Available online at www.sciencedirect.com

ScienceDirect

Transportation Research Procedia 00 (2018) 000–000

Transportation Research Procedia

www.elsevier.com/locate/procedia

World Conference on Transport Research - WCTR 2019 Mumbai 26-31 May 2019

Short-Term travel speed prediction for urban expressways using convolutional neural network and tensor decomposition

HAN, Tianyang[a]*,TANG, Keshuang[b] ,OGUCHI, Takashi[a]

[a]*Institution of Industry Science, University of Tokyo, Tokyo, 1538505, Japan*
[b]*College of Transportation Engineering, Tongji University Shanghai, 201804, China*

Abstract

Prediction is an important part of the traffic management system (TMS) which supports route planning, dynamic traffic control, and information provision. We developed a multi-dimensional learning machine for predicting the traffic speed. Proposed methodology considered both historical experience and near past observation of traffic data by combining a convolutional neural network (CNN) with tensor decomposition (TD) predictor. TD based method is treated as an effective traffic predictor considering temporal-spatial neighbourhood data. However, limited by learning mechanism, such a method cannot extract historical traffic pattern from large datasets. Our main contribution is converting the traffic prediction into a tensor imputation problem, which considers the historical pattern information from CNN by constructing input tensor. Besides, multiple low-rank choosing and weighted optimization are introduced to improve the accuracy of TD-based prediction. We validated our methodology using empirical detector data of urban expressway in Shanghai. Compared with single algorithms, our method has smaller absolute and relative error.

Keywords: Traffic Speed Prediction; Convolutional Neural Network; Tensor Decomposition; Fusion Algorithm; Expressway; Loop Detector;

1. Motivation

Traffic congestion is becoming severer with urban development. Traffic management system (TMS) can mitigate the congestion based on real-time prediction of traffic status. High resolution of spatial-temporal traffic data can support complex data-driven management scheme. While, large-scale traffic data also bring difficulty for traditional prediction algorithm. Nowadays, commercial traffic service providers (Microsoft Research, 2016) and researchers

* Corresponding author. Tel.: +81-3-5452-6419; fax: +81-3-5452-6420.
E-mail address: hanty@iis.u-tokyo.ac.jp

have developed many traffic prediction methods using machine learning. In these studies, deep learning has been widely applied due to its outstanding performance and capability of processing large-scale data.

However, state-of-art machine learning based methods still have insufficiencies in traffic prediction. Firstly, most algorithms are limited in time series analysis of data that cannot make full use of spatial-temporal information. Second, historical traffic patterns and near past data characteristics cannot be integrated in most cases. Besides, machine learning methods are treated as a black-box problem which interior procedures could not be observed or controlled generally.

To improve the insufficiencies above, this study develops a multi-dimensional traffic speed predictor considering both historical traffic pattern and near past data characteristics. The idea is to mimic human being that judging near future by both historical experience and recent observation. It is realized by combining CNN with Tucker Decomposition (TKD). Firstly, the traffic pattern is leaned by training CNN with historical datasets. Then, the pattern information from CNN could be combined with near past data as input of TKD. At last, a weighted gradient descent (WGD) is employed to derive optimized decomposition. The optimized approximation by TKD estimate the future accurately. The contributions of this study include

- The temporal spatial information can be considered by multi-dimension structure of the proposed method.
- CNN can learn the pattern information from historical data while TKD can integrate the pattern information and near past data.
- A weighted learning method is proposed to optimize decomposition. The prior knowledge is combined with machine learning.

This paper is organized as follows. Firstly, related works are reviewed in Chapter 2. Proposed methodology is mainly introduced in Chapter 3. With empirical data introduced in Chapter 4., the proposed method is validated, and the analysis results are presented as Chapter 5. Finally, in Chapter 6., we sum up our work and discuss the future work.

2. Literature review

In general, traffic congestion is identified based on the temporal changes in traffic volume or speed. Mathematical statistical methods and machine learnings are often used in congestion prediction.

Mathematical and statistical methods include historical average models, time series models, the Kalman filter (KF) and nonparametric models. Historical average algorithm and time series analysis are usually combined into auto regression integrated moving average (ARIMA), a traffic predictor most widely used in traffic this field. Otoshi et al. (2015) proposed a seasonal ARIMA algorithm applied in long-term traffic variation prediction while short term traffic variation was taken into the model to accommodate prediction uncertainty incurred by temporal traffic changes and data errors. Pavlyuk (2017) discussed the differences between multivariate auto regression moving average (VARMA) and ARIMA. VARMAs allow any dependencies between space points, while ARIMAs consider sections spatially independent. KF model has the robustness of long-term prediction (Kumar, 2017) but shows lower accuracy with the increase of traffic flow variation. Compared with the above algorithms, nonparametric models such as discrete probabilistic models (Liebig et al., 2016), K-nearest neighbor (KNN) (Chen et al., 2017) and Markov chains (Xia et al., 2016) are employed widely. These methods have better performance for traffic flow of different temporal granularities with low computation cost. But the data-driven mechanism leads accuracy dependent on data quality.

Generally, Machine learnings including any trainable algorithm learn information from error. Main state-of-art traffic prediction methods apply Neural Networks (NNs). Deep Learning (DL) methods including Stacked Auto-Encoder (SAE) (Lv et al., 2015; Duan et al., 2016), Recurrent NN (RNN) and Long Short-Term Memory NNs (LSTM) (Ma, X. et al 2015), Deep Belief Network (DBN) (Koesdwiady, et al., 2016; Soua et al., 2016) and CNN (Ma et al., 2017) are popular methods recently. Besides, TD based methods are also attractive since its simplicity in generalization of high-dimensional problem (Tan et al., 2016).

Almost all above mentioned researches treat pattern learning and prediction as a local problem with the fixed road section and short duration. And only time the series characteristic is considered. Tan et al. (2016) proved that the multi-dimensional prediction is more powerful than time series model by both literature review and empirical experiment. Although some mathematical statistical methods (Xia et al., 2016; Pavlyuk, 2017) have also conducted spatial-temporal structure discussing the correlation among road sections, the complex statistics need reliable data for calibration. Moreover, statistical measure can only show the phenomenon observed in data but not the nature of traffic flow. Polson et al. (2017) developed the two-dimensional (2D) structure that arranges traffic data as images using semi-affine transformation. This structure expanded DL to solve multi-dimensional problem. Recently, with the development of computer vision, multi-dimensional learning machines, originally used to process image and video, are introduced to traffic congestion research domains. In such researches, CNN and TD are highlighted because of its high-dimensional operators without any geometric transformation. Motivated by the simplicity in high-dimensional calculation, Ma et al (2017) proposed a 2D prediction method which arranges traffic speed data into temporal-spatial matrix and treat the matrix as an image.

