Understanding Network Traffic States using Transfer Learning

Krishnakumari, Panchamy; Perotti, Alan; Pinto, Viviana; Cats, Oded; van Lint, Hans

DOI
10.1109/ITSC.2018.8569450

Publication date
2018

Document Version
Final published version

Published in
Proceedings of the 21st International Conference on Intelligent Transportation Systems (ITSC)

2018 21st International Conference on Intelligent Transportation Systems (ITSC)
Maui, Hawaii, USA, November 4-7, 2018

Understanding Network Traffic States using Transfer Learning

Panchamy Krishnakumari[1], Alan Perotti[2], Viviana Pinto[3], Oded Cats[1] and Hans van Lint[1]

***Abstract*— Large-scale network traffic analysis is crucial for many transport applications, ranging from estimation and prediction to control and planning. One of the key issues is how to integrate spatial and temporal analyses efficiently. Deep Learning is gaining momentum as a go-to approach for artificial vision, and transfer learning approaches allow to exploit pretrained models and apply them to new domains. In this paper, we encode traffic states as images and use a pretrained deep convolutional neural network as a feature extractor. Experimental results show how the extracted feature vectors cluster naturally into meaningful network traffic states and illustrate how these network states can be used for traffic state prediction.**

***Index Terms*— Network traffic, convolutional neural networks, deep learning, transfer learning**

I. Introduction

For many applications within transportation, such as estimation and prediction of traffic conditions, and network-wide traffic control and management, methods to efficiently analyze large datasets associated with large-scale traffic networks are crucial. For prediction purposes, for example, by and large two categories of approaches can be identified - data-driven and model-based. Data-driven approaches use general purpose parameterized mathematical models to capture (learn) from data the correlations between traffic variables (speed, travel time, flow) over space and time. Examples include simple generalized linear regression [1], [2], support vector regression approaches [3], [4], and a wide range of different ANN models [5]–[7] to name a few (there are many overviews, e.g. [8]–[11]). In the last few years data-driven estimation and prediction approaches have been developed that operate on entire networks, particularly using deep learning techniques [12]–[15]. Data-driven approaches require few prior assumptions; are typically robust to data failure; and can operate largely autonomously at good performance. The downside is that the past is not always a good predictor for the future, e.g. in case of incidents and accidents. Moreover, the lack of explanatory power makes them difficult to use for studying and exploring the actual traffic patterns, what-if reasoning or traffic management and control optimization.

At the other end of the spectrum we find (simulation) model-based methods. Examples include traffic flow models coupled with (extended) Kalman filters or more generally sequential Bayesian estimators [16]–[19]. The great advantage is that these methods provide an integrated solution for network wide state estimation and prediction, and that they use tractable behavioral and physical relationships, which make them highly suitable for studying and explaining traffic patterns, what-if reasoning, control optimization and application under non-recurrent conditions. The price for this explanatory power is that model-based methods are generally complex to design and maintain, and sensitive to data errors. Moreover, they require many inputs (e.g. traffic demand and control settings) and contain many parameters (driving and choice behavior) that need to be calibrated or even predicted from data. As such, model-based approaches, particularly in large networks, present many ill-posed problems and are typically highly underdetermined solutions given the available data. Even in cases we have abundant amounts of data, typically these do not encompass sufficient information (e.g. demand, behavioral relationships, traffic mix, route choices) for a large-scale model-based approach.

In this paper we explore whether a data-driven technique *can* be used to shed light on spatiotemporal traffic patterns in large-scale networks. Many network spatial analyses methods are based on the assumption that a so-called macroscopic fundamental diagram is well defined for a homogeneous region [20]. Identifying such homogeneous regions allow us to model sub networks as reservoirs with predictable characteristics. Some of the works that are based on macroscopic fundamental diagrams create these homogeneous zones using network partitioning methods such as k-means [21] or snake similarity [22]. One recent and promising approach is to incorporate time into the spatial partitioning method based on macroscopic fundamental diagram, thus creating 3D zones [23] which can be used to create 3D network states [24]. Two recent papers take a different perspective to traffic data: they analyze *network* traffic as images [25], [26]. In the first paper [25], traffic data of a corridor is converted into spatiotemporal speed map; a convolutional neural network (CNN) is then trained on these maps (images) to make a traffic speed prediction. In the second paper [26], a so-called long short-term memory (LSTM) model is used to perform the predictions. The power of these ideas is that they allow application of machine learning techniques from the computer vision domain—e.g. deep convolutional neural networks—to traffic data.

