, and $\text{Softmax}(z_i) = e^{z_i}/\sum_j e^{z_j}$. We summarize our representative memory selection process in Figure 3 (a).

Then, for each memory context M_j, PM-MemNet calculates pattern-level attention scores, $\alpha_{i,j}^l$ as follows:

$$\alpha_{i,j}^l = \text{Softmax}(\frac{h_i^{l-1}(M_j^l)^\top}{\sqrt{d_h}}), \tag{2}$$

where $h_i^{l-1} \in \mathbb{R}^{d_h}$ is the previous hidden state of node i. We denote an attention matrix as $C^l \in \mathbb{R}^{N\times N}$, where $C_{i,j}^l = \alpha_{i,j}^l$. Then, given memory unit $M^{l+1} \in \mathbb{R}^{N\times d_h}$ for the next layer $l+1$, we calculate output feature, o^l with graph convolution as shown below:

$$o^l = \sum_i^h \left(W_{A,i}^l M^{l+1}A + W_{\tilde{A},i}^l M^{l+1}\tilde{A} + W_{C,i}^l M^{l+1}C^l\right), \tag{3}$$

where $A \in \mathbb{R}^{N\times N}$ is an adjacency matrix and $\tilde{A} = \text{Softmax}(ReLU(E_1 E_2^\top))$ is a learnable matrix, which captures hidden spatio-temporal connections (Wu et al., 2019; Shi et al., 2019). E_1 and E_2 are learnable node embedding vectors and W is a learnable matrix. We use $ReLU$ as an activation function. Then, we update hidden states by $h^l = h^{l-1} + o^l$. Before we update the hidden states, we apply batch normalization on o^l.

3.4 Encoder Architecture

PM-MemNet handles traffic patterns and its corresponding memories. Although the patterns in the memory provide enough information for training, there exist other types of data that can be also used for prediction. For example, different roads would have their own patterns that may not be captured in advance (e.g., unique periodicity of roads around industry complex (Lee et al., 2020). In addition, there would be some noise and anomalous patterns due to various events (e.g., accidents), which are not encoded when patterns are grouped. As such we provide embedding for time and noise which the encoder uses for generating input representations with given patterns. Specifically, for the time series $\mathbf{T} = [t - T' + 1, \ldots, t]$ and noise $N_i = X_i - p_1$, we calculate its representation, h_i^0 as shown below:

$$h_i^0 = emb(T) + W_n N_i, \tag{4}$$

where emb and W_n represent a learnable embedding for the time of day and a learnable matrix, respectively. PM-MemNet updates h_i^0 by L-layer GCMem.

3.5 DECODER ARCHITECTURE

We build a base decoder architecture, using a GRU followed by L-stacked GCMem. For each t step prediction, our decoder predicts the value of next time step $t+1$ with previous prediction $\hat{y}^t$. The hidden states from GRU turn to input for the L-stacked GCMem. Similar to our encoder architecture, GCMem updates hidden states with attention and GCNs. Instead of using updated hidden states for prediction, we utilize layer-level self-attention in the decoder. Specifically, for each GCMem layer l and node i, we calculate energy, $e_{i,l}$ using the previous hidden state h_i^{l-1} and memory context M_i^l as shown below:

$$e_{i,l} = \frac{(h_i^{l-1})(M_i^l W^l)^\top}{\sqrt{d_h}}, \tag{5}$$

where d_h is hidden size and W^l is learnable matrix. Then, with output feature o_i^l of each layer l and node i, we can predict $\hat{y}_i^t$ as:

$$\hat{y}_i^t = \sum_{l}^{L} \alpha_{i,l} o_i^l W_{proj}, \tag{6}$$

where $W_{proj} \in \mathbb{R}^{d_h \times d_{out}}$ is a projection layer and $\alpha_{i,l} = \text{Softmax}(e_{i,l})$. By layer-level attention, PM-MemNet utilizes information from each layer more effectively.

