Pattern Sensitive Prediction of Traffic Flow Based on Generative Adversarial Framework

Yilun Lin, Xingyuan Dai, Li Li, *Fellow, IEEE*, and Fei-Yue Wang, *Fellow, IEEE*

***Abstract*— Traffic flow prediction is one of the most popular topics in the field of the intelligent transportation system due to its importance. Powered by advanced machine learning techniques, especially the deep learning method, prediction accuracy noticeably increases in recent years. However, most existing methods applied a data-driven paradigm and tend to ignore the outliers, which result in poor performance while handling burst phenomena in the traffic system. To overcome this problem, the prediction model needs to recognize different patterns and handle them in different ways. In this paper, we propose a new prediction model (called pattern sensitive network) that can handle different traffic patterns automatically. By using adversarial training, our model can make more accurate predictions in unusual states without compromising its performance in usual states. Experiments demonstrate that our method can work well in both usual traffic states and unusual traffic states.**

***Index Terms*— Traffic flow prediction, deep learning, generative adversarial network.**

I. Introduction

Traffic flow prediction plays a key role in many traffic applications [1]–[4], e.g., signal optimization [5], [6], route planning [7]. Using some powerful models, such as **Autoregressive moving-average model (ARMA)** [8], [9], **Support Vector Regression (SVR)** [10], [11], and especially the **Deep Learning (DL)** [12], the prediction accuracy for real-world application has already been significantly increased.

Deep Learning methods have achieved the state-of-art performance on traffic flow prediction problem [13]–[15]. Such methods combine data-driven paradigm with representation learning, allowing machine learning model to automatically discover the patterns within the data [16]. However, unlike other missions deep learning methods are usually applied to, such as computational vision or speech recognition, the traffic prediction problem has a different nature. One of its most significant characteristics is the violent change of traffic flows. Though such bursts rarely occur, they may lead to noticeable prediction errors or economics cost [17], [18]. Applying deep neural network directly without considering the diverse patterns of traffic system leads to unsatisfactory performances in extreme situations, as shown in the experiments of this paper, though its overall score is still one of the best. A further analysis on the origin of this problem reveals that many conventional data-driven prediction methods do not fully consider the training objective for traffic prediction problems. Since the occurrence of unusual situations is rare, the prediction errors on the associated data points contribute relatively little to the total prediction errors. That makes prediction models easy to remember the patterns of usual data but not the patterns of unusual data. As a result, conventional prediction methods turn to ignore these unusual traffic states implicitly, thereby a preprocessing to categorize the data will be needed. Some previous works address this problem by training separate models dedicated for different traffic states [19]. Such methods require us to label the collected traffic flow data, so the model can tell the difference between the usual and unusual cases. It can achieve good performance only if we know which kind of model should be used and have enough sources to label the data.

Instead of supervised methods, we proposed a novel model called **Pattern-Sensitive Networks (PSNs)** to solve this problem by unsupervised learning. PSNs are first trained to learn the joint distribution of the historical-future data pairs by applying an adversarial learning paradigm [20], [21], then fine-tuned to better describe the precise conditional distribution (causal relation) between historical and future data. In such way, PSNs can automatically distinguish between the data of different cases, and use the learned diverse representations for better prediction. Experiments show that this kind of networks performs comparably or even better than some state-of-art approaches, especially for unusual states when burst phenomena occur.

To further explain our approach, the rest of this paper is arranged as follows. *Section II* introduces the conventional methodology of traffic flow prediction, then points out its hidden problem. *Section III* proposes the principle of a solution along with the structure and training scheme of PSN. *Section IV* provides some numerical results to verify the effectiveness of the proposed method. Finally, *Section V* concludes the paper.

II. Traffic Prediction Methods

A. Conventional Traffic Prediction Method

The core of conventional traffic flow prediction methods is to appropriately establish a special (usually learnable) mapping relation that links historical records (including traffic flow data and other traffic information) and future traffic flow data [17], [22]. Along this paper, we use the notation s_t to represent the true traffic flow at time step t, and $s_{0:N}$ to represent the sequence of $N+1$ historical records $\langle s_0, s_1, \ldots, s_N \rangle$. Moreover, we use $s_{0:t-1}$ to indicate the historical records with a fixed length before s_t for simplicity.

This prediction problem can be viewed from a probabilistic perspective. We can view the traffic flow at time step t as a random variable depends on the historical data, and the conditional probability $p\left(s_t|s_{0:t-1}\right)$ obeys a certain distribution. A probabilistic inference method can be used to find this unknown distribution. The conventional way is to set up a learnable model $q\left(s_t|s_{0:t-1}\right)$ that is controlled by one or some parameters θ, then minimizes the difference between the learnable model $q\left(s_t|s_{0:t-1};\theta\right)$ and true conditional distribution $p\left(s_t|s_{0:t-1}\right)$ [23].

