Deep Learning: A Generic Approach for Extreme Condition Traffic Forecasting

Rose Yu*† Yaguang Li*† Cyrus Shahabi† Ugur Demiryurek† Yan Liu†

Abstract

Traffic forecasting is a vital part of intelligent transportation systems. It becomes particularly challenging due to short-term (e.g., accidents, constructions) and long-term (e.g., peak-hour, seasonal, weather) traffic patterns. While most of the previously proposed techniques focus on normal condition forecasting, a single framework for extreme condition traffic forecasting does not exist. To address this need, we propose to take a deep learning approach. We build a deep neural network based on long short term memory (LSTM) units. We apply Deep LSTM to forecast peak-hour traffic and manage to identify unique characteristics of the traffic data. We further improve the model for post-accident forecasting with Mixture Deep LSTM model. It jointly models the normal condition traffic and the pattern of accidents. We evaluate our model on a real-world large-scale traffic dataset in Los Angeles. When trained end-to-end with suitable regularization, our approach achieves 30%-50% improvement over baselines. We also demonstrate a novel technique to interpret the model with signal stimulation. We note interesting observations from the trained neural network.

1 Introduction

Traffic forecasting is the core component of the intelligent transportation systems (ITS). The problem has been studied for decades in various communities ranging from transportation system (e.g. [19, 23]), through economics (e.g. [18, 6]), to data mining (e.g.[14, 13]). While normal condition traffic patterns are easy to predict, an open question in traffic forecasting is to forecast traffic for extreme conditions, which include both peak hour and post-accident congestions. National statistics shows that in 2013, traffic congestion costs Americans $124 billion direct and indirect loss. The county of Los Angeles, whose traffic data is studied in the paper, has continuously been ranked as one of the most traffic-congested cities in North America, costing drivers to spend 34% more time on their routes as compared to normal traffic condition [5]. Therefore, accurate traffic forecasting of extreme conditions can substantially enhance traffic control, thereby reducing congestion cost, and leading to significant improvement in societal welfare.

The task is challenging mainly due to the chaotic nature of traffic incidents. On one hand, recurring incidents like peak-hours would cause steep drop in traffic speed, leading to non-stationary time series. On the other hand, non-recurring incidents like accidents are almost unpredictable, introducing unexpected delay in traffic flow. Traditional methods are mostly limited to linear models. They perform well for normal conditions but forecast poorly for extreme conditions. For example, historical average depends solely on the periodic patterns of the traffic flow, thus can hardly be responsive to dynamic changes. Auto-regressive integrated moving average (ARIMA) time series models rely on stationary assumption of the time series, which would be violated in the face of abrupt changes in traffic flow [4]. Applying neural network to capture the non-linearity in traffic flow has been studied in the early days [19, 4, 11]. However, the models studied were single layer networks with few hidden units.

The deep learning approach provides automatic representation learning from raw data, significantly reducing the effort of hand-crafted feature engineering. For traffic forecasting, early attempts include deep belief network (DBN) [10], stacked autoencoder [15] and stacked denoising autoencoder [3]. However, they fail to capture temporal correlations. Deep recurrent neural network, with great promise in representing dynamic behavior, has recently achieved massive success in sequence modeling, especially for video segmentation [12] and speech recognition [20]. In this paper, we take a very pragmatic approach to investigate and enhance the recurrent neural network by studying a large-scale and high-resolution transportation data from LA County road network. The datasets are acquired in real time from various agencies such as CalTrans, City of LA Department of Transportation, California Highway Patrol and LA Metro. The data include

*Authors have equal contributions.

†Department of Computer Science, University of Southern California, {qiyu,yaguang,shahabi,demiryur,yanliu.cs}@usc.edu

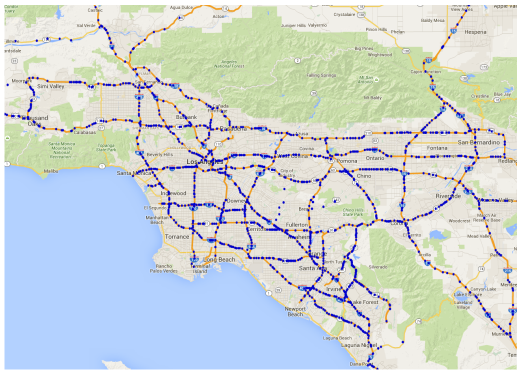


Figure 1: Visualization of the geographic distribution of loop detector sensors in LA/OC area

traffic flow records from under-pavement loop detectors as well as accidents information from police reports. The geographical distribution of those loop detectors is visualized in Figure 1.

We start with a state-of-the-art model leveraging Long Short Term Memory (LSTM) recurrent neural network. Working with real world traffic data, we identify several of their unique characteristics. First, directly feeding in traffic flow sequence is not enough for long-term forecasting during peak-hour. Time stamp features such as *time of day* and *day of week* are important for accurate peak-hour forecasting. Normalization and missing data imputation are also critical for stable training. Second, post-accident traffic can be better modeled as a mixture of normal traffic flow patterns and accident-specific features. Towards this end, we propose the *Mixture Deep LSTM* neural network, which treats the normal traffic as a background signal and interruptions from accidents as event signals. The outputs of the two components are concatenated and then aggregated through a regression layer. Third, LSTM learns the traffic patterns by "memorizing" historical average traffic as well as the interruption caused by accidents. The stacked autoencoder component of the proposed mixture model performs denoising on accident-specific features. Figure 2 illustrates the pipeline of our learning procedure.