4 EVALUATION

In this section, we conduct experiments for comparing PM-MemNet to existing models in terms of accuracy. We use two datasets in the experiment–METR-LA and NAVER-Seoul. METR-LA contains 4-month speed data of 207 sensors of Los Angeles highways (Li et al., 2018). NAVER-Seoul has 3-month speed data collected from 774 links in Seoul, Korea. As NAVER-Seoul data is consisted of main arterial roads in Seoul, it can be considered as a more difficult data with many abruptly changing speed patterns, compared to METR-LA data. Both datasets have five-minute interval speed and timestamp data. Before training PM-MemNet, we fill out missing values using historical data and apply z-score normalization. We use 70% of data for training, 10% for validation, and the rest for evaluation, as Li et al. (2018); Wu et al. (2019); Zheng et al. (2020) have done in their work.

4.1 EXPERIMENTAL SETUP

In our experiment, we use 18 sequence data points as a model input ($T' = 18$, one and half hours) and predict the next 18 sequences. For the k-NN, we utilize cosine similarity function for the similarity measurement. To extract $\mathbb{P}$, we utilize the training dataset and initialize parameter and embedding using Xavier initialization. After performing a greedy search among $d_h = [16, 32, 64, 128]$, $L = [1, 2, 3, 4]$, and $|\mathbb{P}| = [500, 1000, 10000]$, we set d_h as 128, L as 3, and $|\mathbb{P}|$ as 1000. We apply Adam optimizer with a learning rate of 0.001 and use mean absolute error (MAE) as a loss function.

We compare PM-MemNet to the following baseline models: (1) Graph Convolution Recurrent Neural Network (GCRNN); (2) DCRNN (Li et al., 2018), a sequence-to-sequence model that combines diffusion convolution in the gated recurrent unit; (3) Graph-WaveNet (GWNet) (Wu et al., 2019), which forecasts multiple steps at once by integrating graph convolution and dilated convolution; and (4) GMAN (Zheng et al., 2020), Graph Multi-Attention Network, which integrates spatial attention and temporal attention with gated fusion mechanism. GMAN also predicts multiple steps at once. To allow detailed comparisons, we train the baseline models to forecast for the next 90 minutes of speeds with 5-minute intervals, given 18 past 5-minute interval speed data points. We train the baseline models in the equivalent environment with PyTorch[1], using the public source codes and settings provided by the authors. With respect to GWNet, we stack two more blocks since it provides better learning conditions to the model that aggregates longer time series with CNNs.

4.2 EXPERIMENTAL RESULTS

Table 1 displays the experimental results of NAVER-Seoul and METR-LA for the next 15 minutes, 30 minutes, 60 minutes, and 90 minutes, using mean absolute error (MAE), mean absolute percent-

[1] https://pytorch.org

Table 1: Experiment Results

Dataset	T	Metric	HA	GCRNN	DCRNN	GWNet	GMAN	PM-MemNet
NAVER-Seoul	15min	MAE	6.54	4.87	4.86	4.91	5.40	**4.57**
		MAPE	18.24	15.23	15.35	14.86	17.39	**14.48**
		RMSE	9.32	7.18	7.12	7.24	8.58	**6.71**
	30min	MAE	7.16	5.73	5.67	5.26	5.49	**5.06**
		MAPE	20.15	18.17	18.38	16.16	17.71	**16.34**
		RMSE	10.18	9.03	8.80	8.13	8.82	**7.92**
	60min	MAE	8.22	6.58	6.40	5.55	5.61	**5.32**
		MAPE	23.37	20.95	21.09	**16.97**	18.09	17.08
		RMSE	11.54	10.58	10.06	8.77	9.07	**8.60**
	90min	MAE	9.24	7.14	6.86	5.87	5.75	**5.47**
		MAPE	26.40	22.86	22.74	17.89	18.63	**17.50**
		RMSE	12.77	11.43	10.69	9.33	9.30	**8.85**
METR-LA	15min	MAE	4.23	2.59	**2.56**	2.72	2.86	2.67
		MAPE	9.76	6.73	**6.67**	7.14	7.67	7.10
		RMSE	7.46	5.12	**5.10**	5.20	5.77	5.29
	30min	MAE	4.80	3.08	**3.01**	3.12	3.14	3.03
		MAPE	11.30	8.72	**8.42**	8.66	8.79	8.49
		RMSE	8.34	6.32	6.29	6.34	6.54	**6.28**
	60min	MAE	5.80	3.74	3.60	3.58	3.48	**3.43**
		MAPE	14.04	11.50	10.73	10.30	10.10	**9.88**
		RMSE	9.86	7.71	7.65	7.53	7.30	**7.27**
	90min	MAE	6.65	4.23	4.06	3.85	3.71	**3.66**
		MAPE	16.37	13.49	12.53	11.39	11.00	**10.69**
		RMSE	10.97	8.79	8.58	8.12	**7.71**	7.76