Regarding TD based methods, more dimensions could be considered in pattern learning and prediction of traffic due to the low computation cost. Tan et al. (2016) proposed a short-term traffic prediction method based on dynamic high-order Principle Component Analysis (PCA). This method considered not only time and space, but also day and week variation of traffic flow. The four-dimensional model could learn the pattern that traffic demand changes periodically within one week and one month. Except for high-order PCA, TKD is also proved to have superior performance in learning traffic data pattern (Goulart et al., 2017; Chen et al., 2018). The TKD method is used in imputation of incomplete traffic data by decomposing the data tensor and learning the traffic variation feature of each dimension. Different from PCA, TKD employs high-odder singular value decomposition (SVD) method to decompose data tensor.

3. Methodology

For making equations and the pseudo codes clear, we follow the notation as follows. Scalar, vector, matrix, and higher order tensor could be denoted by character lower-case, bold lower-case, bold upper-case and bold upper-case calligraphy respectively, such as a, $\boldsymbol{a}$, $\boldsymbol{A}$, $\boldsymbol{\mathcal{A}}$. Besides, upper-case characters stand for range which a lower-case number belong to, as $i \in [1, I]$. Besides the scalar product, product and Kronecker product denoted by $\boldsymbol{\mathcal{A}}\boldsymbol{\mathcal{B}}, \boldsymbol{\mathcal{A}} \times \boldsymbol{\mathcal{B}}, \boldsymbol{\mathcal{A}} \otimes \boldsymbol{\mathcal{B}}$.

3.1. Overview

Proposed method including 3 main steps to combine CNN and TKD.

- Learn traffic pattern information from large historical data using CNN.
- Construct the input tensor with both pattern information and near past data.
- Estimate the target future by TKD approximation.

The overview of proposed methodology is as follows.

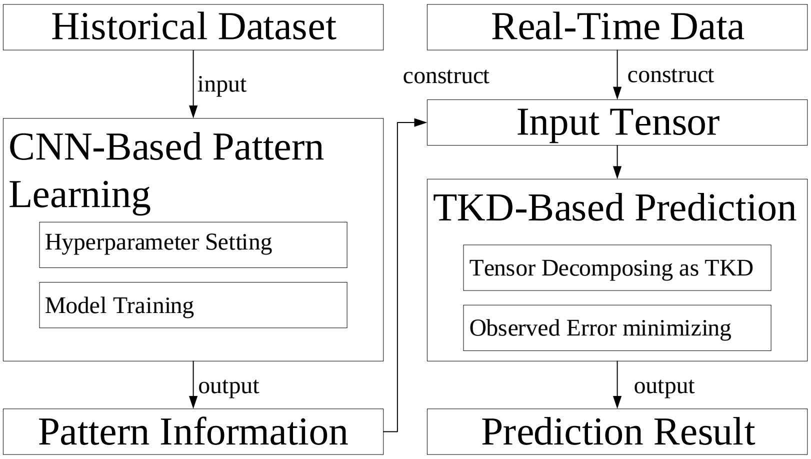


Figure 1The procedure framework of proposed methodology

CNN in step one is a pixel-to-pixel learning model. The sequent-day data is input as training label, so that the model parameter can describe day-to-day variation of traffic data as Zang et al. (2017). We select a two-dimensional structure that consider road section and time stamps of a day within one data sample. As for TKD in step three, we follow the three-dimensional structure as Chen et al. (2018). And it shown be optimal structure of tensor decomposition in later study (Chen et al., 2019).

Considering the difficulty to combine different dimension algorithms, we combine CNN and TKD exteriorly. The input tensor of TKD-based prediction consists of pattern information from CNN and real-time data. In detail, for every prediction task, the input tensor could be construct as Fig. 2.

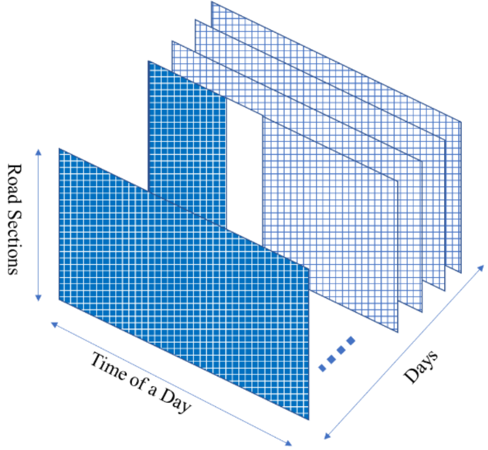


Figure 2 Input tensor for TKD composed by real-time data (blue grid) prediction data by CNN (white grid) and under prediction area (empty)

Note that CNN is a supervised learning method that extract day variation pattern in large historical data. However, after enough training, the model parameter of CNN becomes insensitive for slight difference data. While, TKD is unsupervised model that focus on the observed error of input data only. The single CNN or TKD model is similar to human beings follow only experience or observation. For better prediction performance, we try to combine experience and observation in this study.

3.2. Convolutional Neural Network

The CNN proposed by Lecun et. al. (1995) is unique which could accept multi-dimensional traffic data and learn the dimensional feature by locally connected neurons. Trainable convolution operators composed by these neurons' weight can form a convolution layer (C). Except for convolution layers, traditionally, pooling layers (S) and full-connected layers (FC) should be included in CNNs. To employ CNN in traffic data mining, we follow the idea from Ma et al. (2017) which learning traffic data as image.

In convolution layers, kennels process the temporal-spatial traffic data by trainable parameters. The convolutional kennels can reflect the traffic data characteristics after iterations. Using f as $[X_1, X_2]$ input and g as a $[M, N]$ filter the convolution $f * g$ could be denoted as follows.

$$f * g(x_1, x_2) = \sum_{m=1}^{M} \sum_{n=1}^{N} f(x_1 - m, x_2 - n) g(m, n); x, y \in R^{X_1 \times X_2} \tag{1}$$

Where x and y are the index of input data. Through the convolutional layer, the feature maps of input data could be obtained.