Manuscript received December 9, 2017; revised May 17, 2018 and July 11, 2018; accepted July 14, 2018. Date of publication August 17, 2018; date of current version May 29, 2019. This work was supported by the National Natural Science Foundation of China under Grant 61533019 and Grant 71232006. The Associate Editor for this paper was S. C. Wong. *(Corresponding author: Li Li.)*

Y. Lin and X. Dai are with the State Key Laboratory for Management and Control of Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100080, China, with University of Chinese Academy of Sciences, Beijing 100049, China, and also with Qingdao Academy of Intelligent Industries, Qingdao 266109, China.

L. Li is with the Tsinghua National Laboratory of Information Science and Technology, Department of Automation, Tsinghua University, Beijing 100084, China, and also with the Qingdao Academy of Intelligent Industries, Qingdao 266109, China (e-mail: li-li@tsinghua.edu.cn).

F.-Y. Wang is with the State Key Laboratory for Management and Control of Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100080, China, and also with the Qingdao Academy of Intelligent Industries, Qingdao 266109, China.

Digital Object Identifier 10.1109/TITS.2018.2857224

The difference between two distribution is usually measured by Kullback-Leibler (KL) divergence [24]:

$$KL\left(p\left(s_t|s_{0:t-1}\right) \parallel q\left(s_t|s_{0:t-1};\theta\right)\right) = \sum_S p\left(s_t|s_{0:t-1}\right)\log\frac{p\left(s_t|s_{0:t-1}\right)}{q\left(s_t|s_{0:t-1};\theta\right)} \quad (1)$$

where S is the set of all possible values of s_t

Using Kullback-Leibler divergence, the best parameters should satisfy

$$\theta^* = \arg\min_\theta KL\left(p\left(s_t|s_{0:t-1}\right) \parallel q\left(s_t|s_{0:t-1};\theta\right)\right) \quad (2)$$

Then a gradient based optimization method can be applied to calculate θ^*.

Let us use $\tilde{s}_t$ to represent the predictive future traffic flow given by a conditional parametric model. Once the model has been fitted to the training data, the prediction $\tilde{s}_t^*$ can then be drawn from conditional distribution in either a stochastic or deterministic way:

$$\tilde{s}_t^* \sim q\left(s_t|s_{0:t-1};\theta^*\right) \quad \text{or } \tilde{s}_t^* = \arg\max_{\tilde{s}_t} q\left(s_t|s_{0:t-1};\theta^*\right) \quad (3)$$

Using a complex model, such as deep neural network as the parametric model $q\left(s_t|s_{0:t-1}\right)$, the conventional prediction methods can achieve high accuracy in general.

B. The Bursts in Traffic System

Traditionally, data-driven methods tend to neglect the outliers in order to prevent overfitting. The outlier refers to the observation that is distant from other observations. In the context of the sequence prediction problem, it means that, for similar historical records $s_{0:t-1}$ and $s'_{0:t-1}$, the conditional probabilities of following values in unusual cases s'_t are relatively low. The relation between the usual cases and unusual cases can be formalized as:

$$p\left(s_{0:t-1}\right) \simeq p\left(s'_{0:t-1}\right), \quad p\left(s_t|s_{0:t-1}\right) > p\left(s'_t|s'_{0:t-1}\right) \quad (4)$$

The KL-divergences in usual cases are most likely larger than those in unusual cases due to $p(s_t|\cdot) > p(s'_t|\cdot)$, which makes the gradient-based methods tend to ignore the outliers since $\nabla\theta_s > \nabla\theta_{s'}$.

The extreme conditions in traffic dynamic are treated as outliers in most of the previous works, since the prediction error in extreme conditions is minor under some statistical criteria, such as mean square error and mean absolute percentage error. For example, if we categorize the observed traffic states in PeMS dataset [25] into two cases: the usual case and the unusual case, and define the unusual case as follow:

$$\frac{\left|s_t - s_{t-1}\right|}{s_{t-1}} \geq 1, \quad s_t > 5 \quad (5)$$

Then there is only 2% of the records satisfy such definition.

However, such mispredictions may result in traffic congestion and economic loss [19], [26], and it implies a mismodeling of traffic dynamic [27]. Therefore, an accurate forecasting of extreme conditions is urgently needed for traffic control.

III. Pattern Sensitive Networks

A. Pattern Sensitive Prediction

As above analysis has shown, the data-driven methods are designed to neglect the unusual patterns. To let the model pay more attention to the unusual patterns, one alternative solution is to use the KL-divergence of joint distribution instead of conditional distribution. We prove its effectiveness in the following chapters.

