Characterizing Driving Styles with Deep Learning

Weishan Dong¹, Jian Li¹,², Renjie Yao¹, Changsheng Li¹, Ting Yuan¹, Lanjun Wang¹,³
¹IBM Research - China
²Nanjing University, China
³University of Waterloo, Canada
{dongweis, rjyaobj, lcsheng, ytyuanyt}@cn.ibm.com, lioniellijian@hotmail.com, lanjun.wang@uwaterloo.ca

Abstract—Characterizing driving styles of human drivers using vehicle sensor data, e.g., GPS, is an interesting research problem and an important real-world requirement from automotive industries. A good representation of driving features can be highly valuable for autonomous driving, auto insurance, and many other application scenarios. However, traditional methods mainly rely on handcrafted features, which limit machine learning algorithms to achieve a better performance. In this paper, we propose a novel deep learning solution to this problem, which could be the first attempt of studying deep learning for driving behavior analysis. The proposed approach can effectively extract high level and interpretable features describing complex driving patterns from GPS data. It also requires significantly less human experience and work. The power of the learned driving style representations are validated through the driver identification problem using a large real dataset.

I. INTRODUCTION

Deep neural networks have been intensively studied in recent years, and many record-breaking progresses have been made on solving computer vision, speech recognition, and natural language processing problems [1]–[3]. However, so far few attempts have been made on applying deep learning to trajectory data analysis, which is a key research topic in spatiotemporal data analytics, urban computing, intelligent transportation, and Internet-of-Things (IoT) areas. In this work, we study an important real-world problem of trajectory analysis and propose a deep learning based solution. The problem comes from the automotive industry, especially the auto insurance and the telematics domain, that is, to characterize the driving styles of car drivers from vehicle sensor data, e.g., GPS (Global Positioning System). Because of individual differences, each driver has a signature driving style which is a complex combination of fine-grained driving behaviors and habits. Ideally, it should cover the way of accelerating, braking, turning, etc., and their (temporal) combinations given specific driving contexts such as road levels, road shapes, traffic conditions, and even weather.

A good driving style representation is useful in many ways. For instance, it can be particularly useful in autonomous driving that has become a hot topic in both industries and academia. A better understanding of how human drive a car is undoubtedly helpful to teach machines drive like a human. Other examples include to assess drivers' driving risks when correlated with external labels such as claims, accidents, and traffic violations [4]. In auto insurance businesses (e.g., pay-as-you-drive and pay-how-you-drive), this is a key pricing reference. Another common and interesting application is driver identification, i.e., to identify the true driver of anonymized trips [1] which is useful in scenarios including claim fraud detection, estimating how many drivers share a car for insurance pricing, and the design of intelligent driver assistance systems. When the number of candidate drivers is large (e.g., 1000), it becomes a much harder classification problem than just differentiate safe and unsafe driving behaviors. If a good driving style representation can solve the driver identification problem, we have reasons to believe that other problems based on driving behavior characterizations can also be solved well. Therefore, in this paper we take the driver identification problem as a sample task to evaluate the effectiveness of driving style characterization.

The state-of-the-art methods of modeling driving styles are mainly based on handcrafted features [1]. However, manually defining the driving style by traditional feature engineering is challenging: (1) It heavily relies on domain knowledge and human experience. (2) The discriminative power of the features is often unknown before feeding into machine learning algorithms; so a common practice is to enumerate as many features as possible and then apply feature selection, which requires considerable efforts. (3) The best descriptors of driving patterns may change given different data and contexts (e.g., drivers in China may have different driving patterns from those in US), thus a generic model is hard to obtain. (4) The feature designs are usually separated from the learning algorithms, which cannot guarantee a best synergia between features and algorithms. (5) Driving behaviors are typically a sequence of operations, therefore possible combinations of features to define such sequences can be huge. It is hardly to find an optimal driving style representations just by enumerations.