age error (MAPE), and root mean square error (RMSE). For the detailed results like 45-minute future forecasts, see Table 3 in Appendix. The table shows that PM-MemNet achieves state-of-the-art performance with both datasets. Specifically, PM-MemNet yields the best performance in all intervals with NAVER-Seoul data, while it outperform other models in the long-term prediction (i.e., 60 and 90 minutes) with the METR-LA data.

In both datasets, we find interesting observations. First, RNN-based models perform better than other models in short term periods (i.e., 15 minutes), but they show a weakness with the long-term periods. This problem occurs in sequence-to-sequence RNNs due to error accumulation caused by its auto-regressive property. Compared to the RNN-based models, PM-MemNet shows a less performance decrease although its decoder is also based on RNN architecture. This is because of GCMem and pattern data which makes representations for traffic forecasting more robust than other manners. In the case of NAVER-Seoul, we observe that all models suffer from decreased accuracy due to the more complicated urban traffic patterns and road networks than those in METR-LA. In spite of these difficulties, PM-MemNet proves its efficiency with representative traffic patterns and memorization technique. The result strengthens our motivation that the traffic data can be generalized with a small number of pattern, used for traffic forecasting, even when the traffic data is complex and hard to forecast.

4.3 Ablation Study

We further evaluate our approach by conducting ablation studies (Table 2, Table 4 in Appendix). First, we check whether GCMem can effectively model spatio-temporal relationships among road networks. To this end, we compare the performance of PM-MemNet to that of SimpleMem, a simplified version of PM-MemNet in which the memory layer only depends on pattern-level attention and does not consider any graph-based relationship. Comparing PM-MemNet and SimpleMem, we observe that SimpleMem has almost 10% decreased performance, which shows the importance of a graph structure in modeling traffic data. In the same context, according to Table 1 and Table 2, SimpleMem has lower accuracy than previous models like DCRNN, which considers a graph structure.

Next, we combine GCMem with various decoder architectures to investigate how the performance of GCMem changes with different combined decoders. The results indicate that all decoder com-

Table 2: Ablation study result.

Dataset	T	Metric	PM-MemNet	SimpleMem	CNN Decoder	RNN Decoder	PM-MemNet w/ L=1
NAVER-Seoul	15min	MAE	4.57	5.72	**4.56**	4.67	4.72
		MAPE	14.48	18.18	**14.40**	14.84	14.98
		RMSE	6.71	8.79	**6.71**	6.83	6.87
	30min	MAE	**5.06**	5.84	**5.06**	5.19	5.22
		MAPE	**16.34**	18.86	16.36	16.87	16.97
		RMSE	7.92	9.24	**7.90**	8.02	8.03
	60min	MAE	**5.32**	6.38	**5.32**	5.47	5.52
		MAPE	**17.08**	21.42	17.19	17.91	17.97
		RMSE	8.60	10.08	**8.51**	8.69	8.70
	90min	MAE	**5.47**	6.95	5.55	5.70	5.72
		MAPE	**17.50**	23.89	17.99	18.73	18.63
		RMSE	8.85	10.88	**8.82**	9.10	9.05
METR-LA	15min	MAE	2.67	3.01	**2.63**	2.68	2.68
		MAPE	7.10	8.03	**6.98**	7.10	7.11
		RMSE	**5.29**	5.94	5.32	5.31	5.31
	30min	MAE	3.03	3.27	**3.01**	3.06	3.06
		MAPE	8.49	9.20	**8.46**	8.56	8.59
		RMSE	6.28	6.68	6.36	6.32	**6.27**
	60min	MAE	3.43	3.72	**3.41**	3.46	3.47
		MAPE	**9.88**	10.94	**9.88**	10.02	10.07
		RMSE	7.27	7.70	7.28	7.31	**7.25**
	90min	MAE	3.66	4.09	**3.65**	3.71	3.73
		MAPE	**10.69**	12.25	10.85	10.87	10.98
		RMSE	7.76	8.38	**7.71**	7.81	7.75