Similar to the conventional method, the learning process of parametric model is to minimize the KL-divergence between its model distribution $q\left(s_{0:t-1}, s_{:t};\theta\right)$ and true distribution $p\left(s_{0:t-1}, s_t\right)$:

$$\begin{aligned} &KL\left(p\left(s_{0:t-1}, s_t\right) \parallel q\left(s_{0:t-1}, s_t\right)\right) \\ &= \sum_S p\left(s_{0:t-1}, s_t\right)\log\frac{p\left(s_{0:t-1}, s_t\right)}{q\left(s_{0:t-1}, s_t\right)} \\ &= \sum_S p\left(s_t|s_{0:t-1}\right)p\left(s_{0:t-1}\right)\log\frac{p\left(s_t|s_{0:t-1}\right)p\left(s_{0:t-1}\right)}{q\left(s_t|s_{0:t-1}\right)q\left(s_{0:t-1}\right)} \end{aligned} \quad (6)$$

Notice that $p\left(s_{0:t-1}\right)$ does not depend on $p\left(s_t\right)$ and $p\left(s_{0:t-1}, s_t\right)$, combining (6) with (1), we have

$$KL\left(p\left(s_{0:t-1}, s_t\right) \parallel q\left(s_{0:t-1}, s_t\right)\right) = p\left(s_{0:t-1}\right)KL\left(p\left(s_t|\cdot\right) \parallel q\left(s_t|\cdot\right)\right) + \log\frac{p\left(s_{0:t-1}\right)}{q\left(s_{0:t-1}\right)} \quad (7)$$

Since $p\left(s_{0:t-1}\right) \in [0, 1]$, and $\log\frac{p(s_{0:t-1})}{p(s'_{0:t-1})}\frac{q(s_{0:t-1})}{q(s'_{0:t-1})}$ approaches the limit zero while distribution q getting closer to p, we can further infer that

$$\begin{aligned} &KL\left(p\left(s_{0:t-1}, s_t\right) \parallel q\left(s_{0:t-1}, s_t\right)\right) - KL\left(p\left(s'_{:t}\right) \parallel q\left(s'_{:t}\right)\right) \\ &\leq KL\left(p\left(s_t|\cdot\right) \parallel q\left(s_t|\cdot\right)\right) - KL\left(p\left(s'_t|\cdot\right) \parallel q\left(s'_t|\cdot\right)\right) \\ &\quad + \left(\log\frac{p\left(s_{0:t-1}\right)}{q\left(s_{0:t-1}\right)} - \log\frac{p\left(s'_{0:t-1}\right)}{q\left(s'_{0:t-1}\right)}\right) \\ &\simeq KL\left(p\left(s_t|\cdot\right) \parallel q\left(s_t|\cdot\right)\right) - KL\left(p\left(s'_t|\cdot\right) \parallel q\left(s'_t|\cdot\right)\right) \end{aligned} \quad (8)$$

Inequation (8) indicates that, by learning the joint distribution instead of conditional distribution, we can reduce the gap between KL-divergences under different cases, and let the model pay more attention to the unusual patterns. Since such prediction method is more sensitive to the different patterns, we call it **Pattern-Sensitive Prediction (PSP)**.

B. Pattern-Sensitive Networks

For the traffic prediction problem, deep neural network exhibits the highest capability to fit the records than previous models [14]. Moreover, its flexibility allows multimodal data fusion to be performed, which boosts its performance even further [28]–[32].

However, the neural network is usually modeled as a conditional distribution instead of a joint distribution. Since the neural network performs a deterministic mapping process, there has to be an input in order to get a meaningful output. A recently developed approach, called Generative adversarial network (GAN) is one of the exceptions. GAN is in the family of implicit density models, which can be trained while interacting only indirectly with the data distribution by sampling from it [20].

The basic idea of GAN is to set up an adversary game between two differentiable networks, call generator and discriminator. The generator maps a source of noise $z \sim p(z)$ to the input space. The discriminator receives either a generated sample or a true data sample and tries to distinguish between them. Hopefully, this competition will converge to an equilibrium while generative samples are indistinguishable from the discriminator. That indicates the generator is an approximation of the real samples distribution.

By introducing GAN into the prediction model, we are able to perform a pattern sensitive prediction using neural networks. A novel structure, called **Pattern Sensitive Networks (PSNs)** is proposed by combining the adversary structure with auxiliary losses. This model can first infer the joint distribution of historical record and

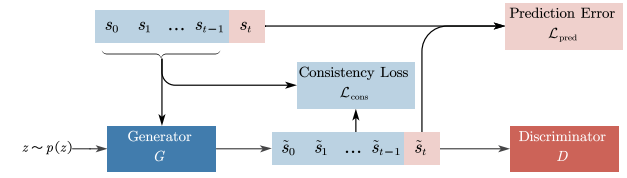

Fig. 1. Schematic of PSN architecture.