When trained end-to-end on real world traffic data, our approach shows superior performance on both forecasting tasks, with forecasting error reduced by 30% - 50% as compared to those of the baseline models. Qualitative study shows that the learned model is able to "memorize" the periodic patterns in traffic flow as well as the accident effects. In summary, our contributions can be stated as follows:

- We investigate the deep learning approach for extreme conditions traffic forecasting. In particular, we enhance the Deep LSTM network with careful data cleaning and feature engineering. We augment the Deep LSTM model with stacked autoencoder to account for the accident features.
- We apply our framework to forecast traffic speed in peak-hour and post-accident conditions. The architecture can model both the normal traffic patterns and the dynamics of extreme conditions. We observe superior performance of the proposed model for both tasks.
- We also perform model inspection to interpret the learning process. We observe that our model seems to memorize the historical average speed as well as the interruptions caused by accidents.

2 Related Work

A large body of the related work for traffic forecasting lies in transportation research, see a recent survey paper [24] and the references therein. Another line of work touches upon incidents and their impact, with focus on accident delay prediction. For example, in [7], a multivariate regression model is developed based on predictors such as the number of lanes affected, the number of vehicle involved and the incidence duration. The setting is questionable as the traffic delay caused by incidence can only be predicted after the incident is cleared. More recently, in [17], the authors hand-craft a set of features to predict the time-varying spatial span of the incident impact.

Applying neural networks to traffic forecasting is not a new idea. But most of them only consider neural networks with single hidden layer and few hidden units, which have limited representation power. For example, in [4], the authors conduct a careful comparison between neural network and time series models such as ARIMA. The study concludes that statistical techniques generally give slightly better performance for short term traffic forecasting. Similar observations are presented

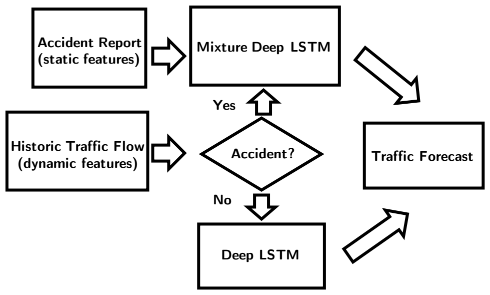

Accident Report (static features)
Mixture Deep LSTM
Yes
Historic Traffic Flow (dynamic features)
Accident?
Traffic Forecast
No
Deep LSTM

Figure 2: Training pipeline of the proposed deep learning approach for extreme condition traffic forecasting

in [11], which studies two hours ahead forecasting using data recorded every 30 minutes. But the model suffers from noticeable performance drop as the forecasting horizon increases. In [22], a single layer RNN is used for travel time prediction, but with unsatisfactory performance. In [16], the authors proposes to use single layer LSTM network for short term traffic forecasting. Early computers did not have enough processing power to effectively handle the long running time required by training large neural networks.

A pioneering work of applying deep learning for traffic forecasting is proposed in [10]. The authors combine DBN with multi-task learning. The model contains two steps: feature learning and forecasting. DBN serves as unsupervised pre-training step. The output of the DBN is fed into a regression layer for forecasting. The empirical results show improved performance over traditional methods, especially for long term and rush hour traffic flow prediction. However, DBN fails to account for the temporal dependence in speed time series. In the input layer, DBN assumes the speed reading at each time stamp is independent and the learned representation can hardly reflect the complex dynamics in traffic flow.

We make the first attempt to apply deep LSTM recurrent neural network for traffic forecasting. LSTM has recently been popularized through success in machine translation [20], image caption generation [25] and clinical diagnosis [2]. An attractive property of the LSTM is that it is capable of learning both long term and short term temporal dependencies. With careful data cleaning and normalization, we manage to obtain the up to 1 hour ahead forecasting with less than 10% error. When applied to accident forecasting, our architecture merges the outputs from the two components, instead of stacking them together.

3 Methodology

In this section, we first introduce the basic concepts in neural networks. Then we study two extreme condition traffic forecasting tasks: peak-hours and post-accidents. For each task, we provide problem description, model specification and training details.

3.1 Basic Concepts

Autoencoder An autoencoder takes an input vector $\mathbf{x}$ and transforms it into a hidden representation $\mathbf{h}$. The transformation, typically referred as encoder, follows the following equation:

$$\mathbf{h} = \sigma(\mathbf{W}\mathbf{x} + \mathbf{b})$$

where $\mathbf{W}$ and $\mathbf{b}$ correspond to input to hidden weights and the bias.