Deep convolutional neural networks, when trained, act as hierarchical detectors of features, so that the learned features get progressively more complex (from segments to lines and contours) from the first layers further into the network, whereas the last convolutional layer detects high-level features [27]. In a trained convolutional network, the features detected by the convolutional layers are then

[1]Delft University of Technology, The Netherlands; Email: [p.k.krishnakumari,o.cats,j.w.c.vanlint]@tudelft.nl
[2]ISI Foundation, Turin, Italy; Email: alan.perotti@isi.it
[3]aizoOn, Turin, Italy; Email: viviana.pinto@aizoon.it

correlated with the class labels by means of fully-connected layers. The main disadvantages of deep learning models are the computational resources needed, and the large amount of training data required to train a deep network from scratch. One of the emerging fields that try to overcome these limitations is transfer learning [28]. Transfer learning follows the intuition that many shapes and visual features are not domain-specific (e.g. a circular shape could correspond to an 'eye' in a face detector and with a 'headlight' in a vehicle detector), and therefore the features extracted from a network (trained only once on some generic-purpose dataset) can be used for different tasks.

To this end, we propose to use an opensource pretrained network to extract the relevant features from the traffic data represented as images. This is the first work to introduce transfer learning in transport, to the best of our knowledge. As of now, only limited number of features have been used to define network traffic states such as homogeneous speed regions, network connectivity, etc. From this work, the aim is to investigate whether features extracted from the pretrained model can be used to distinctly define network traffic states. These extracted features can then be used to enrich our knowledge into network traffic states. In this work, we also illustrate how these different network states can be used for look-ahead predictions, predicting the next traffic state given the current one.

The paper is organized as follows: Section II discusses the proposed clustering technique for network traffic states using pretrained models and the one-step prediction. In section III we discuss Amsterdam data and parameters used for the methods. In section IV we quantitatively and qualitatively discuss the results and conclude the paper in section V.

II. METHODS

In order to use transfer learning, we have to use already existing pretrained models. However, most of the sophisticated models that are readily available are trained on images and not traffic speed data. So, the first step is to convert the network traffic variables into meaningful images that preserves both traffic data and spatial information. Once the data is transformed into images, we use pretrained deep learning networks to extract high level visual (spatial) features of the traffic network. These features are then used to automatically identify the different network traffic states. The traffic states are identified by clustering the feature vectors using hierarchical clustering and using the medoid of the clusters to represent the traffic state of that cluster. In order to illustrate the importance of identifying these traffic states and to access the quality of the clusters for different transport applications, we train a classifier to predict the next traffic state's clustering label, given the current traffic state. Thus, for network traffic state prediction using transfer learning, there are five steps - (A) data transformation, (B) feature vector extraction, (C) network traffic state clustering, (D) medoid construction and (E) one step prediction. These steps are explained in detail below.

A. Data Transformation

Most of the pretrained deep learning models are trained on images. There are different ways to convert network traffic data into images. One such way is to construct the adjacency matrix weighted with speed and convert this matrix into an image. However, because of the scale-free property of most car traffic networks, the adjacency matrix is too sparse to extract any visible information. Furthermore, the adjacency matrix does not preserve the network nodes' relative positions (as rows and columns can be arbitrarily permuted). Another promising data representation method was introduced in [26]. The traffic data is encoded in a grid-like matrix where each grid represents a spatial region of $0.0001° \times 0.0001°$ (latitude $\times$ longitude), which is approximately 10m $\times$ 10m. In order to map the traffic data into this grid, the transportation network represented by latitude and longitude needs to be converted to a regular data grid with the aforementioned grid resolution and then the traffic data is encoded into the data grid.