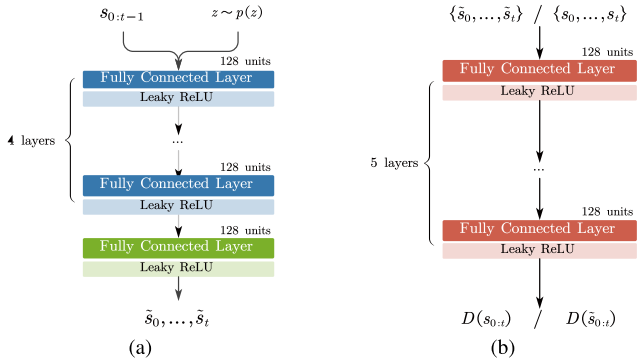

Fig. 2. The implementation of PSN used in this paper. (a) Generator. (b) Discriminator.

future state, therefore automatically discovers the unusual patterns and categories the data into different classes by providing diverse representations. Then the representation it learned from the previous stage can be used for better prediction. The structure of this network is illustrated in Fig. 1.

As illustrated in the figure, the main components of PSN are two adversary networks, in which a generator receives both random noises and historical records $s_{0:t}$. The generator outputs a sequence of values $\tilde{s}_{0:t}$ and is optimized according to three criteria.

First is the discriminative criterion $D(G(z, s_{0:t-1}))$, which is provided by the discriminator to indicate if the generative sequences can be told from the true ones. The consistency loss $\mathcal{L}_{\text{cons}}$ is used to decrease the reconstruction error of the historical record, i.e., $\tilde{s}_{0:t-1} \approx s_{0:t-1}$. And the last criterion is the prediction error $\mathcal{L}_{\text{pred}}$, which measures the prediction accuracy. The total objective function of the generator can then be formed as:

$$\mathcal{L}_G = c_1 \cdot D\left(G\left(z, s_{0:t-1}\right)\right) + c_2 \cdot \mathcal{L}_{\text{cons}} + c_3 \cdot \mathcal{L}_{\text{pred}} \tag{9}$$

where c_i is the coefficient of different losses.

C. Implementation of PSN

In this paper, we implement a pattern sensitive network to predict the traffic flow of freeway system across all major metropolitan areas in California, US.

Our implementation adapts a variant of GAN called Wasserstein GAN with gradient penalty (WGAN-GP) [33], containing two networks: generator G with weights θ, and discriminator D with weight w. Compared with original GAN, this variant optimizes Earth-Mover's distance instead of Jensen-Shannon divergence.

As shown in Fig. 2b, the discriminator receives either true or generated samples and then outputs a scalar. The goal of discriminator is to maximize the expected difference between $D(s_{0:t})$ and $D(\tilde{s}_{0:t})$. To guarantee the discriminator to be optimal, a gradient penalty, $(\parallel \nabla_{\hat{s}} D_w(\hat{s}) \parallel_2 -1)^2$ will also be applied. $\hat{s}$ is a random mix of true and generative samples

$$\hat{s} = \epsilon s_{0:t} + (1-\epsilon)\tilde{s}_{0:t} \tag{10}$$

where ϵ is a random variable sampled uniformly from 0 to 1.

The objective function of discriminator can then be formed as:

$$\mathcal{L}_D = D(\tilde{s}_{0:t}) - D(s_{0:t}) + \lambda(\parallel \nabla_{\hat{x}} D_w(\hat{x}) \parallel_2 -1)^2 \tag{11}$$

Algorithm 1 Pretraining Stage of PSN. We Use Default Values of $\lambda = 10, n_{disc} = 5$

Require: Traffic records collected from different stations; Random distribution $p(z)$; Preprocessing time steps T_1; Fine-tuning time steps T_2;

Initialize θ, w.
while θ has not converged **do**
  for $t = 1$ to n_{disc} **do**
    for $i = 1$ to m **do**
      sample real record $s^{(i)}_{0:t-1}$ from random station, latent variable $z \sim p(z)$, a random number $\epsilon \sim U[0, 1]$.
      $\tilde{s}^{(i)}_{0:t} \leftarrow G(s^{(i)}_{0:t-1}, z)$, $\hat{s}^{(i)}_{0:t} \leftarrow \epsilon s^{(i)}_{0:t} + (1-\epsilon)\tilde{s}^{(i)}_{0:t}$
      calculate $\mathcal{L}^{(i)}_D$ according to (11)
    end for
    $w \leftarrow w + \nabla_w \frac{1}{m} \sum_{i=0}^{m} \mathcal{L}^{(i)}_D$
  end for
  for $i = 1$ to m **do**
    sample real record $s^{(i)}_{0:t-1}$ from random station, latent variable $z \sim p(z)$
    $\tilde{s}^{(i)}_{0:t} \leftarrow G(s^{(i)}_{0:t-1}, z)$
    calculate $\mathcal{L}^{(i)}_G$ according to (12)
  end for
  $\theta \leftarrow \theta + \nabla_\theta \frac{1}{m} \sum_{i=0}^{m} \mathcal{L}^{(i)}_G$
end while
return Generator network G_θ

where λ is the coefficient of gradient penalty.