The resulting hidden representation $\mathbf{h}$ is then mapped back into the reconstructed feature space $\mathbf{y}$ as follows:

$$\mathbf{y} = \sigma(\mathbf{W}'\mathbf{h} + \mathbf{b}')$$

$\mathbf{W}, \mathbf{W}', \mathbf{b}, \mathbf{b}'$ denote the weight and bias respectively, and σ is the sigmoid function (or the soft-max function when dealing with multivariate case). This procedure is called decoder. Autoencoder is trained by minimizing the reconstruction error $\|\mathbf{y}-\mathbf{x}\|$. Multiple autoencoders can be connected to construct stacked autoencoder, which can be used to learn multiple levels of non-linear features.

Recurrent Neural Network (RNN) Recurrent neural network is a feature map that contains at least one feed-back loop. Denote the input vector at time stamp t as $\mathbf{x}_t$, the hidden layer vector as $\mathbf{h}_t$, the weight matrices as $\mathbf{W}_h$ and $\mathbf{U}_h$, and the bias term as $\mathbf{b}_h$. The output sequence $\mathbf{o}_t$ is a function over the current hidden state. RNN iteratively computes the hidden layer and outputs using the following recursive procedure:

$$\mathbf{h}_t = \sigma(\mathbf{W}_h \mathbf{x}_t + \mathbf{U}_h \mathbf{h}_{t-1} + \mathbf{b}_h)$$

and

$$\mathbf{o}_t = \sigma(\mathbf{W}_o \mathbf{h}_t + \mathbf{b}_o)$$

Here $\mathbf{W}_o$ and $\mathbf{b}_o$ represent the weight and bias for the output respectively.

Long Short Term Memory LSTM is a special type of RNN that is designed to avoid the vanishing gradient issue in the original RNN model. It replaces the ordinary summation unit in RNN with carefully designed memory cell which contains gates to protect and control the cell state [9]. The key to LSTMs is the cell state, which allows information to flow along the network. LSTM is able to remove or add information to the cell state, carefully regulated by structures called gates, including input gate, forget gate and output gate. Denote $\mathbf{i}_t$, $\mathbf{f}_t$, $\mathbf{o}_t$ as input gate, forget gate and output gate at time t respectively. Let $\mathbf{h}_t$ and $\mathbf{s}_t$ be the hidden state and cell state for memory cell at time t. The architecture of the LSTM is specified as follows.

$$\begin{aligned}
\mathbf{i}_t &= \sigma(\mathbf{W}_i \mathbf{x}_t + \mathbf{U}_i \mathbf{h}_{t-1} + \mathbf{b}_i) \\
\mathbf{c}_t &= \tanh(\mathbf{W}_c \mathbf{x}_t + \mathbf{U}_c \mathbf{h}_{t-1} + \mathbf{b}_c) \\
\mathbf{f}_t &= \sigma(\mathbf{W}_f \mathbf{x}_t + \mathbf{U}_f \mathbf{h}_{t-1} + \mathbf{b}_f) \\
\mathbf{o}_t &= \sigma(\mathbf{W}_o \mathbf{x}_t + \mathbf{U}_o \mathbf{h}_{t-1} + \mathbf{b}_o) \\
\mathbf{s}_t &= \mathbf{s}_{t-1} \circ \mathbf{f}_t + \mathbf{c}_t \circ \mathbf{i}_t \\
\mathbf{h}_t &= \mathbf{s}_t \circ \mathbf{o}_t
\end{aligned}$$

where $\circ$ denotes Hadamard product.

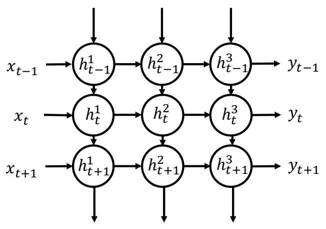


Figure 3: Graphic illustration of the Deep LSTM Network

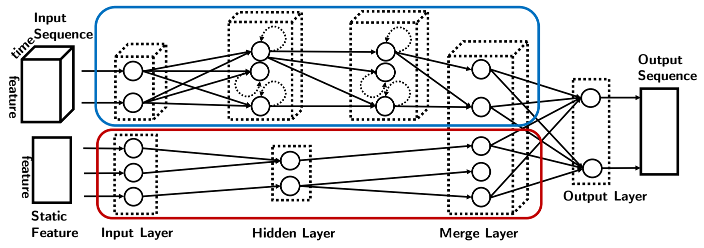


Figure 4: Graphic illustration of mixture deep LSTM. Component in blue rectangle is deep LSTM and component highlighted in red rectangle is stacked autoencoder

3.2 Peak-hour Traffic Forecasting Peak-hour is the period when traffic congestion on roads hits its highest. It normally happens twice every weekday. In this paper, we define peak-hour periods as 6-9 am and 4-7 pm. During peak-hour congestion, the amount of uncertainty in truck traffic and intersection turning movements make it difficult to accurately predict the traffic flow. The majority of existing traffic forecasting models are designed for short-term forecasting, usually 5-15 minutes ahead during peak-hour. In addition, the forecasting is usually limited to single time stamp with fixed forecasting horizon.