To build the regular data grid, the maximum and minimum latitude and longitude of the transportation network are used to construct a boundary. A linearly spaced matrix with the given grid resolution is constructed. Now that we have the data grid, the next step is to encode the traffic data. The traffic data is usually mapped on a road stretch represented by a set of a consecutive coordinates. However, these coordinates are not uniformly distributed and might be more than the grid resolution, say $0.0001°$, apart. Therefore, the coordinates of the road stretch are evenly spaced with $0.0001°$ between the points and then the grids that intersect with these points are filled with the traffic data value of the road stretch. An illustration of the steps involved in traffic data transformation is given in Fig 1.

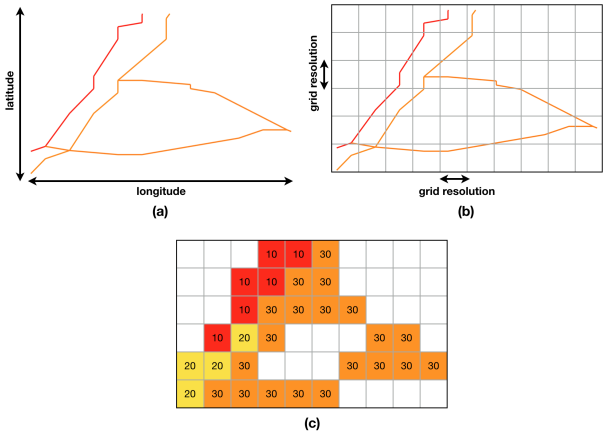


Fig. 1: Data transformation - traffic data to image (a) Sample traffic network. The color represents the traffic variable value (b) Grid of specific resolution overlayed on the network (c) Traffic data embedded in the grid.

B. Feature Vector Extraction

Transfer learning [28] can be implemented in several ways, but the common underlying approach is to use a

pretrained model and either remove the last layers(s) or retrain them on the destination dataset (a process called *fine-tuning*). In this work, we are using the first scenario of the transfer learning where we use the pretrained model as the feature extractor by removing the fully connected layers, thus keeping the output of the convolutional layers. Most of the openly available pretrained networks are trained on ImageNet [29]: the ImageNet project is an ongoing effort and currently has 14197122 images from 21841 different categories; AlexNet, VGGNet, Inception, ResNet are some of the popular pretrained networks. We used Inception-ResNet-v2 network, an improved Inception network from Google, for the feature extraction which achieves 95.1% top-5 accuracy on ImageNet [30]. The assumption is that all the relevant features that make different traffic states distinguishable are captured by the deep convolutional layers and replacing the final layers with a simple clustering technique can provide meaningful groups of traffic states.

Due to weight sharing in the convolutional layers, it is easy to run a pretrained network on images of different size [28]. We fed each image, corresponding to a snapshot of the entire network at a given time, through the Inception model to extract the features. The Inception model has 128 filters of 3×3 kernel size and 32 filters of 5×5 size [30]; the dimensional space is progressively reduced by maxpooling, eventually resulting in a feature vector of dimension 1536.

C. Network Traffic State Clustering

To distinguish different traffic states, we cluster the feature vectors obtained from the pretrained deep nets. Since there is no general rule for defining the different network states, we have no prior information on how many traffic states there are. Two obvious anticipated categories are free flow and congested network states. Other than that, the rest of the network states heavily depend on the topology of the network and the demand and supply conditions. The main questions we hope to answer through the clustering is first, if the feature vector are distinguishable for obvious categories and second, if the feature vector can satisfactorily identify different and meaningful traffic states.

There are lot of options for feature vector classification. However, given that we do not have a priori information of the distribution of the types of network states, we opted for unsupervised hierarchical clustering. Hierarchical clustering builds a hierarchy of states based on the dissimilarity between them. By looking at the hierarchy construction, we can make a more informed decision about the number (and nature) of traffic states for a given network. The hierarchy can be constructed in two ways - agglomerative and divisive [31]. Agglomerative clustering is a bottom-up approach where all the samples start as a single cluster and then these clusters are merged together based on a distance dissimilarity measure to form new clusters, thus building up the hierarchy. In the top-down divisive clustering, all the observations or samples start in one cluster and are then split based on a distance dissimilarity measure for building the hierarchy.