The objective function of the generator has three components, as introduced in the previous section and Equation (9). The discriminative criterion is $-D(\tilde{s}_{0:t})$ under the WGAN-GP framework. The prediction error and consistency loss are defined as mean square errors in this paper. Therefore, the objective function of the generator is

$$\mathcal{L}_G = -c_1 \cdot D(\tilde{s}_{0:t}) + c_2 \cdot (s_{0:t-1} - \tilde{s}_{0:t-1})^2 + c_3 \cdot (s_t - \tilde{s}_t)^2 \tag{12}$$

For traffic prediction problems, the training scheme of PSN can be separated into two stages.

The first training stage of PSNs is an unsupervised pretraining process, performing on a mixed data set collected from different stations $S_0, S_1, \ldots, S_N$. The learning process follows the way introduced in literature [33], in which the discriminator will be trained for n_{disc} epochs while the generator for only one epoch. By adversarial training upon mixing dataset obtained in different lanes, the generator can fit the joint distribution of traffic states and obtain a glimpse of the flow patterns. The pretraining procedure is demonstrated in Algorithm 1.

Then the representation gained from the unsupervised stage can be used to perform supervised tasks, so the generator network can best fit a specific station S_i. The most direct way is to optimize the generator according to the prediction error. This procedure performs a better estimation of the conditional distribution on given records and improves model's prediction accuracy for a specific area.

Both generator and discriminator are built by stacking 5 fully connected layers with 128 units, each following a Leaky version of a Rectified Linear Unit (Leaky ReLU) [34]. The last layer in the generator is always trainable during the whole training process, while the other layers can only be trained during the unsupervised learning stage. This implementation is illustrated in Fig. 2.

IV. Numerical Comparison Results

In this section, we conduct experiments to answer whether pattern sensitive model can better handle unusual cases.

A. Data Sources and Data Preprocessing

The experimental dataset is obtained from Caltrans Performance Measurements Systems (PeMS) [25], which is the most widely used dataset for traffic prediction. In PeMS, the traffic flow data are collected by inductive loop sensors throughout the freeways and then aggregated as counts of cars into 5-min periods. The proposed model is applied to the data in the whole year of 2016, in which the data of the first 9 months are used as the training set and the data of the remaining 3 months are used as the test set. We only focus on 1397 sensors in district 4 which functioned consistently throughout the studied period. The missing data of these sensors are imputed using simple average trend. Due to the limited computational resource, 50 sensors are selected to perform one-to-one prediction (to predict the traffic flow of one station using its own records only).[1]

Detrending is widely used in analyzing traffic flow time series and proved to be critical for neural network methods [17], [35], [36]. We consider the weekly seasonality of traffic flow by using a simple moving average method. All data will be detrended by subtracting the average flow at the same time of last few weeks:

$$\text{trend}_t = \frac{1}{N}\sum_{i=1}^{N} s_{t-i*weeks} \tag{13}$$

We look back four weeks to calculate the trend in this paper, which means $N = 4$. For example, for the data obtained at 10:00 A.M. in the Wednesday of the fifth week, the input flow data will subtract the mean value of the traffic flow at 10:00 A.M. in the Wednesday of the previous four weeks. The input flow data in each sensor are also normalized to be 0 mean and 1 standard deviation depending on the value of training set.

B. Performance Indexes and Models Used for Comparison

We compare different models under two performance indexes, the mean absolute error (MAE) and mean absolute percentage error (MAPE), which are defined as:

$$\text{MAE}\,(s_t, \tilde{s}_t) = \frac{1}{T}\sum_{t=0}^{T} |s - \tilde{s}_t| \tag{14}$$

$$\text{MAPE}\,(s_t, \tilde{s}_t) = \frac{1}{T}\sum_{i=0}^{T} \left|\frac{s_t - \tilde{s}_t}{s_t}\right| \tag{15}$$

where T is the length of the test set. For the MAPE index, we only consider the situation when $s_t > 5$. The standard deviations of prediction error are also considered in our experiment, defined as

$$Std\,(MAE) = \sqrt{\frac{\sum_{i=1}^{M}\left(MAE_i - \overline{MAE}\right)^2}{M-1}} \tag{16}$$

$$Std\,(MAPE) = \sqrt{\frac{\sum_{i=1}^{M}\left(MAPE_i - \overline{MAPE}\right)^2}{M-1}} \tag{17}$$

$M = 50$ is the station numbers we have in the dataset. $\overline{MAE}$ and $\overline{MAPE}$ are the average error on all stations. The performances are calculated after we scale back the data to the original representation and add the trend back to it.

[1]The id of each sensor is: 400000, 400001, 400002, 400006, 400009, 400011, 400014, 400015, 400017, 400025, 400028, 400030, 400031, 400039, 400041, 400043, 400048, 400049, 400050, 400060, 400067, 400069, 400073, 400074, 400078, 400079, 400083, 400085, 400088, 400090, 400091, 400093, 400096, 400098, 400101, 400103, 400105, 400107, 400112, 400113, 400115, 400118, 400119, 400122, 400125, 400126, 400127, 400132, 400137, 400141.