The pooling (subsampling) layer makes feature maps blur to save computation and increase robustness. This procedure makes CNN able to deal with large-scale and high-resolution traffic data. Take $[I, J]$ stochastic pooling as example, the stochastic point $(x - i, y - j)$ is chosen as a representation of the area $[x_1 - I: x_1, x_2 - J: x_2]$.

$$stochasticpooling(x_1, x_2) = f(x_1 - i, x_2 - j); x, y \in R^{X_1 \times X_2}; i \in [0, I-1]; j \in [0, J-1] \tag{2}$$

In full-connected layer, all the feature map produced by last layer are flattened as a feature vector. Set this vector as input of a NN (single or multi-layers) we can obtain the output of the NN as final output. The full-connected structure could be denoted as

$$\hat{y} = activefunc(\sum Wx + b) \tag{3}$$

Where $\hat{y}$ is one of the estimated outputs. x is the input that all output of last layer should be included in summation. W is the weight of connection and the b is the bias of the output $\hat{y}$. Active function of output could be various including sigmoid function, tanh function, rectifier function (known as ReLU), etc. For training the weight W, we follow the gradient descent method while momentum and regularization is added to increase the training performance. The error backward propagation could be denoted as follows.

$$W_{updated} = W + \gamma \Delta W + \alpha \frac{\partial(E(W,b) + \lambda \sum W^2)}{\partial W} \tag{4}$$

Where, γ is the momentum parameter, α is the learning rate and λ is the penalty factor of regularization. E is the error between outputs and train labels. The updating of bias also follows this method.

To the best of our knowledge. In our study, we conduct a comparison between 7-layer structure (C-S-C -S-C-S-FC) and 9-layer structure (C-C-S-C-C-S-C-S-FC) and record the top-5 highest accuracy hyperparameter option as Tab. 1. Here the normalized mean square error (MSE) (Eq.15) is used as accuracy criterion in evaluation.

Table 1 Top-5 hyperparameter option of CNN-7 and CNN-9 (*sigm=sigmoid function **relu = rectifier function)

ranking	α	activation function	γ	λ	Training MSE	Testing MSE
CNN-7						
1	0.01	sigm*	0.9	0	29.3987	33.1774
2	0.003	sigm+linear	0.9	0	31.1531	35.4218
3	0.01	sigm	0.9	0	33.2355	36.4882
4	0.01	sigm+relu**	0.9	0	33.4506	36.5449
5	0.01	sigm	0.9	0.00005	33.2748	36.9482
CNN-9						
1	0.003	sigm	0.9	0	33.6534	36.2400
2	0.01	sigm	None	0.00005	33.3132	36.4782
3	0.01	sigm+relu	None	0.00005	33.5012	36.7472
4	0.01	sigm	0.9	0	34.5914	36.7549
5	0.01	sigm+relu	0.9	0.0001	33.2710	36.8228

In the experiment, CNNs are trained for reconstructing data using one data sample as both input and label. We introduce different-sized convolution kernels to keep the same length of feature vectors. Different from square kernels in AlexNet (Krizhevsky et al., 2012), we introduce asymmetrical kennels as Zang et al. (2017) to solve difference between time and space dimension in traffic data.

From the result table, we can make a summary as. 1) Proposed CNN-7 has higher accuracy than CNN-9 in average that means a deeper structure likely has no improvement for accuracy. 2) sigmoid active function is proved that suit the nonlinearity of traffic data better compared with linear function, tanh and rectifier function. 3) regularization seems not necessary for learning traffic pattern from data.

According to the experiments, traffic pattern learning is shown some differences from image classification researches. According to the experiment, we determined the architecture and hyperparameter option as follows.

Table 2 architecture and hyperparameter option of proposed CNN

Leaning rate, momentum parameter and regularization penalty: $\alpha = 0.01; \gamma = 0.9; \lambda = 0;$				
type	operator size	stride	feature maps	active function
C	[5,13]	1	8	sigmoid
S (average)	[2,4]	[1,4]	8	
C	[5,15]	1	16	sigmoid
S (average)	[2,5]	[2,5]	16	
C	[2,4]	1	32	sigmoid
S (average)	[2,2]	[2,2]	32	
FC				sigmoid

The architecture of proposed CNN is shown in Fig. 3

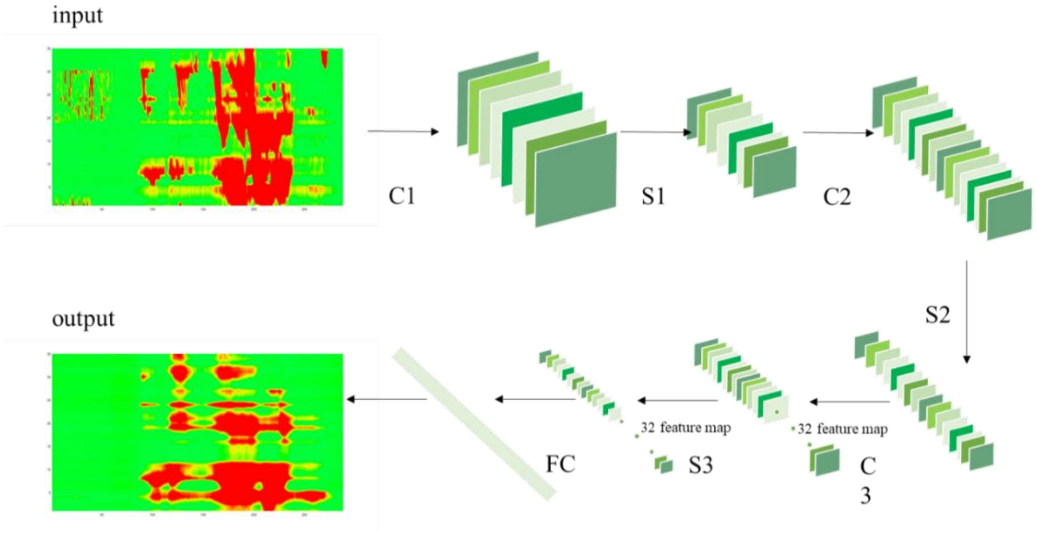


Figure 3 proposed CNN architecture

3.3. Tucker decomposition

The concept of TKD (Tucker, L. 1966) is to decompose a tensor as one core multiplied by factor matrices of every dimension that could be denoted as

$$\boldsymbol{\mathcal{X}} \approx \boldsymbol{\mathcal{G}} \times_1 \boldsymbol{F}_1 \times_2 \boldsymbol{F}_2 \times_3 \ldots \times_P \boldsymbol{F}_P \tag{5}$$