The re-occurring nature of peak-hour traffic motivates us to take advantage of its long term history. We extract historical traffic reading sub-sequences as input features. We treat the future time stamps as output. Input-output pairs are generated by sliding a fixed length window along the entire time series. This moving window approach is commonly adopted by time series analysis community. However, classic time series models such as Autoregressive moving average (ARMA) or Autoregressive integrated moving average (ARIMA) can only capture the linear temporal dependency of the sequences. We tackle this challenge with deep neural network, which automatically learns the non-linear structure using multiple layers of transformation. We start with the Deep LSTM network.

3.2.1 Deep LSTM Deep LSTM network is a type of RNN with multiple hidden layers. It uses the LSTM cell to replace conventional recurrent unit, thus avoiding the vanishing gradient issue of traditional RNN. Deep LSTM is able to learn long-term dependencies. Comparing with single layer LSTM, deep LSTM naturally employs a temporal hierarchy with multiple layers operating at different time scales [8]. Similar models have been applied to solve many real-life sequence modeling problems [20, 25]. Figure 3 shows the standard structure of the Deep LSTM network.

When applying Deep LSTM for traffic forecasting, we find that time stamp features such as *time of day* and *day of week* are important for accurate peak-hour forecasting. We also normalize the input sequences to zero mean and unit variance for faster convergence. We modify the training objective function to ignore the zero-valued missing observations. In this work, we model individual sensor readings and train them independently. One can also exploit sensor-sensor correlation to model all the sensor jointly . We examined the case by training 10 sensors all together. However, due to the instability of the gradient in the training, we were not able to obtain improved results for jointly training case.

3.3 Post-Accident Traffic Forecasting Traffic accidents interrupt the periodic patterns of daily traffic. We are interested in predicting traffic flow immediately after the accident occurs. The duration of the recovery depends on incident factors such as accident severity and up/downstream traffic. We formalize the problem as to predict the sequence after accident using the historical sequence. We also wish to utilize accident-specific features including severity, location and time.

We first tried to use deep LSTM by feeding in the sequences right before the accidents and generating the sequences after, but were not able to obtain satisfactory results. We hypothesize this is due to the common delay of the accident reports. The input sequences already contain the interruptions caused by the accidents. Thus the neural network can hardly differentiate between the normal traffic pattern and effect caused by accidents. Our next solution is motivated by the real world traffic accident scenario. If we treat the traffic in normal conditions as a background signal and the interruption from extreme incidents as an event signal, the traffic flow in extreme conditions can be modeled as the mixture of the two.

3.3.1 Mixture Deep LSTM It is desirable to have a framework to model the joint effects of normal traffic and accidents. As LSTM can capture long term temporal dependency, we use it for normal traffic modeling. For accidents, we utilize autoencoder to extract the latent representations of those static features. We stack multiple layers of LSTM and autoencoder and combine them at the end using a linear regression layer. As shown in Figure 4, the model consists of two components. One is the stacked LSTM layers, highlighted in the blue rectangle. LSTM models the temporal correlations of the traffic flow of multiple scale. The other is the stacked autoencoder, which learns the latent representation that are universal across accidents. This leads to our design of the Mixture Deep LSTM network.

Mathematically speaking, the model takes the input of N sequence of length T and dimension D_1, which naturally forms a tensor $\mathcal{X} \in \mathbb{R}^{N \times T \times D_1}$. In addition, the model is provided with N features of dimension D_2, denoted as $X' \in \mathbb{R}^{N \times D_2}$. The stacked autoencoder output $Y' \in \mathbb{R}^{N \times D_3}$ is duplicated over time dimension, resulting a tensor $\mathcal{Y}' \in \mathbb{R}^{N \times T \times D_3}$. At the merge layer, a sequence of dimension $D_1 + D_3$ is generated by concatenating the output from deep LSTM and stacked autoencoder. Then the output layer aggregates the concatenated sequence into the desired results. The model outputs a sequence of $N \times T'$ where T' is the length of the output sequence.

When applying Mixture Deep LSTM to post-accident traffic forecasting, we use as input the sequences one week before the accidents to approximate the normal traffic pattern, which addresses the delay issue in the accident reports. We feed the accident-specific features into the autoencoder input layer to describe the interruptions caused by the accidents. We unify the two components by duplicating the output from stacked autoencoder across each timestamp of the input sequence. The resulting sequence is then concatenated with the sequence generated by deep LSTM. Similar to the regression architecture in deep LSTM model, the model aggregates the concatenated sequence from deep LSTM and the stacked autoencoder component at the merge layer. Finally, a fully-connected layer is applied to transform the concatenated sequence into the desired output sequence.

4 Experiments

We compare the proposed approach with competitive baselines for the two forecasting tasks in extreme traffic conditions and present both quantitative and qualitative results of the proposed framework.

Traffic Data. The traffic data used in the experiment are collected by $2,018$ loop detectors, located on the highways and arterial streets of Los Angeles County (covering 5400 miles cumulatively) from May 19, 2012 to June 30, 2012. We aggregate the speed readings every 5 minutes. As there is a high ratio of missing values in the raw data, we filter out mal-functioning sensors with more than 20% missing values.