C. Compared Methods

To demonstrate the improvement brought by GAN technique, we compare our implementation of PSN, which is described in section III-C with some previous state-of-art methods, including Autoregressive Moving Average Models (ARMA) [37], Support Vector Regression (SVR) [10], [38], Scalable End-to-End Tree Boosting System (XGBoost) [39], Stack Auto Encoders (SAE) [14] and Long Short Term Memory Network (LSTM) [40]–[42].

For the non-neural-network-based models, we adopt the default settings of hyperparameters, which is believed to be suited in most cases. For ARMA, the number of lag order p is set to 12 and the order of moving average q is set to 1. For SVR, the radial basis function (RBF) is used as the kernel function, and the penalty parameter is set to empirical value 1.0. For XGBoost, there are 100 estimators, each with depth up to 3, and the learning rate is 0.1.

For the neural-network-based models, we use different structures with similar complexity. A 1-layer LSTM network with 128 cells and an SAE with the same structure as the generator in PSN are used in the experiment. LSTM and SAE are trained on the data of each station for 20 epochs in a batch size of 128, while PSN is pretrained for 15 epochs and then fine-tuned for 5 epochs in a batch size of 128 as well. All neural networks are optimized by Adam method [43] with hyperparameters $\alpha = 0.0001, \beta_1 = 0, \beta_2 = 0.9$. The networks are implemented by using Tensorflow framework [44]. All experiments are performed on a workstation with an Intel Core i7-6700K CPU and two Nvidia GeForce GTX Titan X Graphics Cards.

D. Testing Results of One-to-One Prediction

Since short-term prediction is more challenging, we conduct experiments using the data in previous 1 hour to predict the traffic flow of next five minutes in this paper. For the PeMS dataset, that means setting the length of the historical time window T to be 12 and the time interval to 5 minutes.

We evaluate the prediction performance of the proposed PSN together with other models. The average prediction errors on 50 stations along with the standard deviations of errors are shown in Table I. The top-3 performances are shown in bold in this table.

It is clear that PSN methods work well in both usual and unusual cases under MAE and MAPE criteria. In usual cases, PSN works almost the same as LSTM and XGBoost. It achieves a test error rate (MAPE) of 11.96, which is 5.53% better and variant 54.04% less than SAE with the same structure.

Most importantly, PSN outperforms other advanced models in unusual cases. It outperforms SAE by 7.48% in MAE criteria and 1.91% in MAPE. Comparing with models that have similar performance, it is about 3.14% better than LSTM and 3.31% better than XGBoost under MAE criteria, and about 2.38% better than both competitors under MAPE criteria.

Among all the models, PSN is the only kind that performs well under both criteria. Most complex models are inferior to simpler models such as ARMA and SVR under MAE criteria while outperforming them under MAPE criteria. This indicates that compared with traditional methods, recent methods tend to be more error when the traffic flow is higher. The achievements of PSN imply its excellence for both peeks and idle hours predictions.

Our method brings practical benefit as well. As summarized in Table II, comparing different neural networks for their number of parameters and the total training time for 50 stations prediction, we find PSN can achieve close performance as LSTM while using only about one-fifth of the training time.

TABLE I

PERFORMANCE COMPARISON OF DIFFERENT MODELS FOR A 5-MINUTES ONE-TO-ONE PREDICTION

	MAE (Usual case)		MAPE (%, Usual Case)		MAE (Unusual case)		MAPE (%, Unusual Case)	
PSN	**23.34**	**±7.96**	**11.96**	**±2.90**	**68.99**	**±40.74**	**36.91**	**±16.36**
LSTM	**23**	**±7.87**	**11.7**	**±2.83**	71.23	±43.86	**37.8**	**±22.96**
XGBoost	**23.14**	**±7.93**	**11.97**	**±3.22**	71.35	±43.07	37.81	±27.34
SAE	23.42	±7.99	12.66	±6.31	74.57	±47.39	**37.63**	**±17.99**
SVR	23.51	±7.93	12.93	±5.78	**59.58**	**±35.72**	38.04	±41.56
ARMA	34.6	±9.68	24.56	±34.26	**48.05**	**±31.59**	39.67	±47.67

TABLE II

COMPLEXITY OF DIFFERENT NETWORKS

Networks	Parameters	Total Training Time (s)
SAE	67841	2185.2
LSTM	66689	20100.12
PSN (Pretraining Stage)	135682	3505.35
PSN (Fine-tuning Stage)	67841	550

V. CONCLUSION

In this paper, we propose Pattern-Sensitive Network to better capture the variation patterns of traffic flow and thus make a better prediction in extreme conditions. Testing results show that, by letting the neural network learn the joint distribution of future flow and historical records explicitly, we can handle the violent changes in traffic system more robustly without compromising its prediction accuracy. Our method also reduce the need for complex structures and thereby reducing the time cost as well.