Where $\boldsymbol{\mathcal{X}}$ is the Pth order tensor and $\boldsymbol{\mathcal{X}} \in \mathrm{R}^{\mathrm{N}_1 \times \mathrm{N}_2 \times \ldots \times \mathrm{N}_\mathrm{P}}$. N_P size of the tensor in mode p. $\boldsymbol{\mathcal{G}}$ is the tensor core and $\boldsymbol{\mathcal{G}} \in \mathrm{R}^{\mathrm{R}_1 \times \mathrm{R}_2 \times \ldots \times \mathrm{R}_\mathrm{P}}$. R_p is the low rank of mode p. $\boldsymbol{F_p}$ is the factor matrix of mode p. $\times_\mathrm{p}$ is the mode_p production of tensor which means the product of tensor's mode_p unfolding and corresponding matrix, as.

$$\boldsymbol{\mathcal{X}} \times_p \mathbf{A} = \mathit{folding}(\mathbf{A}' \times \mathbf{X}_{(p)}); \mathbf{A} \in R^{N_p \times I_p} \tag{6}$$

Where, $\boldsymbol{\mathcal{X}}_{(\mathrm{p})} \in \mathrm{R}^{\mathrm{N_p} \times N_1 \dots N_{p-1} N_{p+1} \dots N_P}$ is the mode_p unfolding of tensor $\mathcal{X}$.

In this study, a 2-step TKD based prediction is introduced to approximate the near future traffic speed. The first step is to initialize the core and factor matrices with truncated SVD method (Chen et al., 2018). The second step is to optimize the core and factor matrices using WGD. We named these two steps by decomposition and optimization.

The step decomposition is to find low rank representation of input tensor. This method assumes several principle cardinalities can summarize all data. Then any entries could be estimated by existed cardinalities. In traffic prediction task the number of cardinalities $\mathrm{R_p}$ (low rank of p-dimension) should be less than the tensor size $\mathrm{N_p}$ to promise enough observation to judge the future.

In Eq. 5 the cardinalities of each dimension compose the factor matrix $\mathbf{F_p} \in R^{\mathrm{N_p} \times \mathrm{R_p}}$**.** Correspondingly, the core $\boldsymbol{\mathcal{G}} \in R^{\mathrm{R_1} \times \dots \times \mathrm{R_P}}$ represent the co-effective of all dimension’s features. we can also denote each entry in tensor by Tucker decomposition as Eq.7

$$x_{n_1,\dots,n_P} \approx \sum_{r_1=1}^{R_1} \cdots \sum_{r_P=1}^{R_P} g_{r_1 \dots r_P} \prod_{p=1}^{P} f_p(n_p, r_p) \tag{7}$$

Where, x, g, f_p are entries in input tensor, core and factor matrices, and n_p, r_p are the index. With Tucker decomposition, we can represent the prediction target by pattern extract from input data. However, we should find appropriate low ranks of each dimension in advanced. Chen, B. et al. (2015) proposed an optimal low rank TKD approximation that optimize core size in advance. In transport research area, Goulart, J. D. M. et al. (2017) applied the variable low rank called soft core determined iteration by iteration. However, in case of real-time prediction, the optimize may cause serious delay.

Therefore, proposed method follow the truncated SVD method with three fixed thresholds as Algorithm 1.

Algorithm 1. Truncated SVD with fixed threshold for mode-p.

```
input: mode-p unfolding of tensor $\mathbf{X}_{(\mathrm{p})}$ multiple thresholds thres
do SVD as $\mathbf{X}_{(\mathrm{p})} = \boldsymbol{U\Sigma M^T}$
if $\sum_1^2 \sigma \,/\, \sum_1^{N_p} \sigma < \mathrm{thres_i}$
        for $r$ =2:$\mathrm{N_p}$
                if $\sum_1^r \sigma \,/\, \sum_1^{N_p} \sigma > \mathrm{thres_i}$
                        $\mathrm{R_{p,i}} = r$; break;
else
        $\mathrm{R_{p,i}} = 2$;
$\mathrm{F_{p,i}} = \mathrm{R_{p,i}}$ left colums U (Truncated U)
return: $\mathrm{R_{p,1}}, \mathbf{F_{p,1}}, \mathrm{R_{p,2}}, \mathbf{F_{p,2}}, \mathrm{R_{p,3}}, \mathbf{F_{p,3}}$
```

σ is the singular value of $\mathbf{X}_{(\mathrm{p})}$, the summation including left/ largest 2, r and $\mathrm{N_p}$ σ. In our methodology, the 3D data tensor makes the p could be 1, 2 or 3 in Algorithm 1. The threshold represents the proportion of principle component in data. With truncated SVD we can initialize the factor matrices for all dimension. Then, the initialized core could be calculated using Eq. 8.

$$\boldsymbol{\mathcal{G}} = \boldsymbol{\mathcal{X}} \times_1 \boldsymbol{F}_1^T \times_2 \dots \times_P \boldsymbol{F}_P^T \tag{8}$$

Where, P=3 for 3D data tensor. The factor matrices with different low rank threshold could be chosen in one calculation. For convenience, we omit the threshold index in formula during paper remained. After calculating the estimated tensor $\hat{\mathcal{X}}$ with Eq. 5, the square error E is the ℓ_2 norm of difference between $\hat{\boldsymbol{\mathcal{X}}}$ and $\boldsymbol{\mathcal{X}}$ denoted by Eq. 10.

$$\mathrm{E} = \|\boldsymbol{\mathcal{X}} - \hat{\boldsymbol{\mathcal{X}}}\|_{\mathrm{F}} = \sqrt{\sum_{n_1,\dots,n_P}^{N_1,\dots,N_P} (x_{n_1,\dots,n_P} - \sum_{r_1=1}^{R_1}\cdots\sum_{r_P=1}^{R_P} g_{r_1\dots r_P} \prod_{p=1}^{P} f_p(n_p, r_p))^2} \tag{9}$$

Here the P = 3 in proposed methodology. For speed up the optimization, the regularization term L as in Eq. 4 is introduced into optimization objective. In tensor level, the ℓ_2 norm of the is applied. L is square summation of each entries in core and factor matrices as Eq. 11.

$$\mathrm{L} = \sum g^2 + \sum f_1{}^2 + \dots + \sum f_P{}^2 \tag{10}$$

Then the cost function could be denoted as Eq. 12

$$\min J = \mathrm{E} + \lambda \mathrm{L} \tag{11}$$

With objective J, the gradient could be calculated as follows.

$$\begin{aligned} \frac{\partial J}{\partial g} &= -\sum e \prod f_p + \lambda g \\ \frac{\partial J}{\partial f_p} &= -\sum [e(\sum \dots \sum f_{p'})] + \lambda f_p ;\ p' \neq p \end{aligned} \tag{12}$$

Where the e is the entries in residential tensor $\boldsymbol{\mathcal{E}} \in \mathrm{R}^{N_1 \times \dots \times N_P}$The core and factor matrices are optimized along with the gradient. The optimization is shown as Algorithm 2.