binations achieve state-of-the-art performance with both NAVER-Seoul and METR-LA. The result implies that GCMem effectively learns representations even with a simple architecture as decoder, such as one-layer RNN. Also, in the perspective of error accumulation, we see that PM-MemNet outperforms the CNN decoder even in the 90-minute prediction. In contrast, the one-layered RNN decoder has lower performance than that of the CNN decoder. However, when comparing the RNN decoder to the existing RNN-based models, we can say that our encoder generates a representation that can represent information for the long-term prediction. This results also show that GCMem is a robust solution for the long-term dependency modeling. Modifying the GCMem layer depth, we discover that PM-MemNet generates sufficiently accurate predictions only with one layer memory (i.e., PM-MemNet w/L=1). Although PM-MemNet needs a three-layered GCMem for the highest accuracy, we can still consider deploying the model with a light-weighted version, while ensuring its state-of-the-art performance. We provide time consumption of the models in Appendix, Table 5.

4.4 Qualitative Evaluation

We present qualitative analysis using the NAVER-Seoul dataset which contains a complex road network and dynamic traffic conditions of urban areas. In the analysis, we evaluate whether our approach effectively patternize traffic data for traffic forecasting problem. If the approach is valid, we expect PM-MemNet is more accurate in predicting difficult trailing patterns due to high correlation between leading and trailing patterns. To verify our expectation, we visualize traffic prediction results in long-term, as shown in Figure 4 (top), where PM-MemNet predicts the end of congestion in 12:00 (B) more accurately. Also, in 7:00 (A), PM-MemNet follows up the unexpected peak speed (it continues about 30 minutes, and PM-MemNet efficiently catches them up within 15 minutes). From Figure 4 (bottom), we see that PM-MemNet accurately predicts a speed drop and its recovery than other models. For example, GMAN and GWNet predict that speed drop will occur faster than real speed drop, and DCRNN predicts speed drop will occur later than real speed drop, as shown in (C). But PM-MemNet predicts the occurrence of slowdowns on time, even when it predicts long-term traffic conditions (D). Overall, with representative traffic patterns and memorization, PM-MemNet effectively handles abruptly changing traffic conditions even in long-term prediction.

5 Limitations, Discussion, and Future Directions

This work is the first attempt to design neural memory networks for traffic forecasting. Although effective as shown in the evaluation results, there are limitations we spot in the approach. First, we extract traffic patterns to memorize in advance, but it is possible that the extracted patterns are re-

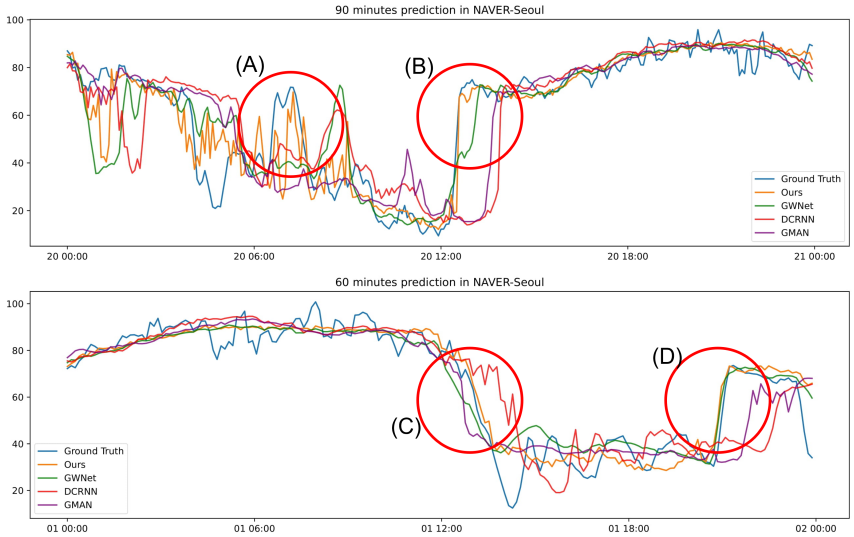

Figure 4: NAVER-Seoul speed prediction visualization for (upper) 90-minute forecasting and (lower) 60-minute forecasting. Red circles show the abrupt speed changing that successfully captured by PM-MemNet