In this work, we are using agglomerative clustering with the Euclidean distance as the dissimilarity distance measure. This method initiates each feature vector x_k as its own cluster. The connectivity between any two feature vectors, x_i, x_j with N dimensions, is calculated using Euclidean distance given by:

$$d(x_i, x_j) = \sqrt{\sum_{k=1}^{N} (x_i^k - x_j^k)^2} \tag{1}$$

To measure the distance between two clusters, the average-link scheme is implemented which takes average Euclidean distance between all feature vector pairs from those clusters. Then, the two closest clusters will be merged to form a larger cluster. This process continues until only one cluster remains. The results of the hierarchical clustering is usually presented in a dendogram [32], a tree diagram illustrating the arrangement of the clusters. This provides a better overview of the structure of the hierarchy to judge the optimal number of clusters for this task.

D. Medoid Construction

Now that we have the number of clusters, the aim is to construct a representative feature vector for each cluster. The most common one is centroids or mean of the clusters. But this leads to creating new data points which is unrealistic for traffic data. Instead, we are constructing a medoid of the cluster which corresponds to an actual data point in that cluster [33]. The medoid is chosen such that the average dissimilarity to all the objects within that cluster is minimized. Given a set of n feature vectors $x_1, x_2, ...x_n$ with N-dimensional real vectors and a dissimilarity Euclidean distance function d, the medoid x_{medoid} is defined in equation 2.

$$x_{medoid} = \underset{y \in \{x_1, x_2, ..., x_n\}}{\arg\min} \sum_{i=1}^{n} d(y, x_i) \tag{2}$$

E. One-Step Ahead Prediction

In order to illustrate how this kind of clustering can be used for real time applications, we demonstrate a rudimentary one-step prediction of traffic state in this paper. Given the current traffic state, we predict the next traffic state using the feature vectors and cluster labels. The accuracy of the prediction is a measure of the usefulness of the feature vectors in defining traffic states and the quality of the clustering.

For the one step prediction, we are using an ensemble of classifiers: Multi-Layer Perceptron, K-Nearest Neighbors, Random Forest, Support-Vector Machine, and a Gaussian Process [34]. First, we randomly split the dataset into a training and a test set (encompassing 80% and 20% of the data, respectively). Each classifier is trained on the feature vectors of time n with the cluster label of time $n+1$ as the output. Note how this approach can be extended for k-lookahead prediction (simply by pairing the features vectors of time n with the labels of time $n+k$) and for taking into account the recent history as well (for instance, by including the feature vectors of time $n-1, n-2, ..., n-k$ as inputs).

Each classifier has hyperparameters that were optimized by means of grid-search and random-search. Each trained classifier, when given an image from the validation set, outputs a probability vector with size equal to the number of clusters. For our prediction task, each element i of this vector corresponds to the estimated likelihood, for the traffic state determined by the input image and according to the trained classifier, that the state will evolve into a state belonging to the i-th cluster. The predictions of the five classifiers are then composed by means of the majority rule, thus creating an ensemble classifier.

A confusion matrix (also called a contingency matrix) [35] is used to analyze the accuracy of this global prediction against the ground truth. In this study, the ground truth is the hierarchical clustering label for the following time slice.

III. Experimental Setup

The data used in this study is travel time data collected from license plate recognition systems at different critical points of the major street network of Amsterdam, The Netherlands. The shortest path between these points was mapped on Open Street Maps (OSM) to create the network shown in Fig 2(a). This mapped Amsterdam network comprises of 7512 links and 6528 nodes, excluding freeways. The travel time is converted into mean speed per link in the network for every 10 minutes between 00:00 AM and 23:50 PM for the 7512 links. Thus, there are 144 time slices for each day. The data is available for 42 days from 22 February 2015 to 4 April 2015. For more details about data preparation for creating the mapped network and converting travel time to speed, we refer to [23].

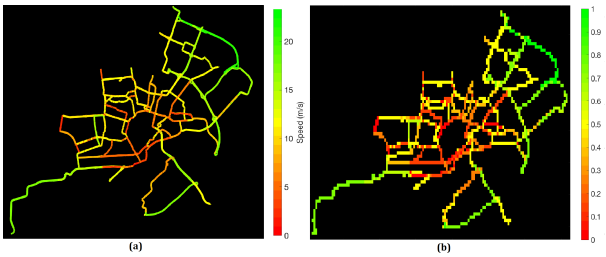


Fig. 2: Data transformation of one time slice of Amsterdam network (a) Amsterdam network with latitude as x-axis, longitude as y-axis and the color represents the speed in m/s at that time slice (b) Corresponding data matrix with grid resolution $0.001°$. Speed is normalized between 0 and 1.