Algorithm 2. Optimization with weighted GD

```
input: dynamic tensor $\boldsymbol{\mathcal{X}}$, binary tensor $\boldsymbol{\mathcal{B}}$ to define the prediction area with zero and other area with one, weighted tensor $\boldsymbol{\mathcal{W}}$, initialization $\boldsymbol{\mathcal{G}}, \mathbf{F_1}, \dots, \mathbf{F_p}$ (selected from different low rank option), learning rate α, penalty factor γ, converge condition ϵ, maximum iteration number M.
$\boldsymbol{\mathcal{G}}^{(1)} = \boldsymbol{\mathcal{G}}, \mathbf{F_p^{(1)}} = \mathbf{F_p}$
for m=1:M
    $\widehat{\boldsymbol{\mathcal{X}}}^{(m)} = \boldsymbol{\mathcal{G}}^{(\boldsymbol{m})} \times_1 \boldsymbol{F}_1^{(\boldsymbol{m})} \dots \times_{\boldsymbol{P}} \boldsymbol{F}_{\boldsymbol{P}}^{(m)}$
    $\boldsymbol{\mathcal{E}}^{(\mathrm{m})} = \boldsymbol{\mathcal{X}} - \widehat{\boldsymbol{\mathcal{X}}}^{(m)}$
    $\boldsymbol{F}_{\boldsymbol{p}}^{(m+1)} = (1 - \alpha\lambda)\boldsymbol{F}_{\boldsymbol{p}}^{(m)} + \alpha(\boldsymbol{\mathcal{B}}\boldsymbol{\mathcal{W}}\boldsymbol{\mathcal{E}}^{(\mathrm{m})})_{(\mathrm{p})}(\boldsymbol{F}_1^{(\boldsymbol{m})} \otimes \dots \boldsymbol{F}_{\boldsymbol{p}-1}^{(m)} \otimes \boldsymbol{F}_{\boldsymbol{p}+1}^{(m)} \otimes \dots \boldsymbol{F}_{\boldsymbol{P}}^{(m)})\mathbf{G}_{(p)}^{(\boldsymbol{m})}$
    $\boldsymbol{\mathcal{G}}^{(\boldsymbol{m}+1)} = (1 - \alpha\lambda)\boldsymbol{\mathcal{G}}^{(\boldsymbol{m})} + \alpha\boldsymbol{\mathcal{B}}\boldsymbol{\mathcal{W}}\boldsymbol{\mathcal{E}}^{(\mathrm{m})} \times_1 \boldsymbol{F}_1^{\boldsymbol{T}} \dots \times_{\boldsymbol{P}} \boldsymbol{F}_{\boldsymbol{P}}^{\boldsymbol{T}}$
    $\widehat{\boldsymbol{\mathcal{X}}}^{(m+1)} = \boldsymbol{\mathcal{G}}^{(\boldsymbol{m}+1)} \times_1 \boldsymbol{F}_1^{(\boldsymbol{m}+1)} \dots \times_{\boldsymbol{P}} \boldsymbol{F}_{\boldsymbol{P}}^{(m+1)}$
    if E < ϵ
        break;
return $\widehat{\boldsymbol{\mathcal{X}}}^{(m+1)}, \boldsymbol{\mathcal{G}}^{(\boldsymbol{m}+1)}, \boldsymbol{F}_{\boldsymbol{p}}^{(m+1)}$
```

In Algorithm 2. we introduced the weighted tensor to improve the optimization. The traffic speed has a continuity in time and space, that means the current traffic status is influenced by short-term preceding status and neighboring road sections dramatically. While the data far away from prediction duration has less relation with our optimization objective. Therefore, we determine each entry in weight tensor by inverse value of Euclidean distance to center of prediction area.

Besides, the maximum iteration is set to control the computing time. Since proposed procedure estimated the short-term traffic speed parallelly with several group, the maximum iteration can also help to find fast convergence option. After all groups finishing optimization, we choose the optimal estimation with lowest square residual.

4. Empirical data

To demonstrate the proposed methodology, an empirical study is conduced. In this chapter, we give an introduction about empirical data in our study.

4.1. Study site

North side of YAN AN Expressway in Shanghai is selected as the empirical study site in this study. The data source is the in-ground loop detectors. 35 detected cross-sections distributed evenly within 14.2 Km expressway. These sections cover the lane number change from 2 to 5 because of the influence of 14 ramps and 2 interchanges. Fig. 4 shows the structure of road section. The speed data is collected each 5 min in full year 2011 while from 20th to 24th March data is missing.

We arrange the traffic speed into spatial-temporal matrix as Eq. 14. Each entry in matrix is weighted mean speed of cross-section. The vertical axis is the spatial dimension and the traffic flow direction is from top to bottom. The horizontal axis is the temporal dimension which cover 288 intervals in one day.

$$\text{matrix}_{\text{speed}} = \begin{bmatrix} v_{1,1} & \cdots & v_{288,1} \\ . & . & . \\ v_{35,1} & \cdots & v_{35,288} \end{bmatrix} \tag{13}$$

4.2. Data recovery and normalization

In-ground detector and GPS probe (e.g. floating car) as the main resource of traffic data has various quality problem. Detectors will be influenced by maintenance, weather and aging problem, while GPS probe need a necessary sampling rate to promise the reliability of data. that means missing and error in traffic data set is inevitable. Moreover, the real-time prediction cannot support large computing for precise imputation.

For solving imputation problem above, we introduce a rank-1 tensor entries approximation. The concept is to use neighbour data recover the missing after arranging a set of data into tensor. In our case, we approximated missing entries located in $(\mathrm{n1}, \mathrm{n2}, \mathrm{n3})$ as Eq. 15.

$$x_{n1,n2,n3} = \mu \omega_{n1} \omega_{n2} \omega_{n2} \tag{14}$$

Where, μ is the mean value of all known entries and $\omega_{\mathrm{n1}}, \omega_{n2}, \omega_{n3}$ are coefficient which belong to coefficient vector $\boldsymbol{\omega_1}, \boldsymbol{\omega_2}, \boldsymbol{\omega_3}$ of 3 dimensions of data tensor. Eq.15 could be treat as the special case of Eq. 8 where the low rank in each dimension is 1 and the core has only one fixed entry μ. After initializing coefficient vectors with normalized random number, the gradient descent is applied for optimize the approximation.