dundant even after strict filtering or may not be used. As such there is a need for finding important patterns and optimizing the number of memory slots. For example, a future study may investigate how to learn and extend the key space during the training phase (Kaiser et al., 2017). Second, the learning of PM-MemNet only proceeds in referred patterns. Because there are no further losses to optimize memory itself, patterns not referred are not trained. This training imbalance among memories is of interest, as a model could not generate meaningful representations from rare patterns. A future study may research not only how such representation imbalance affects the performance, but also design a loss function to reduce the representation gap between rare and frequent events. Third, we use cosine similarity in this work, but it may not be an optimal solution since it causes mismatching with noisy traffic data. Also, the optimal window size for the pattern matching is one of the remaining questions. A future study may focus on approaches to effectively compute the similarity of traffic patterns. Designing a learnable function for the computation is one possible direction. Lastly, we show that a model effectively forecasts traffic data with a small group of patterns. It implies a new research direction of comparing results and learning methods that work with sparse data, such as meta learning and few or zero shot learning (Kaiser et al., 2017).

6 Conclusion

In this work, we propose PM-MemNet, a novel traffic forecasting model with graph convolutional memory architecture GCMem. By integrating GCNs and neural memory architectures, PM-MemNet effectively captures both spatial and temporal dependency. By extracting and computing representative traffic patterns, we reduce the data space into a small group of patterns. The experiment results with METR-LA and NAVER-Seoul indicate that PM-MemNet outperform state-of-the-art models, proving our perspective is reasonable–accurate traffic forecasting can be achieved with a small set of representative patterns. We also demonstrate that PM-MemNet quickly responds to abruptly changing traffic patterns and achieves high accuracy compared to other models. Lastly, we discuss limitations of this work and future research directions with neural memory networks and a small set of patterns for traffic forecasting. As future work, we plan to investigate if PM-MemNet can be applied for different spatio-temporal domains and the insights and lessons from this work work can be generalized to other domains.

REFERENCES

Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. In *International Conference on Learning Representations*, 2014.

Weiwei Jiang and Jiayun Luo. Graph neural network for traffic forecasting: A survey. *CoRR*, abs/2101.11174, 2021. URL `https://arxiv.org/abs/2101.11174`.

Lukasz Kaiser, Ofir Nachum, Aurko Roy, and Samy Bengio. Learning to remember rare events. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017.

C. Lee, Y. Kim, S. Jin, D. Kim, R. Maciejewski, D. Ebert, and S. Ko. A visual analytics system for exploring, monitoring, and forecasting road traffic congestion. *IEEE Transactions on Visualization and Computer Graphics*, 26(11):3133–3146, 2020.

Yaguang Li and Cyrus Shahabi. A brief overview of machine learning methods for short-term traffic forecasting and future directions. *SIGSPATIAL Special*, 10(1):3–9, 2018.

Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In *International Conference on Learning Representations*, 2018.

Wei-Chao Lin, Chih-Fong Tsai, Ya-Han Hu, and Jing-Shang Jhang. Clustering-based undersampling in class-imbalanced data. *Information Sciences*, 409-410:17–26, 2017.

Andrea Madotto, Chien-Sheng Wu, and Pascale Fung. Mem2seq: Effectively incorporating knowledge bases into end-to-end task-oriented dialog systems. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pp. 1468–1478, 2018.

Cheonbok Park, Chunggi Lee, Hyojin Bahng, Yunwon Tae, Seungmin Jin, Kihwan Kim, Sungahn Ko, and Jaegul Choo. ST-GRAT: A novel spatio-temporal graph attention networks for accurately forecasting dynamically changing road speed. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*, pp. 1215–1224. ACM, 2020.