On the other hand, recent advances of deep learning reveal that deep neural network is a promising method of extracting features from sensor signal data (e.g., speech). Inspired by this, in this paper, we propose a novel deep learning approach for characterizing driving styles, which consists of: (1) a special design of transforming raw sensor data (such as GPS) into feature matrices, and (2) employing deep neural networks such as convolutional neural network (CNN) and recurrent neural network (RNN) to learn driving style features from the feature matrices. In such a way, high level discriminative features characterizing how one drives a car can be effectively obtained. Compared with existing methods, it results in significantly less human work and better accuracy. To the best of our knowledge, this is the first work of designing a deep learning framework for driving style representation learning. Experiments on large real data will show that, in terms of identifying the true driver of a trip, the proposed deep learning approach dramatically

¹https://www.kaggle.com/c/axa-driver-telematics-analysis
outperforms the state-of-the-art methods by a large margin. If considering additional trip level features (such as trip length, trip shape, etc.) other than just the driving behavior features, performance of traditional machine learning methods can be improved, but is still worse than the proposed deep learning approach that only considers driving style features. This indicates that deep learning can be a powerful tool for characterizing driving styles.

II. PROPOSED APPROACH

The proposed approach consists of two parts: data transformation and feature learning by deep networks. For simplicity, we consider raw GPS data as the only input. Nonetheless, as can be seen later, our approach can be easily generalized to work with other types of sensor data and rich driving contexts.

A. Data Transformation: from Geospatial Domain to Driving-Feature Domain

Deep neural networks have been proved powerful in learning from speech data [2], [5], [6]. GPS sensor data are also kinds of time series having similar characteristics with speech signals, which is the primary motivation for us to develop deep learning methods for driving feature representation learning. However, a huge gap is that GPS data in its raw format – a sequence of point geo-locations defined by 2-D coordinates \((x, y)\), each having a timestamp \(t\) – encode spatiotemporal information in an implicit way. Our empirical studies showed that simply treating raw GPS data as three-dimensional signal inputs for either traditional machine learning or deep learning algorithms just does not work. A way of transforming GPS data sequences (or say, trajectories) into an easier consumable format for deep learning needs to be developed.

We define the GPS trajectory as a sequence of tuples \((x, y, t)\), whose length can be varying. Inspired by the idea of N-Gram probabilistic model proposed by [7], where each word depends only on the last \(n - 1\) words, we utilize the context window concept to make our vision focused on fixed length’s trajectory in periods of time. During each window, we may discover potential patterns through observing the behaviors of the driver over different situations. For examples, some drivers may go through a sharp corner quickly while the others would like to slow down, and heavy accelerations can often happen to a group of drivers while the others may never have such aggressive driving behaviors. In fact, the behaviors are interdependent in time where the length of this period can be \(L_s\). We can roughly say that a current driving behavior depends on what happened in the last \(L_s - 1\) time points. Hence, we consider \(L\) as the window length, and then calculate the basic features from the raw GPS sequence. Then, for each segment, we assign \(L_f\) neighboring points into a frame and generate the statistical feature matrix.

![Fig. 1. Data transformation. First, we generate segments of length \(L_s\) and calculate the basic features from the raw GPS sequence. Then, for each segment, we assign \(L_f\) neighboring points into a frame and generate the statistical feature matrix.](image)

Trajectories. We propose to use the following five features to replace the frequency axis of speech data’s feature map: speed norm, difference of speed norm, acceleration norm, difference of acceleration norm, and angular speed. We call them as basic features derived from GPS data at every time point. As a result, each segment has a basic feature matrix sized \(5 \cdot L_s\). Notably, these are all point-wise instantaneous features.

To reduce the possible impacts of outliers (which may be generated by small GPS errors) in such point-wise features, we further derive the statistical information by ‘framing’ the segments. In each segment, we put every \(L_f\) \((L_f < L_s)\) neighboring points into a frame with a shift of \(L_f/2\), and then calculate the mean, minimum, maximum, 25%, 50%, and 75% quartiles, and standard deviation of the basic features in each frame, totally seven statistics. Such frame level statistical features can be regarded as a more stable representation of the basic features in every time period of length \(L_f\). The resulting statistical feature matrix \((5 \times 7 = 35\) rows representing the driving feature axis and \(2 \cdot L_s/L_f\) columns representing the time axis) will serve as input to deep neural networks.

In summary, we use a ‘large window’ to segment a GPS sequence into fixed-length ‘patches’ to model the interdependency among the instantaneous features. Meanwhile, we employ a ‘small window’ to further enframe each segment so as to describe the driving features from the statistical perspectives in a short time period. Such a double-windowed feature matrix design not only encodes the instantaneous driving behaviors, but also conveys how the patterns change over time. Importantly, only low level driving behaviors are calculated here and no explicit temporal combination is modeled yet. We expect deep neural networks to learn and extract higher levels of driving style features from such an input format.