The procedure of imputation follows Algorithm 2., while 3 differences should be emphasized as follows,

- Set the binary tensor $\boldsymbol{\mathcal{B}}$ to mark the missing entries but not the prediction duration.
- Core tensor $\boldsymbol{\mathcal{G}}$ equal to the mean value μ which should not be updated in GD. And the factor matrices are coefficient vector which initialized by Gaussian random.
- weighted tensor $\boldsymbol{\mathcal{W}}$ should be set invalid because the missing entries distributed in data tensor are not intensive as the problem of prediction.

In our case, because of malfunction and maintenance of detectors, the missing data (3.64%,) and detection error (2.86%) do exist. The main missing problem example could be shown in Fig.4 including detector malfunction, maintenance and abnormal missing data. the missing entries is recovered by method above and the result is shown in example in Fig. 5.

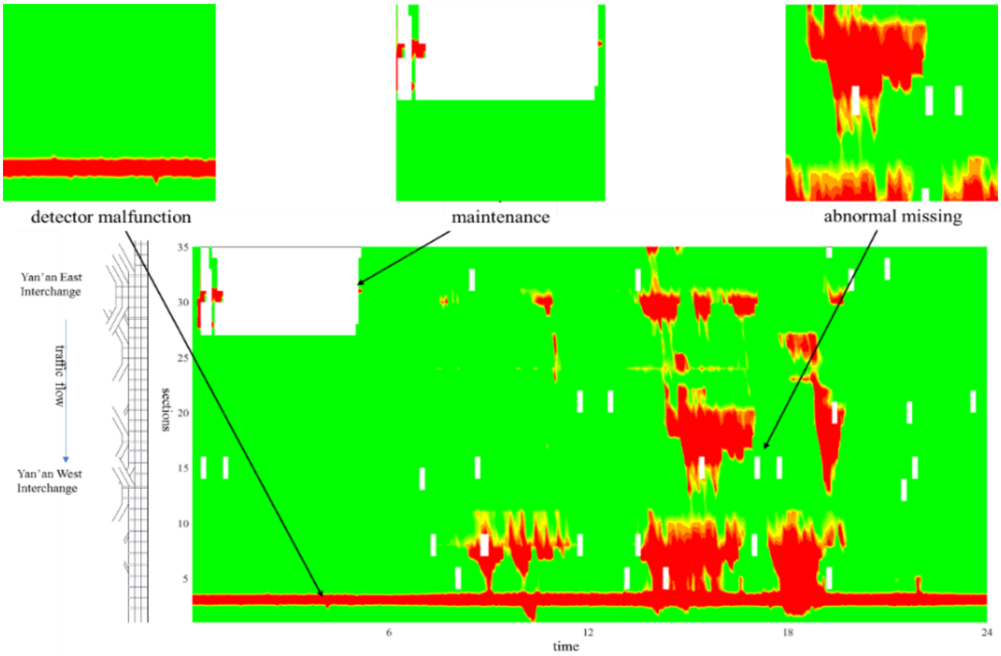


Figure 4 Data missing problem example: smooth (over 40 km/h, green), congestion (25-40 km/h, yellow), jam (below 25 km/h)

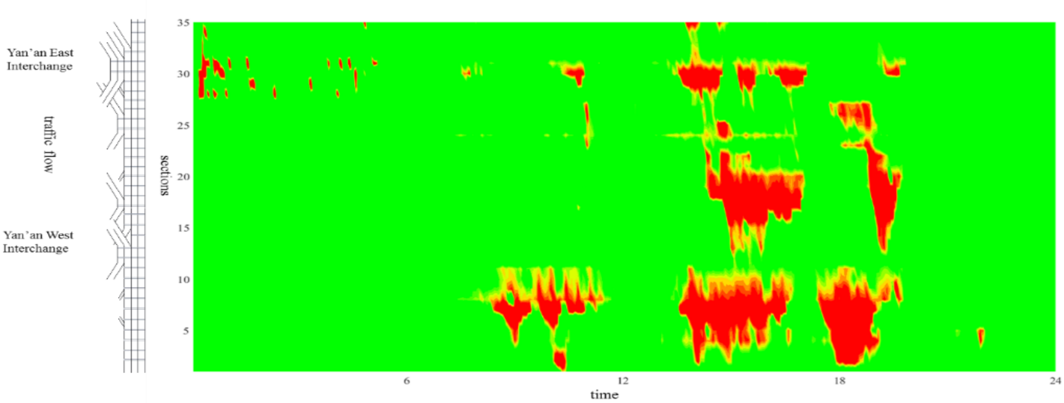


Figure 5 Data imputation result

The maximum normalization is introduced to processing raw data which denoted by kilometer per hour (km/h) speed.

5. Result

In this part, we evaluate the model by 2 accuracy criterion including root mean square error (RMSE) and mean absolutely percentage mean error (MAPE) in Eq.16. Different from the MSE in 3.2, we use the unit km/h for easier understanding here.

$$MSE = \frac{1}{n}\sum (y-\hat{y})^2$$
$$RMSE = \frac{1}{n}\sum \sqrt{(y-\hat{y})^2} \qquad (15)$$
$$MAPE = \frac{1}{n}\sum \frac{|y-\hat{y}|}{y} \times 100\%$$

5.1. CNN Setting

For employing CNN in pattern extraction, we set first 300-days data as training data and the remainder as testing data. Same as the idea of day-to-day prediction (Zang et al., 2017), we extract the day variation as pattern information in travel speed data. To do so, we have trained 4 types of day-to-day learning machine. The all input data input data is the whole day data before prediction target and the training label includes the whole day data of prediction target (d-d) and 3 sequence days (d-1d-d, d-2d-d, d-3d-d). The white grid area in figure 2 is filled by these CNN outputs. The accuracy of CNNs are shown in Tab. 3.

Table 3 Best result of 4 types day-to-day pattern extraction

Prediction type	Train RMSE	Train MAPE	Test RMSE	Test MAPE
d-d	7.45086	19.47170%	7.85923	21.63924%
d-1d-d	7.42523	19.56423%	7.90052	21.77751%
d-2d-d	7.59747	19.39162%	8.01623	21.20925%
d-3d-d	7.29254	19.60855%	7.70009	21.66283%

The result above shows the accuracy is stable with day variation interval. Not like time series methods, CNN are more robust since they consider historical experience only. The difference among input data is tend to be ignored. To demonstrate the advantage of combining experience and observation, we compare the CNN day-to-day prediction with our model in Fig. 6.