Lei Shi, Yifan Zhang, Jian Cheng, and Hanqing Lu. Two-stream adaptive graph convolutional networks for skeleton-based action recognition. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pp. 12026–12035, 2019.

Hava T. Siegelmann and Eduardo D. Sontag. Turing computability with neural nets. *Applied Mathematics Letters*, 4(6):77–80, 1991.

Sainbayar Sukhbaatar, arthur szlam, Jason Weston, and Rob Fergus. End-to-end memory networks. In *Advances in Neural Information Processing Systems*, volume 28, 2015.

Eleni I. Vlahogianni, Matthew G. Karlaftis, and John C. Golias. Short-term traffic forecasting: Where we are and where we're going. *Transportation Research Part C: Emerging Technologies*, 43:3–19, 2014. Special Issue on Short-term Traffic Flow Forecasting.

Jason Weston, Sumit Chopra, and Antoine Bordes. Memory networks. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, and Chengqi Zhang. Graph wavenet for deep spatial-temporal graph modeling. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pp. 1907–1913, 2019.

Peng Xie, Tianrui Li, Jia Liu, Shengdong Du, Xin Yang, and Junbo Zhang. Urban flow prediction from spatiotemporal data using machine learning: A survey. *Information Fusion*, 59:1–12, 2020.

Bing Yu, Haoteng Yin, and Zhanxing Zhu. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pp. 3634–3640, 2018.

Junbo Zhang, Yu Zheng, Dekang Qi, Ruiyuan Li, and Xiuwen Yi. Dnn-based prediction model for spatio-temporal data. In *Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, SIGSPACIAL '16, 2016.

Chuanpan Zheng, Xiaoliang Fan, Cheng Wang, and Jianzhong Qi. GMAN: A graph multi-attention network for traffic prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 1234–1241, 2020.

A APPENDIX

Table 3: Detailed experiment results with state-of-the-art models.

Dataset	T	Metric	HA	GCRNN	DCRNN	GWNet	GMAN	PM-MemNet
NAVER-Seoul	15min	MAE	6.54	4.87	4.86	4.91	5.40	**4.57**
		MAPE	18.24	15.23	15.35	14.86	17.39	**14.48**
		RMSE	9.32	7.18	7.12	7.24	8.58	**6.71**
	30min	MAE	7.16	5.73	5.67	5.26	5.49	**5.06**
		MAPE	20.15	18.17	18.38	16.16	17.71	**16.34**
		RMSE	10.18	9.03	8.80	8.13	8.82	**7.92**
	45min	MAE	7.70	6.24	6.12	5.43	5.55	**5.23**
		MAPE	21.81	19.85	20.06	**16.70**	17.90	16.87
		RMSE	10.89	9.99	9.61	8.53	8.97	**8.39**
	60min	MAE	8.22	6.58	6.40	5.55	5.61	**5.32**
		MAPE	23.37	20.95	21.09	**16.97**	18.09	17.08
		RMSE	11.54	10.58	10.06	8.77	9.07	**8.60**
	75min	MAE	8.73	6.87	6.63	5.68	5.67	**5.37**
		MAPE	24.91	21.91	21.93	17.31	18.34	**17.23**
		RMSE	12.17	11.02	10.39	9.01	9.19	**8.71**
	90min	MAE	9.24	7.14	6.86	5.87	5.75	**5.47**
		MAPE	26.40	22.86	22.74	17.89	18.63	**17.50**
		RMSE	12.77	11.43	10.69	9.33	9.30	**8.85**
METR-LA	15min	MAE	4.23	2.59	**2.56**	2.72	2.86	2.67
		MAPE	9.76	6.73	**6.67**	7.14	7.67	7.10
		RMSE	7.46	5.12	**5.10**	5.20	5.77	5.29
	30min	MAE	4.80	3.08	**3.01**	3.12	3.14	3.03
		MAPE	11.30	8.72	**8.42**	8.66	8.79	8.49
		RMSE	8.34	6.32	6.29	6.34	6.54	**6.28**
	45min	MAE	5.32	3.46	3.34	3.39	3.34	3.27
		MAPE	12.71	10.26	9.71	9.63	9.55	9.33
		RMSE	9.18	7.12	7.08	7.06	7.00	6.90
	60min	MAE	5.80	3.74	3.60	3.58	3.48	**3.43**
		MAPE	14.04	11.50	10.73	10.30	10.10	**9.88**
		RMSE	9.86	7.71	7.65	7.53	7.30	**7.27**
	75min	MAE	6.25	4.01	3.84	3.73	3.60	3.55
		MAPE	15.26	12.59	11.68	10.90	10.56	10.31
		RMSE	10.46	8.21	8.15	7.87	7.52	7.53
	90min	MAE	6.65	4.23	4.06	3.85	3.71	**3.66**
		MAPE	16.37	13.49	12.53	11.39	11.00	**10.69**
		RMSE	10.97	8.79	8.58	8.12	**7.71**	7.76