The PSN can be further improved by using more complex network structures if the data is noisier. Moreover, our work indicates that the research of traffic prediction models can be extended by adopting the recent developments of GAN. We hope further applications, such as making a more precise prediction of various conditions, providing a better interpretation [45] for the prediction results, as well as combinations with software-defined transportation systems, data recovery, virtual-real interaction [46] can benefit from our study.

REFERENCES

[1] F.-Y. Wang, "Parallel control and management for intelligent transportation systems: Concepts, architectures, and applications," *IEEE Trans. Intell. Transp. Syst.*, vol. 11, no. 3, pp. 630–638, Sep. 2010.

[2] J. Zhang, F.-Y. Wang, K. Wang, W.-H. Lin, X. Xu, and C. Chen, "Data-driven intelligent transportation systems: A survey," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 4, pp. 1624–1639, Dec. 2011.

[3] E. I. Vlahogianni, M. G. Karlaftis, and J. C. Golias, "Short-term traffic forecasting: Where we are and where we're going," *Transp. Res. C, Emerg. Technol.*, vol. 43, pp. 3–19, Jun. 2014.

[4] Y. Lv, Y. Chen, X. Zhang, Y. Duan, and N. Li, "Social media based transportation research: The state of the work and the networking," *IEEE/CAA J. Autom. Sinica*, vol. 4, no. 1, pp. 19–26, Jan. 2017.

[5] H. Yang and S. Yagar, "Traffic assignment and signal control in saturated road networks," *Transp. Res. A, Policy Pract.*, vol. 29, no. 2, pp. 125–139, 1995.

[6] L. Li, Y. Lv, and F.-Y. Wang, "Traffic signal timing via deep reinforcement learning," *IEEE/CAA J. Autom. Sinica*, vol. 3, no. 3, pp. 247–254, Jul. 2016.

[7] F.-Y. Wang, N.-N. Zheng, D. Cao, C. M. Martinez, L. Li, and T. Liu, "Parallel driving in CPSS: A unified approach for transport automation and vehicle intelligence," *IEEE/CAA J. Autom. Sinica*, vol. 4, no. 4, pp. 577–587, 2017.

[8] G. E. P. Box, G. M. Jenkins, and J. F. MacGregor, "Some recent advances in forecasting and control," *J. Roy. Stat. Soc. Ser. C, Appl. Stat.*, vol. 23, no. 2, pp. 158–179, 1974.

[9] M. Levin and T. Yen-Der, "On forecasting freeway occupancies and volumes (abridgment)," *Transp. Res. Rec.*, no. 773, pp. 47–49, 1980.

[10] D. Basak, S. Pal, and D. C. Patranabis, "Support vector regression," *Neural Inf. Process. Lett. Rev.*, vol. 11, no. 10, pp. 203–224, 2007.

[11] M. Castro-Neto, Y.-S. Jeong, M.-K. Jeong, and L. D. Han, "Online-SVR for short-term traffic flow prediction under typical and atypical traffic conditions," *Expert Syst. Appl.*, vol. 36, no. 3, pp. 6164–6173, 2009.

[12] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep Learning*, vol. 1. Cambridge, MA, USA: MIT Press, 2016.

[13] W. Huang, G. Song, H. Hong, and K. Xie, "Deep architecture for traffic flow prediction: Deep belief networks with multitask learning," *IEEE Trans. Intell. Transp. Syst.*, vol. 15, no. 5, pp. 2191–2201, Oct. 2014.

[14] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang, "Traffic flow prediction with big data: A deep learning approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 2, pp. 865–873, Apr. 2015.

[15] X. Ma, Z. Tao, Y. Wang, H. Yu, and Y. Wang, "Long short-term memory neural network for traffic speed prediction using remote microwave sensor data," *Transp. Res. C, Emerg. Technol.*, vol. 54, pp. 187–197, May 2015.

[16] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, May 2015. [Online]. Available: http://www.nature.com/nature/journal/v521/n7553/abs/nature14539.html

[17] L. Li, X. Su, Y. Zhang, Y. Lin, and Z. Li, "Trend modeling for traffic time series analysis: An integrated study," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 6, pp. 3430–3439, Dec. 2015.

[18] S. Feng, R. Ke, X. Wang, Y. Zhang, and L. Li, "Traffic flow data compression considering burst components," *IET Intell. Transp. Syst.*, vol. 11, no. 9, pp. 572–580, Nov. 2017.

[19] R. Yu, Y. Li, C. Shahabi, U. Demiryurek, and Y. Liu, "Deep learning: A generic approach for extreme condition traffic forecasting," in *Proc. SIAM Int. Conf. Data Mining*, Jun. 2017, pp. 777–785.

[20] I. Goodfellow *et al.*, "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.

[21] K. Wang, C. Gou, Y. Duan, Y. Lin, X. Zheng, and F.-Y. Wang, "Generative adversarial networks: Introduction and outlook," *IEEE/CAA J. Autom. Sinica*, vol. 4, no. 4, pp. 588–598, Sep. 2017.