Accident Data. The accident data are collected from various agencies including California Highway Patrol (CHP), LA Department of Transportation (LADOT), and California Transportation Agencies (CalTrans). We select accidents of three major types: Rolling, Major Injuries and Minor Injuries. The total number of accidents is 6,811, spreading across 1,650 sensors. Each accident is associated with a few attributes such as accident type, downstream post mile, and affected traffic direction.

4.1 Baselines

For peak-hour forecasting, we compare our model with widely adopted time series analysis models. 1) ARIMA: Auto-Regressive Integrated Moving Average model that is widely used for time series prediction; 2) Random Walk: the traffic flow is modeled as a constant with random noise; 3) Historical Average: the traffic flow is modeled as a seasonal process, and the weighted average of previous seasons is used as the prediction. In ARIMA, time information is provided to the model as exogenous variables. We also tried seasonal ARIMA (SARIMA) with season length equals to a day or a week. The performance of SARIMA is usually the same or slightly better than ARIMA, while it requires much longer training/testing time (often 10×). Thus, we only use ARIMA for large scale experiments.

For post-accident forecasting, as we are forecasting the entire sequence after the accident with two different sets of features, we formulate it as a multi-task sequence-to-sequence learning problem. Each prediction task is to forecast a future time point traffic speed given historical observations as well as traffic accident attributes. The following approaches are used as baselines: 1) Linear regression: analogous to ARIMA model, but with additional accident-specific features; 2) Ridge regression : linear regression with L2 regularization; 3) multi-task Lasso regression: linear regression with L1 regularization.

ARIMA/SARIMA are implemented based on the statsmodels[1] library. Regression models are created using Sklearn[2], and all the neural network architectures are built using Theano [3] and Keras [4]. Grid search is used for parameter tuning, and early stopping [1] is adopted to terminate the training process based on the validation performance. We also experiment with other deep neural network architectures such as feed-

[1] https://github.com/statsmodels/statsmodels
[2] https://github.com/scikit-learn/scikit-learn
[3] https://github.com/Theano/Theano
[4] https://github.com/fchollet/keras

forward neural network baselines and RNN with gated recurrent unit. However, the obtained results so far are not comparable to other baselines. Thus, we choose to omit the results for the convenience of illustration.

4.1.1 Deep LSTM For Deep LSTM, our best architecture consists of 2 hidden LSTM layers of size 64 and 32. We also conduct experiments with 3 or more LSTM layers, but have not seen clear improvement. This may because the input sequence is of low-dimension and larger/deeper recurrent networks tend to overfit [26].

We incorporate *normalized speed, time in a day, day in a week* as input features, and use fully connected layer with linear activation function to aggregate the features transformed by the hidden layers. In addition, a dropout rate of 10% is applied to each LSTM layer. We use mean absolute average percentage error (MAPE) as the loss function. To reduce the negative effects of missing values, which has the default value 0, we define our loss function as the MAPE for non-zero entries, i.e., only calculate MAPE on entries that contain valid sensor readings. For hyper-parameters, the batch size is set to 4, the learning rate is set to $1e^{-3}$, and RMSProp [21] is used as the optimizer. The network is set to train for maximum 150 epochs. In the process of training, we truncate the gradients for back-propagation through time (BPTT) to 36 unrolled steps, and clip the norm of the gradients to avoid gradient explosion. Note that, in the process of testing, the state of the model in each step is propagated to the next step, thus it is possible for the model to capture temporal dependency beyond 36 steps.

4.1.2 Mixture Deep LSTM For Mixture Deep LSTM, the best architecture is a 2-layer LSTM of hidden size 64, with a stacked autoencoder of equal size. The outputs of the stacked autoencoder are copied and concatenated to that of the LSTM network. The final prediction layer is implemented with a fully connected layer. We use linear activation for the LSTM layers and the final prediction layer. We choose the sigmoid activation function for the autoencoder. The loss function used is mean square error (MSE) without zero-valued missing observations.

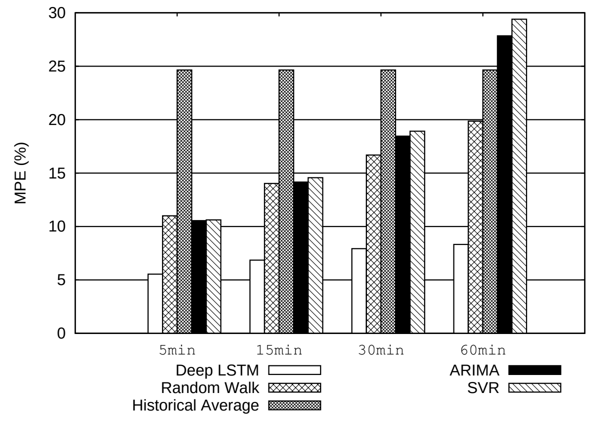
To forecast the post-accident traffic, we extract as inputs the historical sub-sequence one week before the accidents. We feed into the neural network both the raw speed readings and the time stamp features. We also combine 5 static accident features as inputs to stacked autoencoder. The final prediction layer merges the outputs from LSTM and stacked autoencoder. The model generates predicted speed sequence 3 hours after the reported incident. We use SGD with learning rate $5e^{-4}$ with maximum number of epoch as 100. For each LSTM layer, we use a dropout rate of 20%. For the final prediction layer, we impose L_2 regularization on both the weights and the activity.