Table 4: Detailed ablation study result.

Dataset	T	Metric	PM-MemNet	SimpleMem	CNN Decoder	RNN Decoder	PM-MemNet w/ L=1
NAVER-Seoul	15min	MAE	4.57	5.72	4.56	4.67	4.72
		MAPE	14.48	18.18	14.40	14.84	14.98
		RMSE	6.71	8.79	6.71	6.83	6.87
	30min	MAE	5.06	5.84	5.06	5.19	5.22
		MAPE	16.34	18.86	16.36	16.87	16.97
		RMSE	7.92	9.24	7.90	8.02	8.03
	45min	MAE	5.23	6.10	5.23	5.38	5.43
		MAPE	16.87	19.98	16.98	17.61	17.66
		RMSE	8.39	9.71	8.32	8.47	8.49
	60min	MAE	5.32	6.38	5.32	5.47	5.52
		MAPE	17.08	21.42	17.19	17.91	17.97
		RMSE	8.60	10.08	8.51	8.69	8.70
	75min	MAE	5.37	6.65	5.39	5.56	5.60
		MAPE	17.23	22.52	17.38	18.20	18.23
		RMSE	8.71	10.48	8.62	8.86	8.85
	90min	MAE	5.47	6.95	5.55	5.70	5.72
		MAPE	17.50	23.89	17.99	18.73	18.63
		RMSE	8.85	10.88	8.82	9.10	9.05
METR-LA	15min	MAE	2.67	3.01	**2.63**	2.68	2.68
		MAPE	7.10	8.03	**6.98**	7.10	7.11
		RMSE	**5.29**	5.94	5.32	5.31	5.31
	30min	MAE	3.03	3.27	**3.01**	3.06	3.06
		MAPE	8.49	9.20	**8.46**	8.56	8.59
		RMSE	6.28	6.68	6.36	6.32	**6.27**
	45min	MAE	3.27	3.52	3.25	3.30	3.30
		MAPE	9.33	10.18	9.30	9.44	9.47
		RMSE	6.90	7.28	6.94	6.93	6.86
	60min	MAE	3.43	3.72	**3.41**	3.46	3.47
		MAPE	**9.88**	10.94	**9.88**	10.02	10.07
		RMSE	7.27	7.70	7.28	7.31	**7.25**
	75min	MAE	3.55	3.91	3.52	3.59	3.60
		MAPE	10.31	11.61	10.35	10.47	10.56
		RMSE	7.53	8.06	7.50	7.59	7.53
	90min	MAE	3.66	4.09	**3.65**	3.71	3.73
		MAPE	**10.69**	12.25	10.85	10.87	10.98
		RMSE	7.76	8.38	**7.71**	7.81	7.75

Table 5: The computation times for each model in METR-LA

Computation Time	DCRNN	GWNet	GMAN	PM-MemNet	SimpleMem	CNN Decoder	RNN Decoder	PM-MemNet w/ L = 1
Training time per epoch (sec)	691.32	102.06	500.19	196.6	169.6	104.78	39.94	127.53
Inference Time (sec)	56.30	6.00	9.34	14.9	12.14	10.2	4.26	10.77