For the data transformation, we used a grid density of $0.001°$ for both latitude and longitude, thus creating images of resolution 143×286 as shown in Fig 2(b). Some of the time slices in the 42 days have no observations due to faulty equipment and missing data. After removing these time slices, we have 5775 images for the 42 days. Inception-Resnet-v2 was trained on images that were scaled to 0 to 1. In order to have consistent data, we used a maximum speed of 25m/s (90km/hr) to normalize all the images, corresponding to the highest speed limit in the case study network.

IV. Results and Discussion

This section presents the results of the clustering and the prediction. The feature vector of 1536 dimensions are obtained from the Inception-Resnet-v2 model for the 5775 images. Thus, we reduced the complexity from 7512 to 1536 for a single time slice, approximately 80% reduction. Since some of these features might not be informative for the traffic problem, a further direction of work would be to further reduce the feature vector dimension either by investigating which of the layers in the deep network is returning a constant output for all images or using dimensionality reduction methods such as principal component analysis (PCA), linear discriminant analysis (LDA), etc. In this paper, we use the 1536 dimensions to analyze the network states.

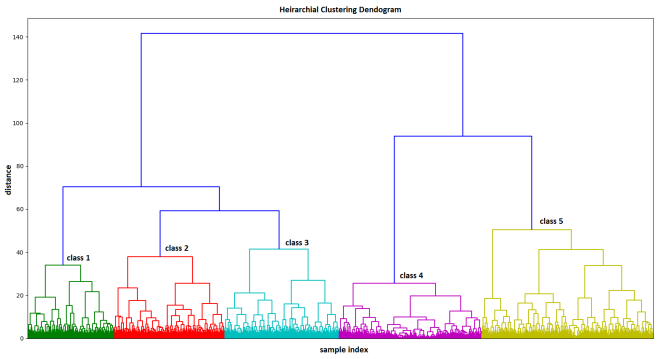
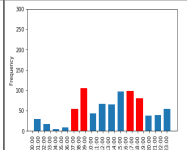
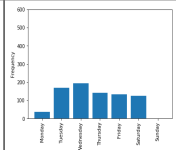
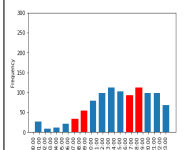
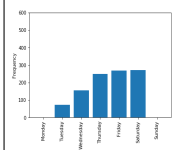
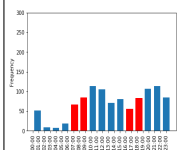
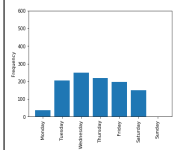
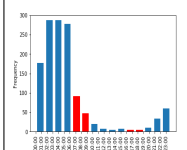
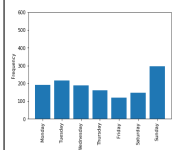
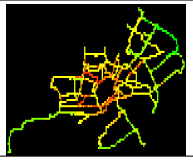
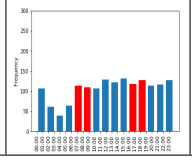
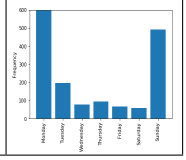


Fig. 3: Dendrogram of the feature vectors

The dendogram of the hierarchical clustering of these feature vectors is presented in Fig 3. From the dendogram, the feature vectors are clearly distinguishable. The vertical height is an indication of the distance between the individual data points (feature vectors) or the clusters. We decided to set the number of clusters to 5 for understanding the different network states in each cluster. Each cluster is represented by a different color in the dendogram structure as shown in Fig 3. The cluster size can be modified per the application requirement.

From the dendogram, we can observe that there are clearly two distinct branches comprising of (i) classes 1,2,3 and (ii) classes 4,5; and even further down the dendogram, there is clear separability. A more in-depth assessment on these classes is given in Table I. From the medoid of these classes in Table I, we can broadly identify these two branches as congested and free flow branch respectively. The congested branch comprising of classes 1,2,3 are more closer in feature space compared to the free flow classes as seen in the dendogram. This can be confirmed visually as well. It is hard to visually judge the difference between the medoid of these classes (Table I) without additional meta information such as a difference images of the medoids or quantitative measures such as the average speed of the medoid of each class. A closer inspection does provide insight into the difference which is related to the difference in the spread of congestion into the network links. Fig 4 provides a quantitative variability between and within the classes - the

TABLE I: Description of Amsterdam network traffic states.