4.2 Peak-hour Traffic Forecasting We evaluate the performance for our proposed technique by comparing with baselines. Figure 5 shows the performance of deep LSTM and other baselines from 5 minutes to up to 1 hour ahead forecasting. The prediction results are averaged across different sensors. In the experiments, we first generate the prediction on the entire testing sequence and then evaluate the performance on peak-hour (Figure 5(a)) as well as off-peak hours (Figure 5(b)) separately.

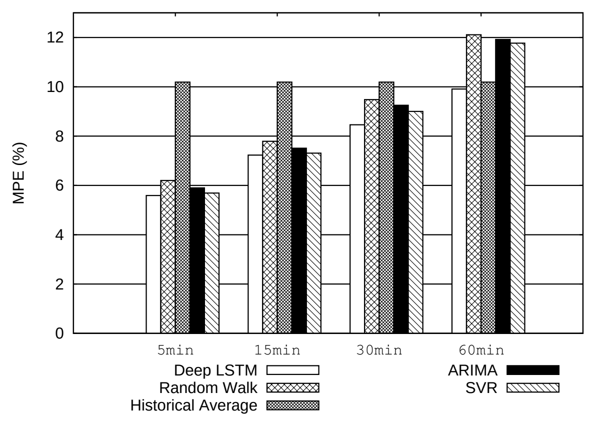
Generally, as the forecasting horizon increases, the performances of most methods degrade. The only exception is historical average based method which conducts prediction by aggregating traffic speeds from previous days and weeks, thus is not sensitive to the change of forecasting horizon. For normal conditions at off-peak hours, almost all the methods have similar performances. The advantage of deep learning approach becomes clear when performing forecasting during peak-hour traffic. Deep LSTM achieves as low as 5% MAPE, which is almost half of the other baselines. Interestingly, we note that the performance of deep LSTM stays relatively stable while baseline methods suffer from significant performance degrade when evaluated at peak-hour.

4.3 Post-Accident Traffic Forecasting We evaluate the empirical performance of Mixture Deep LSTM for the task of post-accident forecasting. Figure 6 displays the forecasting MAPE for different methods with forecasting horizons vary from 5 minutes to up to 3 hours. Mixture Deep LSTM is roughly 30 % better than baseline methods. Random walk cannot response to the dynamics in the accident, thus suffers from poor performance especially during the first 5 minutes after the incident. The regularization in the multi-task ridge regression and the multi-task lasso regression avoids the over-fitting of model, and is more adaptable to the post-accident situation. Mixture Deep LSTM, which jointly models the normal condition traffic as well as the traffic accident scenario, achieves the best result.

Table 1 shows the forecasting performance evaluated across the complete 3-hour span after the accident. Mixture deep LSTM performs significantly better than other baselines. To justify the use of recurrent neural network in our architecture as opposed to other neural network design, we also evaluate the performance of a 3-layer feed-forward network. The poor performance of the feed-forward network shows the importance of tak-



(a) Peak hour



(b) Off-Peak hour

Figure 5: Off-peak and peak-hour traffic forecasting MAPE of Deep LSTM over baselines with respect to different forecasting horizons

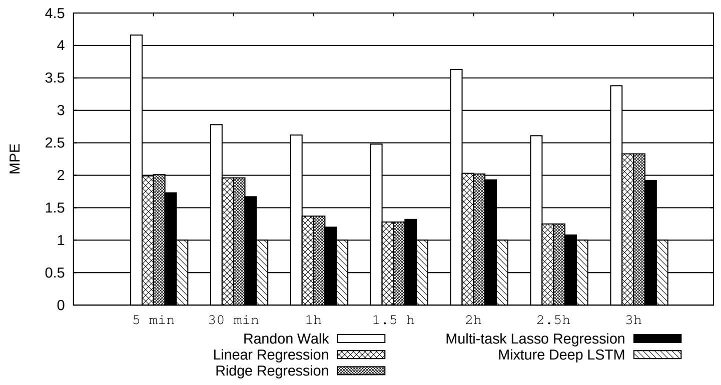


Figure 6: Post-accident traffic forecasting MAPE of Mixture Deep LSTM over baselines with respect to different forecasting horizons

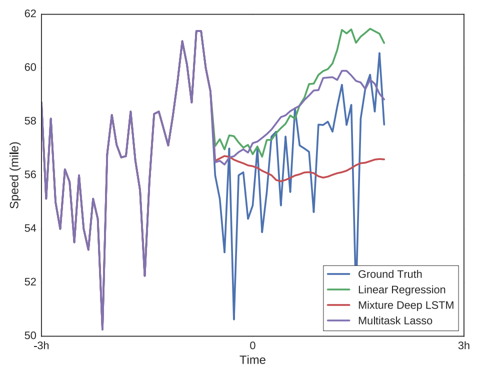


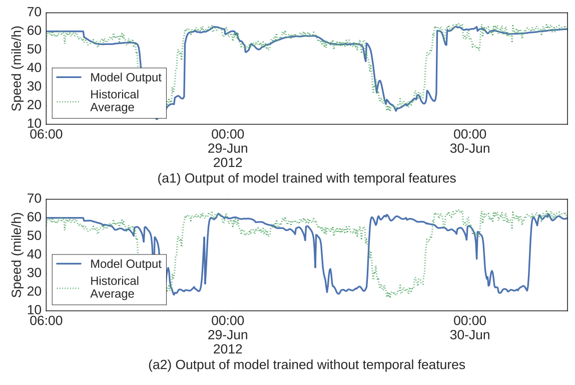
Figure 7: Three-hour post-accident predicted sequence comparison of different methods with respect to ground truth