[22] C. Chen, Y. Wang, L. Li, J. Hu, and Z. Zhang, "The retrieval of intra-day trend and its influence on traffic prediction," *Transp. Res. C, Emerg. Technol.*, vol. 22, pp. 103–118, Jun. 2012.

[23] C. M. Bishop, *Pattern Recognition and Machine Learning* (Information Science and Statistics). New York, NY, USA: Springer, 2006.

[24] S. Kullback and R. A. Leibler, "On information and sufficiency," *Ann. Math. Statist.*, vol. 22, no. 1, pp. 79–86, 1951.

[25] *PeMS, California Performance Measurement System*. [Online]. Available: http://pems.dot.ca.gov/

[26] Y.-D. Chen, L. Li, Y. Zhang, J.-M. Hu, and X.-X. Jin, "Fluctuations in urban traffic networks," *Modern Phys. Lett. B*, vol. 22, no. 2, pp. 101–115, 2008.

[27] A.-L. Barabási, "The origin of bursts and heavy tails in human dynamics," *Nature*, vol. 435, pp. 207–211, May 2005. [Online]. Available: https://www.nature.com/articles/nature03459

[28] S. Zhang, "Using twitter to enhance traffic incident awareness," in *Proc. IEEE 18th Int. Conf. Intell. Transp. Syst.*, Sep. 2015, pp. 2941–2946.

[29] A. Koesdwiady, R. Soua, and F. Karray, "Improving traffic flow prediction with weather information in connected cars: A deep learning approach," *IEEE Trans. Veh. Technol.*, vol. 65, no. 12, pp. 9508–9517, Dec. 2016.

[30] Y. Jia, J. Wu, M. Ben-Akiva, R. Seshadri, and Y. Du, "Rainfall-integrated traffic speed prediction using deep learning method," *IET Intell. Transp. Syst.*, vol. 11, no. 9, pp. 531–536, Nov. 2017.

[31] Y. Li, R. Yu, C. Shahabi, and Y. Liu. (2017). "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting." [Online]. Available: https://arxiv.org/abs/1707.01926

[32] B. Yu, H. Yin, and Z. Zhu. (2017). "Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting." [Online]. Available: https://arxiv.org/abs/1709.04875

[33] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville. (2017). "Improved training of Wasserstein GANs." [Online]. Available: https://arxiv.org/abs/1704.00028
[34] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *Proc. ICML*, vol. 30, no. 1, Jun. 2013, p. 3.
[35] X. Dai, R. Fu, Y. Lin, L. Li, and F.-Y. Wang. (2017). "DeepTrend: A deep hierarchical neural network for traffic flow prediction." [Online]. Available: https://arxiv.org/abs/1707.03213
[36] S. Feng, X. Wang, H. Sun, Y. Zhang, and L. Li, "A better understanding of long-range temporal dependence of traffic flow time series," *Phys. A, Stat. Mech. Appl.*, vol. 492, pp. 639–650, Feb. 2018.
[37] G. E. P. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time Series Analysis: Forecasting and Control*. Hoboken, NJ, USA: Wiley, 2015.
[38] C.-H. Wu, J.-M. Ho, and D. T. Lee, "Travel-time prediction with support vector regression," *IEEE Trans. Intell. Transp. Syst.*, vol. 5, no. 4, pp. 276–281, Dec. 2004.
[39] T. Chen and C. Guestrin. (Mar. 2016). "XGBoost: A scalable tree boosting system." [Online]. Available: https://arxiv.org/abs/1603.02754
[40] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber, "LSTM: A search space odyssey," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 10, pp. 2222–2232, Oct. 2017.
[41] Y. Tian and L. Pan, "Predicting short-term traffic flow by long short-term memory recurrent neural network," in *Proc. IEEE Int. Conf. Smart City/SocialCom/SustainCom (SmartCity)*, Dec. 2015, pp. 153–158.
[42] R. Fu, Z. Zhang, and L. Li, "Using LSTM and GRU neural network methods for traffic flow prediction," in *Proc. 31st Youth Acad. Annu. Conf. Chin. Assoc. Automat. (YAC)*, Nov. 2016, pp. 324–328.
[43] D. P. Kingma and J. Ba. (Dec. 2014). "Adam: A method for stochastic optimization." [Online]. Available: https://arxiv.org/abs/1412.6980
[44] M. Abadi *et al.* (Mar. 2016). "TensorFlow: Large-scale machine learning on heterogeneous distributed systems." [Online]. Available: https://arxiv.org/abs/1603.04467
[45] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, "InfoGAN: Interpretable representation learning by information maximizing generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 2172–2180.
[46] Y. Lv *et al.*, "Generative adversarial networks for parallel transportation systems," *IEEE Intell. Transp. Syst. Mag.*, to be published.