Class	Medoid	Distribution of time slices in each class	Distribution of days in each class
1			
2			
3			
4			
5			

distribution of the average speeds in each class. Even though the congested classes are similar, there is clearly variability in speeds and network structure that was captured by the deep networks. Thus, we broadly defined the class 1,2,3 as severe congestion network state, class 4 as free flow and class 5 as free flow with mild congestion. Based on the application, merging or further division of classes can be done. In this work, we proceed with 5 classes as the aim is not to find optimum number of classes but rather to see if predictable traffic pattern emerges.

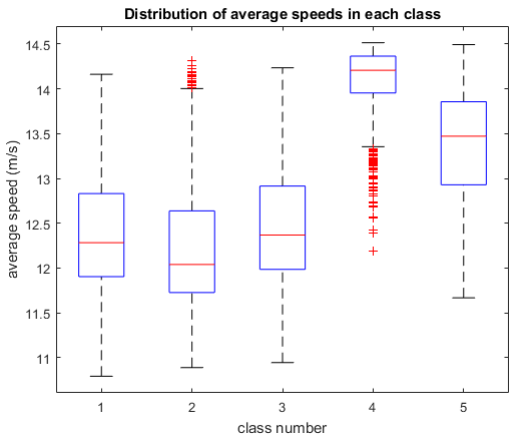


Fig. 4: Variability of speeds in each timeslice within classes

A more in-depth look at the clusters also provides insight into the temporal differences among the clusters. The third column of the Table I shows the distribution of the time slices in each cluster. The first and second set of red bar plots corresponds to the morning (6.30 am to 9 am) and afternoon (4.30pm to 6pm) peak period in the Netherlands respectively. It can be seen that the afternoon peak period is more severe than the morning peak period for these 42 days as the occurrence of free flow class 4 for the afternoon peak period is significantly low. The most frequent congestion pattern during both the peak periods is the class 5, implying mild congestion is the regular network state in Amsterdam. Further study can reveal which network state corresponds to special or non-recurrent incidents, but this requires reliable incident data.

This study also reveals the day-to-day regularity in the congestion patterns by examining the distribution of days within each cluster, shown in the column 4 of Table I. It is clear that severe congestion is absent on Sundays. Only class 4 and 5 network states occurred on Sundays within the 42 days. Another find was that Monday have less severe congestion for a working day with class 5 as its most recurrent network state. Monday is comparatively more similar to Sundays than the weekdays. Among the weekdays, Friday is the most severe in terms of ratio of frequency of congestion to free flow condition occurrences.

TABLE II: 80-20 one-step prediction results

	PREDICTED				
KNOWN	Class 1	Class 2	Class 3	Class 4	Class 5
Class 1	0.28	0.23	0.32	0.07	0.10
Class 2	0.07	0.63	0.23	0.07	0.00
Class 3	0.13	0.29	0.47	0.05	0.06
Class 4	0.00	0.02	0.06	0.84	0.08
Class 5	0.02	0.03	0.08	0.17	0.70

We present one of the application of network state clustering in this study - one-step prediction. This shows how time dimension can easily be integrated. Table II presents the 80 - 20 cross-validation results of the one-step prediction. It gives an average accuracy of approximately 58% using the naive classification approach. Even though the accuracy is not high, the classifier is clearly learning some patterns emerging in the temporal dimension using these feature vectors. More complex time series based methods, such as LSTM, can be used to encode the evolution of the feature vectors in the temporal dimension for better accuracy.

V. CONCLUSION AND FURTHER RESEARCH

In this paper, we introduced transfer learning to solve the problem of spatial clustering in the traffic domain. The assumption is that even though the pretrained networks are trained on natural images, the approach can be generalized to create feature vectors that can used to identify meaningful network states. This is proven in this study. The data dimension is reduced from 7512 to 1536, a reduction of 80% - simply by presenting data-as-images to pretrained neural networks. This process requires no training, and therefore

is fast, scalable, and does not suffer from overfitting. This kind of analysis opens up many possibilities for new ways of looking at network traffic from the perspective of computer vision.