Table 1: 3 hours post-accident forecasting MAPE comparison of Mixture Deep LSTM over baseline models

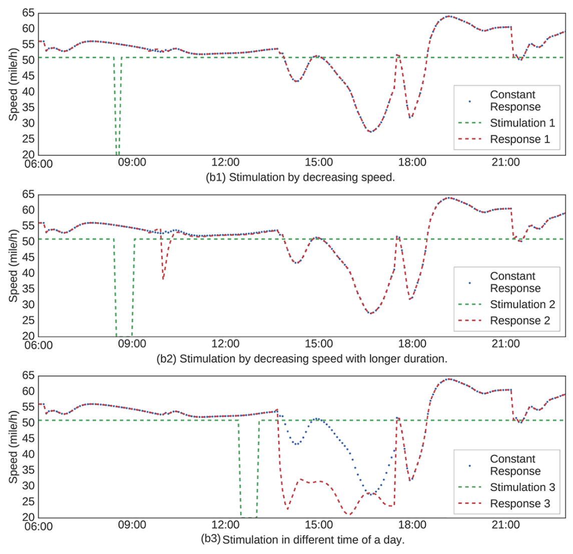
Method	MAPE
Mixture Deep LSTM	0.9700
Deep LSTM	1.003
Randon Walk	2.7664
Linear Regression	1.6311
Ridge Regression	1.6296
Multi-task Lasso Regression	1.4451
Feed-forward Network	3.6432

ing into account the temporal dependencies for the history sequence. Figure 7 shows the predicted sequence from the time when the accident occurs (time 0) to up to 3 hours. The predictions are concatenated with the input sequences to show the temporal dependencies. For the case shown in Figure 7, the sharp drop caused by the accident results in a different pattern from that in the same time in last week. The plot shows that Mixture deep LSTM can correctly predict the mean value of a highly non-stationary long time sequence.

4.4 Model Inspection To further understand the behavior of proposed deep architecture, we conduct a series of qualitative studies. Our method of model inspection is inspired by the electric stimulation study in brain science. If we make the analogy of the trained neural network as a human brain, the weights in the network contain "learned" knowledge. By feeding into different type of stimulation, the response of the neural network would reflect this learned knowledge. The stimulation is conducted by first feeding the trained neural network a constant speed input. The neural network transforms the input signal into 1 hour ahead predictions. Then we take the output sequence as the input signal again and repeat the procedure until 48 hours of outputs are generated. Despite the fact that constant input contains almost zero information about the variations in the time series, we receive interesting responses from the model.



(a) Model Inspection



(b) Drop Response

Figure 8: Learned pattern from Mixture Deep LSTM network. Trained model response under different stimulation signals, compared with groud truth sequence

In peak-hour forecasting, we observe the importance of incorporating temporal feature, i.e., time of the day and day of the week, to the performance. To justify this, we stimulate neural different neural networks: one trained with temporal features and the other without. Figure 8(a) illustrates the model response when stimulated with a constant speed input. As shown in Figure8(a1), with temporal features, the model (blue solid line) tends to generate similar trend as the historical average (green dotted line), which shows that the mode can learn long-term as well as short-term seasonal information from historical data. While the model trained without temporal features (in Figure8(a2)) can also generate similar outputs, but the period is considerably shorter than that of the historical average.

Next, we simulate the post-accident scenario. We inject many different interruptions in our input stimulate signal and investigate model response. In Figure 8(b1), the blue dotted line represents the model's response to a constant speed stimulation. Then we generate stimulation with a 5 minutes sudden drop (green dotted line) to simulate the fluctuation. The model response is shown in the red dashed line. We note that the model response is exactly the same as the constant speed response. This shows that the model is robust to ephemeral traffic change, which is simulated by a small fluctuation in input signal.

However, when the duration of the fluctuation becomes larger, the response starts to appear. Figure 8(b2) shows the response after we increases the duration of the sudden drop to half one hour. After the sudden drop in the input signal, the response quickly decreases and then gradually goes back to normal. This reflects the behavior of the traffic flow when facing an accident. Note that the simulated accident in Figure 8(b2) happens at off-peak hour. The model response is further amplified when the accident happens at the time of peak-hour. As shown in Figure 8(b3), we change the time of the stimulation from morning to afternoon, and a more dramatic change is observed. This is because traffic changes in different time of the day will have different affects, e.g., the speed decrease during afternoon usually indicates the arriving of peak hours.