Figure 7 illustrates the proposed data transformation for GPS data from the original geospatial domain to the driving-feature domain. A long GPS trip thus will be transformed into a number of statistical feature matrices, each corresponds to a segment. The label of the original trip (e.g., driver ID) will be assigned to all the segments for supervised learning. An
example of the data transformation is shown in Figure 2. In the example, GPS data sampling rate is 1Hz, and the location coordinates (in meters) are anonymized with trip origin set to (0, 0). The generated statistical feature matrix sized 35 × 128 corresponds to a trip segment of $L_s = 256s$ and $L_f = 4s$.

In the feature matrix, the order of rows follows our previous introduction: the speed norm related statistics are in the first seven rows, then follow the difference of speed norm related statistics, and so on. For each basic feature, the order of the seven statistics are also in the same order as in which we listed the statistical features. We can see that the feature matrix in Figure 2 clearly shows the sharp turn at the beginning of the trip (see the last seven rows 29–35, indicating large values of angular speed statistics) and the speed is getting higher over time (see the first seven rows 1–7).

### B. Learning with Convolutional Neural Network Using 1-D Convolution And Pooling

Convolutional neural network (CNN) [1, 8] has become popular for image recognitions. It consists of alternating convolution and pooling layers. Its two main characteristics, locality and weight sharing, are also beneficial to learning features from time series data such as audio and speech [2, 9]. We will first employ CNN for learning driving styles from the transformed feature matrix defined in Section II-A.

Given the statistical feature matrix data as inputs, we propose to apply 1-D convolutions only over the time axis (columns) because the convolution over driving features has no practical significance. This is because unlike the frequency axis in speech feature maps where there is local structure from low frequencies to high frequencies in a continuous domain, driving features do not have ordered information and are discrete. Exchanging the orders of features (i.e., rows) in the statistical feature matrix yields exactly the same input semantically. In other words, there is no meaningful local structure along the feature axis, thus convolution is not meaningful either. Similar ideas have been proposed for audio classification [9], where the time-axis convolution helps to learn more effective acoustic features such as phoneme and gender. In our problem, it maps to learning complex driving behaviors. Intuitively, the lower convolution layer in the CNN is able to detect fine-grained driving behaviors such as aggressive accelerations, while the higher layer has the ability to find more abstract and semantic level driving patterns. The pooling layer is another significant component of CNN. Similar to the convolution layers, we propose to apply 1-D Max-Pooling only over the time axis. The feature values computed at different time points are pooled and represented by the maximum. It helps realize translation invariant for driving patterns over time.

An illustration of the 1-D convolution and pooling are shown in Figure 3. The CNN architecture we build for the driver identification problem is as follows. The net has six layers. The first three are convolution-pooling layers and the remaining three are fully-connected. Specifically, assuming the number of frames in each segment is 128, then the first layer filters the $35 \times 128$ input data with 32 kernels of $35 \times 5$ with a stride of 1 frame. The second convolutional layer takes as input the pooled output of the first convolutional layer where the pool size is $1 \times 2$ and filters it with 64 kernels of $35 \times 5$ with a stride of 1 frame. The third convolutional layer also has 64 kernels of size $1 \times 3$ connected to the pooled outputs of the second convolutional layer. Then the fourth and fifth layer are fully connected and have 128 neurons each. Sigmoid activations are applied to the output of every layers. The last layer is a Softmax.

### C. Learning with Recurrent Neural Network

Recurrent neural network (RNN) is another popular deep neural network that has many variations such as Elman’s RNN [10], LSTM [11] and Bi-directional RNN [12]. It is a kind of feedforward neural network augmented by the inclusion of edges that span adjacent time steps, introducing a notion of time to the model [13]. Given an input sequence $s = (s_1, … , s_T)$, each neuron in the recurrent hidden layer receives input from the current data point $s_t$ and also from hidden node values $h_{t-1}$ in the previous time step:

$$ h_t = g(W_{hs}s_t + W_{hh}h_{t-1} + b_h) $$

where $W_{hs}$ is the input-hidden weight matrix, $W_{hh}$ is the matrix of weights between the hidden layer and itself at adjacent time steps, $b_h$ denote bias vectors, and $g$ is the hidden layer function. RNN can be interpreted as an unfolded network across time steps which is inherently deep in time. Since it is very successful on sequence learning tasks such as speech recognition and machine translation, it is natural
to apply RNN to GPS sequences for driving style feature learning. In our case, we regard the transformed statistical feature matrix as a sequence of 35-D frames and feed it into RNN. As analyzed in [14], training RNN is difficult due to vanishing and exploding gradients. Researchers have been working on developing optimization techniques and network architectures to solve this problem. Instead of using more sophisticated methods, [15] proposed an RNN architecture composed of rectified linear units (ReLU) and used the identity matrix or its scaled version to initialize the recurrent weight matrix. Their simple solution is even comparable to a standard implementation of LSTM on certain tasks such as language modeling and speech recognition. For our driver identification task, we tend to use such a simple yet powerful network, which we denote as IRNN. The last layer of IRNN, again, is a Softmax appending to the recurrent layer. We can also construct an IRNN with two stacked recurrent layers, denoted by StackedIRNN. The output of the 1st recurrent layer is also a sequence, which connects to the 2nd recurrent layer as input. And then a Softmax is appended as the last layer. In the next section, we will see that this allows higher level driving feature extraction and leads to better performance.

III. EXPERIMENTS

A major requirement on the data quality for characterizing fine-grained driving behaviors is that the GPS data sampling rate must not be too low. In addition, a regular sampling interval is preferred. It can be imagined that a low sampling rate can result in too much information loss, especially the instantaneous car movement cannot be estimated accurately. Our empirical studies revealed that generally when the sampling rate is lower than 0.1Hz (one record per ten seconds), the performance of any approaches can become poor. We adopt a large public dataset from the Kaggle 2015 competition on Driver Telematics Analysis for experimental studies. The dataset contains 547,200 anonymized real trips of 2,736 drivers. Each driver has 200 driving trips with varying lengths. The dataset contains 547,200 anonymized real trips of 2,736 drivers. Each driver has 200 driving trips with varying lengths. The dataset contains 547,200 anonymized real trips of 2,736 drivers. Each driver has 200 driving trips with varying lengths.

But different from CNN and RNN, GBDT does not explicitly model the locality or the time steps in sequence. Therefore, given the same 35 × 125 matrix as input, GBDT treats it as an unfolded vector of 35 × 128 = 4,480 features.

As another baseline representing traditional feature engineering methodology for characterizing driving styles, we also train GBDT on a set of 57 manually defined driving behavior features. We denote it as TripGBDT. These features were used in a participation to the same Kaggle competition on Driver Telematics Analysis and achieved 0.92 AUC score in detecting the false trips, indicating the effectiveness of these features. The feature set consists of global features and local ones. The global features are trip level statistics, including the mean, min, max, std, and quantiles (25%, 50% and 75%) of speed norm, difference of speed norm, acceleration norm, difference of acceleration norm, and angular speed of a whole trip. In addition, the following ones are also defined as global features: time duration of the whole trip, trip length, average speed (trip length divided by time duration), area of the minimal rectangle containing the trip shape, and lengths of the two edges of the minimal rectangle. The local features are defined as follows. We first extract the moving angles (0 to 180 degrees) for each point (based on a window of three consecutive points), and divide them into eight bins [0,10], [10,20], [20,30], [30,45], [45,60], [60,90], [90,120], and [120,180]. In each bin, we also calculate the mean, min, max, std, and quantiles (25%, 50% and 75%) of speed norm, difference of speed norm, acceleration norm, difference of acceleration norm, and angular speed. These features model the correlations between driving behaviors and the road’s local shape. We first downsample the trip with sampling rates 1, 2, 3, 4, and 5 records per second, and then extract the features on the downsampled trips. This can be seen as applying a smoothing procedure. In total, there are 57 features defined. We can see that not only driving related features are included, but also trip geometry and global statistics are available. In contrast, in the proposed deep learning approach, only segment level statistics about short-time driving behaviors are calculated. The global information of a trip is completely invisible to the neural networks. If predictions for a weighted vote. As the driver identification is a classification problem, we will report segment-level accuracy, trip accuracy, and trip top-5 accuracy in experiments.