There are various promising future directions for this study. One of the main directions is to unravel the deep network to understand which filters are aiding in identifying these patterns. This can help in further reducing the dimensionality of the feature vectors and also in understanding some of the known or unknown traffic phenomena that the deep network identified in its 160 filters. Another topic that is interesting for future research is to look into 3D deep networks with 3D kernels or filters so that the temporal dimensions are incorporated into the deep network rather than approaching it as two step process - spatial, then temporal or vice versa. The spatiotemporal feature vectors could provide more insight into spatiotemporal evolution of congestion in a network. Since we have successfully applied transfer learning for understanding spatial patterns in network traffic patterns, other potential applications can be public transport network analysis, water network or any other domain with spatiotemporal data.

ACKNOWLEDGMENT

The authors would like to thank Sjoerd Linders and AMS for providing the data for this study. This research has been performed as part of the SETA project funded by the European Union's Horizon 2020 Research and Innovation program under the grant agreement No 688082. Map data copyrighted to OpenStreetMap contributors. We thank the anonymous reviewers for their many constructive remarks.

REFERENCES

[1] S. Clark, "Traffic prediction using multivariate nonparametric regression", *Journal of Transportation Engineering*, vol. 129, no. 2, pp. 161-168, 2003.

[2] J. Rice and E. Van Zwet, "A Simple and Effective Method for Predicting Travel Times on Freeways", in *Proceedings of the IEEE Conference on Intelligent Transportation Systems*, Oakland, CA, United States, pp. 227-232, 2001.

[3] Y. Xu, H. Chen, Q. J. Kong, X. Zhai, and Y. Liu, "Urban traffic flow prediction: A spatio-temporal variable selection-based approach", *Journal of Advanced Transportation*, vol. 50, no. 4, pp. 489-506, 2016.

[4] C.-H. Wu, C.-C. Wei, D.-C. Su, M.-H. Chan, and J.-M. Ho, "Travel Time Prediction with Support Vector Regression", in *Proceedings of the 2003 IEEE Conference on Intelligent Transportation Systems*, Shanghai, China, 2003.

[5] J. Wang, I. Tsapakis, and C. Zhong, "A space-time delay neural network model for travel time prediction", *Engineering Applications of Artificial Intelligence*, vol. 52, pp. 145-160, 2016.

[6] Y. Zhang and H. Ge, "Freeway travel time prediction using takagi-sugeno-kang fuzzy neural network", *Computer-Aided Civil and Infrastructure Engineering*, vol. 28, no. 8, pp. 594-603, 2013.

[7] J. W. C. van Lint, "Online learning solutions for freeway travel time prediction", *IEEE Transactions on Intelligent Transportation Systems*, vol. 9, no. 1, pp. 38-47, Mar 2008.

[8] E.I. Vlahogianni, J.C. Golias, M.G. Karlaftis, "Short-term traffic forecasting: Overview of objectives and methods", *Transport reviews*, 24(5), pp.533-557, 2004.

[9] J.W.C. Van Lint, C.P.I.J. Van Hinsbergen, "Short-term traffic and travel time prediction models", *Artificial Intelligence Applications to Critical Transportation Issues*, 22, pp.22-41, 2012.

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

[11] S. Oh, Y.J. Byon, K. Jang, H. Yeo, "Short-term travel-time prediction on highway: a review of the data-driven approach", *Transport Reviews*, 35(1), pp.4-32, 2015.

[12] H. Yang, T. S. Dillon, and Y. P. Chen, "Optimized Structure of the Traffic Flow Forecasting Model With a Deep Learning Approach", *IEEE Transactions on Neural Networks and Learning Systems*, 2016.

[13] A. Koesdwiady, R. Soua, and F. Karray, "Improving Traffic Flow Prediction with Weather Information in Connected Cars: A Deep Learning Approach", *IEEE Transactions on Vehicular Technology*, vol. 65, no. 12, pp. 9508-9517, 2016.