5 Conclusion

In this paper, we studied the problem of traffic forecasting under extreme conditions, in particular for peak hour and post-accident scenarios. We proposed a generic deep learning framework based on long short term memory unit. In particular, we used Deep LSTM for peak-hour traffic forecasting with careful feature engineering. We further proposed a Mixture Deep LSTM architecture that unifies Deep LSTM and stacked autoencoder. When evaluated on a large-scale real-world traffic data, our framework is able to achieve significant performance improvement over baselines for both conditions. We also employed a novel model inspection method by signal stimulation. The model responses from those stimulations provided insights into the trained neural network on traffic data.

Acknowledgment

This research has been supported in part by METRANS Transportation Center under Caltrans contract# 65A0533, the USC Integrated Media Systems Center (IMSC), and unrestricted cash gifts from Oracle. Rose Yu and Yan Liu were additionally supported by NSF research grants IIS-1254206 and IIS-1539608. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of any of the sponsors such as NSF.

References

[1] Y. Bengio, *Practical recommendations for gradient-based training of deep architectures*, in Neural Networks: Tricks of the Trade, Springer, 2012, pp. 437–478.

[2] Z. Che, D. Kale, W. Li, M. T. Bahadori, and Y. Liu, *Deep computational phenotyping*, in SIGKDD, ACM, 2015, pp. 507–516.

[3] Q. Chen, X. Song, H. Yamada, and R. Shibasaki, *Learning deep representation from big and heterogeneous data for traffic accident inference*, in AAAI, 2016.

[4] S. D. Clark, M. S. Dougherty, and H. R. Kirby, *The use of neural networks and time series models for short term traffic forecasting: a comparative study*, in Traffic Engineering and Control, no. 34, 1993, pp. 311–318.

[5] A. Downs, *Still stuck in traffic: coping with peak-hour traffic congestion*, Brookings Institution Press, 2005.

[6] G. Duranton and M. A. Turner, *The fundamental law of road congestion: Evidence from us cities*, The American Economic Review, (2011), pp. 2616–2652.

[7] A. Garib, A. Radwan, and H. Al-Deek, *Estimating magnitude and duration of incident delays*, Journal of Transportation Engineering, 123 (1997), pp. 459–466.

[8] M. Hermans and B. Schrauwen, *Training and analysing deep recurrent neural networks*, in NIPS, 2013, pp. 190–198.

[9] S. Hochreiter and J. Schmidhuber, *Long short-term memory*, Neural computation, 9 (1997), pp. 1735–1780.

[10] W. Huang, G. Song, H. Hong, and K. Xie, *Deep architecture for traffic flow prediction: deep belief networks with multitask learning*, ITS, IEEE Transactions on, 15 (2014), pp. 2191–2201.

[11] H. R. Kirby, S. M. Watson, and M. S. Dougherty, *Should we use neural networks or statistical models for short-term motorway traffic forecasting?*, International Journal of Forecasting, 13 (1997), pp. 43–50.

[12] Q. V. Le, W. Y. Zou, S. Y. Yeung, and A. Y. Ng, *Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis*, in CVPR, IEEE, 2011, pp. 3361–3368.

[13] M. Lippi, M. Bertini, and P. Frasconi, *Short-term traffic flow forecasting: An experimental comparison of time-series analysis and supervised learning*, ITS, IEEE Transactions on, 14 (2013), pp. 871–882.

[14] W. Liu, Y. Zheng, S. Chawla, J. Yuan, and X. Xing, *Discovering spatio-temporal causal interactions in traffic data streams*, in SIGKDD, ACM, 2011, pp. 1010–1018.

[15] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang, *Traffic flow prediction with big data: A deep learning approach*, ITS, IEEE Transactions on, 16 (2015), pp. 865–873.

[16] 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*, Transportation Research Part C: Emerging Technologies, 54 (2015), pp. 187–197.

[17] B. Pan, U. Demiryurek, C. Shahabi, and C. Gupta, *Forecasting spatiotemporal impact of traffic incidents on road networks*, in ICDM, IEEE, 2013, pp. 587–596.

[18] K. A. Small and E. T. Verhoef, *The economics of urban transportation*, Routledge, 2007.

[19] B. L. Smith and M. J. Demetsky, *Traffic flow forecasting: comparison of modeling approaches*, Journal of transportation engineering, 123 (1997), pp. 261–266.

[20] I. Sutskever, O. Vinyals, and Q. V. Le, *Sequence to sequence learning with neural networks*, in NIPS, 2014, pp. 3104–3112.

[21] T. Tieleman and G. Hinton, *Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude*, COURSERA: Neural Networks for Machine Learning, 4 (2012), p. 2.

[22] J. Van Lint, S. Hoogendoorn, and H. Van Zuylen, *Freeway travel time prediction with state-space neural networks: modeling state-space dynamics with recurrent neural networks*, Journal of the Transportation Research Board, (2002), pp. 30–39.

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

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

[25] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio, *Show, attend and tell: Neural image caption generation with visual attention*, in ICML, 2015, pp. 2048–2057.

[26] W. Zaremba, I. Sutskever, and O. Vinyals, *Recurrent neural network regularization*, arXiv preprint arXiv:1409.2329, (2014).