5.2. Combined Model Setting

Following the methodology in last chapter, we design the detail procedure as follows.

- Choosing objective day and predict area (20 min, all cross-section), construct the 7-day dynamic tensor with size 7×35×288 including past 3 days, current days and future 3 days prediction.
- Fill the history data in past area (blue grid area in Fig.2), fill the CNN prediction data in area remained.
- Create the binary tensor with size 7×35×288 fill zero in prediction duration. Fill one in area remained.
- Choose 0.5, 0.6, 0.7 as low rank thresholds to initialize TKD with Algorithm 1.
- Optimize the estimation with Algorithm 2.

5.3. Result Analysis

We demonstrate the significance of our method by comparing with 1) normal gradient descent version of the proposed method; 2) tensor decomposition method in Chen et al. (2018); 3) Day-to day CNN prediction in Zang et al. (2017).

In experiments, 5 days (11th -15th Nov.) in the test dataset of CNN have been taken into predicted experiments. For every 20 min, we predict the short-term travel speed, so we conduct 360 experiments in total. The result is shown in Tab. 4

Table 4 result comparison of combined model

Model	RMSE	MAPE
Our method	7.30321	14.71126%
Our method with original GD	7.34829	15.45562%
Tensor decomposition	7.70056	15.20617%
Day-to-day CNN	7.38486	20.16508%

We can summarize the comparison as follows.

- The weighted GD as one of our innovation can improve the MAPE of prediction while the RMSE is almost the same. It indicates that the low speed area become more precise when applying weighted GD.
- Tensor decomposition without pattern information achieved lower accuracy. A probable reason could be that real data including more noise than pattern information.
- Proposed method reduces MAPE significantly from CNN day-to-day prediction. it indicates the prediction performance of proposed method in low speed area has much higher accuracy. Different form CNN or any filter-based predictor, TD-based methods estimate unknown data by approximating the relationship among entries but not finding history average in some sense. The result validated our idea to make combination.

For Further concluding our significance, we compared the proposed method with day-to-day CNN with data sample on Nov. 12th. The result is shown in Fig. 6. In general, the proposed method can achieve higher accuracy than CNN, especially for congestion (low-speed) case. The failure of CNN may be caused by the variation of rush hour and bottleneck. Firstly, the speed difference between smooth status and congestion status in expressway is higher than low-level urban road. CNN, as learning algorithm, learned average from large history data. This procedure may smoothen the drastic change in traffic status transient. Secondly, we didn’t divide data by public holiday, special event, maintenance and traffic control since lack of information. Such special case may cause non-recurrent congestion that different from recurrent congestion with strong periodicity. Finally, the a 2D structure can only learn spatial-temporal pattern. Representing day variation by adjusting input and label leads to a loss of information between 2 days.

6. Conclusions

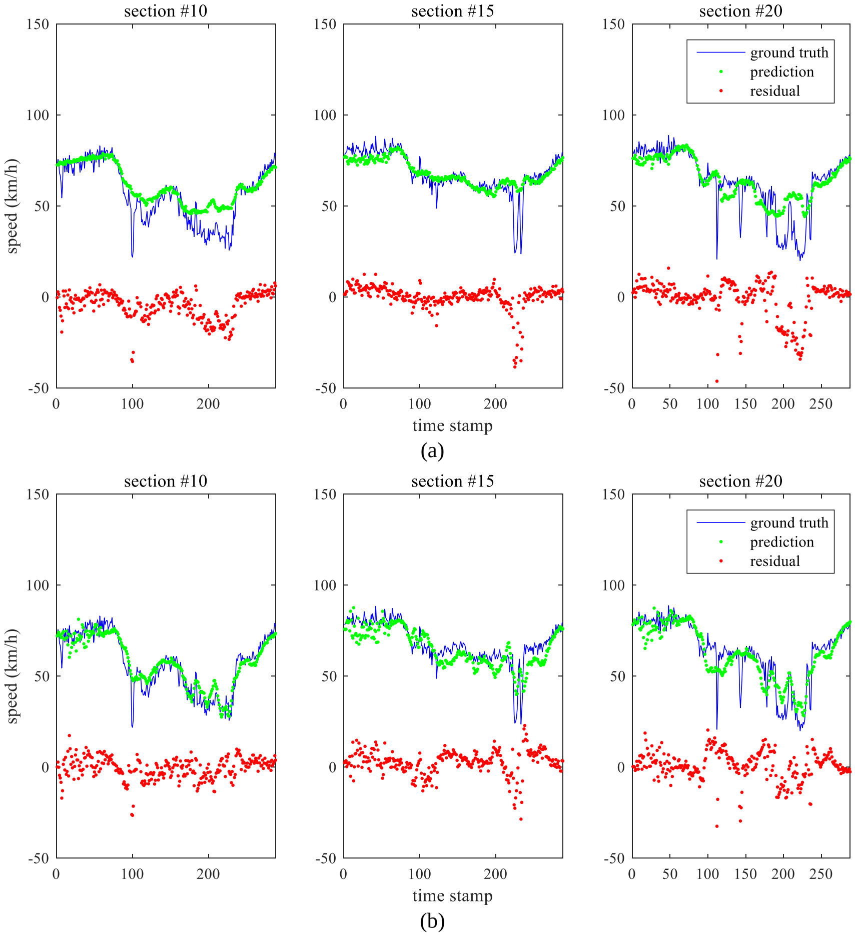
Traffic prediction is a crucial task of TMS, especially for large-scale road network real-time speed prediction. In this paper, a CNN-TKD short-term traffic speed prediction algorithm is proposed. It is applicable for real-time prediction with high accuracy and low computation cost. We utilized the complementarity of 2 sub-algorithms to consider both experience and observation in prediction. And, our concept was validated by the empirical data experiment.

The innovations of this research could be summarized as follows:

- Proposed methodology combined 2 multi-dimensional learning machines. The spatial-temporal features are extracted from data without statistical analysis and geometry transformation.
- Complementarity of CNN and TKD is utilized in our research. CNN learns traffic patterns through large historical data training. TKD could consider pattern information and near past reality.
- We discuss the improvement and control of machine learning by our existent knowledge. WGD is applied to control the optimization.