[14] Y. Lv, Y. Duan, W. Kang, Z. Li, and F. Y. Wang, "Traffic Flow Prediction with Big Data: A Deep Learning Approach", *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 2, pp. 865-873, 2015.

[15] N.G. Polson, V.O. Sokolov, "Deep learning for short-term traffic flow prediction", *Transportation Research Part C: Emerging Technologies*, 79, pp.1-17, 2017.

[16] R. Wang, S. Fan, and D. B. Work, "Efficient multiple model particle filtering for joint traffic state estimation and incident detection", *Transportation Research Part C: Emerging Technologies*, vol. 71, pp. 521-537, 2016.

[17] Y. Yuan, J. W. C. van Lint, R. E. Wilson, F. van Wageningen-Kessels, and S. P. Hoogendoorn, "Real-Time Lagrangian Traffic State Estimator for Freeways," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 1, pp. 59-70, Mar 2012.

[18] C. P. I. J. van Hinsbergen, T. Schreiter, F. S. Zuurbier, J. W. C. van Lint, and H. J. van Zuylen, "Localized Extended Kalman Filter for Scalable Real-Time Traffic State Estimation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 1, pp. 385-394, Mar 2012.

[19] Wang Y., Papageorgiou M., and Messmer A., "A real time freeway network traffic surveillance tool," *IEEE Transactions on control systems technology*, vol. 14, no. 1, 2006.

[20] N. Geroliminis, C.F. Daganzo, "Existence of urban-scale macroscopic fundamental diagrams: Some experimental findings", *Transportation Research Part B: Methodological*, 42(9), pp.759-770, 2008.

[21] Y. Ji, N. Geroliminis, "On the spatial partitioning of urban transportation networks", *Transportation Research Part B: Methodological*, 46(10), pp.1639-1656, 2012.

[22] M. Saeedmanesh, N. Geroliminis, "Clustering of heterogeneous networks with directional flows based on "Snake" similarities", *Transportation Research Part B: Methodological*, 91, pp.250-269, 2016.

[23] C. Lopez, P. Krishnakumari, L. Leclercq, N. Chiabaut, H. Van Lint, "Spatiotemporal Partitioning of Transportation Network Using Travel Time Data", *Transportation Research Record: Journal of the Transportation Research Board*, (2623), pp.98-107, 2017.

[24] C. Lopez, L. Leclercq, P. Krishnakumari, N. Chiabaut, H. Lint, "Revealing the day-to-day regularity of urban congestion patterns with 3D speed maps", *Scientific Reports*, 7(1), p.14029, 2017.

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

[26] H. Yu, Z. Wu, S. Wang, Y. Wang, X. Ma, "Spatiotemporal recurrent convolutional networks for traffic prediction in transportation networks", *Sensors*, vol. 17(7), pp. 1501, 2017.

[27] I. Goodfellow, Y Bengio, A Courville, "Deep Learning", *The MIT Press*, 2016.

[28] K. Weiss, T.M. Khoshgoftaar, D. Wang, "A survey of transfer learning", *Journal of Big Data*, 3(1), pp.9, 2016.

[29] A. Krizhevsky, I. Sutskever, G.E. Hinton, "ImageNet classification with deep convolutional neural networks", in *Proceedings of the 25th International Conference on Neural Information Processing Systems*, Lake Tahoe; Nevada, USA, pp. 1097-1105, 2012.

[30] C. Szegedy, S. Ioffe, V. Vanhoucke, A.A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning", *AAAI*, Vol. 4, pp. 12, 2017.

[31] L. Rokach, O. Maimon, "Clustering methods", *Data mining and knowledge discovery handbook*, pp. 321-352, Springer, Boston, MA, 2005.

[32] B. Everitt, A. Skrondal, "The Cambridge dictionary of statistics", *Cambridge University Press Cambridge*, Vol. 106, 2002.

[33] A. Struyf, M. Hubert, P. Rousseeuw, "Clustering in an object-oriented environment", *Journal of Statistical Software*, 1(4), pp.1-30, 1997.

[34] A. Geron, "Hands-On Machine Learning with Scikit-Learn and TensorFlow", *O'Reilly Media*, 2017.

[35] T. Fawcett, "An introduction to ROC analysis", *Pattern recognition letters*, 27(8), pp.861-874, 2006.