Although such combination and improvement provide innovated idea for traffic data mining and prediction problem, some difficulties still exist. In term of traffic prediction, Polson et al. (2017) figured out that general predictor has the difficulty for predicting the transient part of traffic flow where the speed discontinuously drops. The solution in their work is to utilize the ℓ_1 filter, while the transient part still has larger residual in their validation. Another issue is the missing and low quality of data. Different from data imputation problem, real-time traffic prediction has no sufficient data and computation condition dealing with data error. In term of algorithms, the challenge is the selection of parameters and hyperparameters. CNN has a large set of hyperparameters including architecture and optimizer.

Meanwhile the determination of low rank in tensor decomposition is also a well-defined NP-hard problem (Chen, X. et al., 2018).



(a)

(b)

Figure 6 The prediction result: (a) CNN, (b) our method of 12th Nov. 2011 data

Our study considered artery network at this stage. And limited by modeling technique, experiment condition and data sources, only a few improvement measures are implemented. We plan to extend our model to complex road network and attempt more measures to improve the machine learning based prediction. Moreover, the fusion data case should be taken into consideration. In the future, speed, volume and other information from detectors, floating car, video and even crowd source are expected to be included. We are also focusing on the combination of machine learning and existent knowledge. Since tensor decomposition method has simplicity in mechanism, we are aiming at controls of machine learning based prediction including boundary conditions, new parameters for error propagation and decision fusion with theoretical congestion evolution model.

Acknowledgement

I would like to express my gratitude to all those who helped me during the writing of this paper. I gratefully acknowledge the help of Chen, Xinyu from Guangdong Provincial Key Laboratory of Intelligent Transportation Systems, Research Center of Intelligent Transportation System, Sun Yat-Sen University.

Reference

Chen, B., Li, Z., & Zhang, S. (2015). On optimal low rank Tucker approximation for tensors: the case for an adjustable core size. *Journal of Global Optimization*, *62*(4), 811-832.

Chen, C., Liu, X., Qiu, T., & Sangaiah, A. K. (2017). A short-term traffic prediction model in the vehicular cyber–physical systems. *Future Generation Computer Systems*.

Chen, X., He, Z., & Wang, J. (2018). Spatial-temporal traffic speed patterns discovery and incomplete data recovery via SVD-combined tensor decomposition. *Transportation Research Part C: Emerging Technologies*, *86*, 59-77.

Duan, Y., Lv, Y., & Wang, F. Y. (2016, July). Performance evaluation of the deep learning approach for traffic flow prediction at different times. In *Service Operations and Logistics, and Informatics (SOLI), 2016 IEEE International Conference on* (pp. 223-227). IEEE.

Goulart, J. D. M., Kibangou, A. Y., & Favier, G. (2017). Traffic data imputation via tensor completion based on soft thresholding of Tucker core. Transportation Research Part C: Emerging Technologies, 85, 348-362.

Ji S, Xu W, Yang M, et al. 3D convolutional neural networks for human action recognition[J]. IEEE transactions on pattern analysis and machine intelligence, 2013, 35(1): 221-231.

Koesdwiady, A., Soua, R., & Karray, F. (2016). Improving traffic flow prediction with weather information in connected cars: a deep learning approach. *IEEE Transactions on Vehicular Technology*, *65*(12), 9508-9517.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems* (pp. 1097-1105).

Kumar, S. V. (2017). Traffic flow prediction using kalman filtering technique. *Procedia Engineering*, *187*, 582-587.

LeCun, Y., & Bengio, Y. (1995). Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, *3361*(10), 1995.

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

Liebig, T., Piatkowski, N., Bockermann, C., & Morik, K. (2017). Dynamic route planning with real-time traffic predictions. *Information Systems*, *64*, 258-265.

Lv, Y., Duan, Y., Kang, W., Li, Z., & Wang, F. Y. (2015). Traffic flow prediction with big data: A deep learning approach. *IEEE Trans. Intelligent Transportation Systems*, *16*(2), 865-873.

Ma, X., Dai, Z., He, Z., Ma, J., Wang, Y., & Wang, Y. (2017). Learning traffic as images: a deep convolutional neural network for large-scale transportation network speed prediction. *Sensors*, *17*(4), 818.

Ma, X., Tao, Z., Wang, Y., Yu, H., & Wang, Y. (2015). Long short-term memory neural network for traffic speed prediction using remote microwave sensor data. Transportation Research Part C: Emerging Technologies, 54, 187-197.

Ma, X., Yu, H., Wang, Y., & Wang, Y. (2015). Large-scale transportation network congestion evolution prediction using deep learning theory. *PloS one*, *10*(3), e0119044.

Microsoft Research, 2016. Predictive Analytics for Traffic <http://research.microsoft.com/en-us/projects/clearflow/> (accessed 14-February-2016).

Otoshi, T., Ohsita, Y., Murata, M., Takahashi, Y., Ishibashi, K., & Shiomoto, K. (2015). Traffic prediction for dynamic traffic engineering. *Computer Networks*, *85*, 36-50.

Pavlyuk, D. (2017). Short-term traffic forecasting using multivariate autoregressive models. *Procedia Engineering*, *178*, 57-66.

Polson, N. G., & Sokolov, V. O. (2017). Deep learning for short-term traffic flow prediction. *Transportation Research Part C: Emerging Technologies*, *79*, 1-17.

Soua, R., Koesdwiady, A., & Karray, F. (2016, July). Big-data-generated traffic flow prediction using deep learning and dempster-shafer theory. In *Neural Networks (IJCNN), 2016 International Joint Conference on* (pp. 3195-3202). IEEE.

Tan, H., Wu, Y., Shen, B., Jin, P. J., & Ran, B. (2016). Short-term traffic prediction based on dynamic tensor completion. *IEEE Transactions on Intelligent Transportation Systems*, *17*(8), 2123-2133.

Tucker, L. R. (1966). Some mathematical notes on three-mode factor analysis. *Psychometrika*, *31*(3), 279-311.

Xia, D., Wang, B., Li, H., Li, Y., & Zhang, Z. (2016). A distributed spatial–temporal weighted model on MapReduce for short-term traffic flow forecasting. *Neurocomputing*, *179*, 246-263.

Zang, D., Ling, J., Cheng, J., Tang, K., & Li, X. (2017, October). Using Convolutional Neural Network with Asymmetrical Kernels to Predict Speed of Elevated Highway. In *International Conference on Intelligence Science* (pp. 212-221). Springer, Cham.