A. Candidate Methods For Comparison

Using the same input data, we train five deep neural networks for comparisons:

- **CNN**: see Section II-B
- **NoPoolCNN**: CNN without pooling layers
- **IRNN**: see Section II-C with 100 neurons in the recurrent layer
- **PretrainIRNN**: Use the features extracted at the third convolutional layer in the pre-trained CNN as inputs to train an IRNN
- **StackedIRNN**: see Section II-C with 100 neurons in each recurrent layer

As a baseline of non-deep learning methods, Gradient Boosting Decision Tree (GBDT) [16] has been recognized as one of the most powerful machine learning algorithms. We also include it in comparison with using the same data as inputs. The original problem in the competition was to detect trips that are not real trips. Therefore, we regard the driver labels as true labels, indicating the effectiveness of these features. The feature set consists of global features and local ones. The global features are trip level statistics, including the mean, min, max, std, and quantiles (25%, 50% and 75%) of speed norm, difference of speed norm, acceleration norm, difference of acceleration norm, and angular speed of a whole trip. In addition, the following ones are also defined as global features: time duration of the whole trip, trip length, average speed (trip length divided by time duration), area of the minimal rectangle containing the trip shape, and lengths of the two edges of the minimal rectangle. The local features are defined as follows. We first extract the moving angles (0 to 180 degrees) for each point (based on a window of three consecutive points), and divide them into eight bins [0,10], [10,20], [20,30], [30,45], [45,60], [60,90], [90,120], and [120,180]. In each bin, we also calculate the mean, min, max, std, and quantiles (25%, 50% and 75%) of speed norm, difference of speed norm, acceleration norm, difference of acceleration norm, and angular speed. These features model the correlations between driving behaviors and the road’s local shape. We first downsample the trip with sampling rates 1, 2, 3, 4, and 5 records per second, and then extract the features on the downsampled trips. This can be seen as applying a smoothing procedure. In total, there are 57 features defined. We can see that not only driving related features are included, but also trip geometry and global statistics are available. In contrast, in the proposed deep learning approach, only segment level statistics about short-time driving behaviors are calculated. The global information of a trip is completely invisible to the neural networks. If

2https://www.kaggle.com/c/axa-driver-telematics-analysis/data

3The original problem in the competition was to detect trips that are not real trips. Therefore, we regard the driver labels as true labels, and regard the false trips as noise, which does not affect the evaluation much.
in such a case, deep learning methods can still outperform TripGBDT, it is convincing to conclude that deep neural networks are more powerful in characterizing driving styles.

B. Experiment on 50 Drivers’ Data

The training dataset constructed from 50 drivers’ trips includes over 35,000 segments, which are greatly augmented compared with the original 8,000 trips. The parameters of algorithms are tuned using the standard 5-fold cross-validation. We use batch size 128 for training the neural nets. For CNNs, we use the stochastic gradient descent optimizer with learning rate 0.05, decay 1e-6, and Nesterov momentum 0.9. For RNNs, we use the RMSProp optimizer with learning rate 1e-6, \( \rho = 0.9 \), and \( \epsilon = 1e-6 \). For GBDT and TripGBDT, the max tree depth 6 is used and the stopping iterations leading to the best performance are chosen. The best results obtained by each algorithm are summarized in Table I. We can find that IRNN demonstrates strong advantages over the others. And the obtained accuracies are quite acceptable considering that random guess on trip accuracy should be 2% (1/50). Not surprisingly, GBDT performs the worst among all. The best result (bolded in table) is from StackedIRNN. Although with a simpler architecture of just one recurrent layer, IRNN easily beats the remaining candidates. NoPoolCNN performs worse than CNN, indicating the effectiveness of pooling layers. PretrainIRNN performs better than CNN, however, it still does not outperform IRNN or StackedIRNN that directly run on the feature matrices. But it worth to mention that IRNN’s training time is much longer than CNN and GBDT. The more complex StackedIRNN only exhibits small improvement over the single layer IRNN, while the training time cost is nearly doubled. For TripGBDT, only trip level accuracies are available because there is no segment level training or testing. Although the trip level global information is provided in addition to the driving related features, TripGBDT cannot perform as good as IRNN and StackedIRNN but better than CNN on this small scale test. In general, deep neural networks are capable of learning good driving style representations from the transformed feature matrix and can perform better than traditional methods.

| Method       | Seg (%) | Trip (%) | Trip Top-5 (%) |
|--------------|---------|----------|---------------|
| NoPoolCNN    | 16.9    | 28.3     | 56.7          |
| CNN          | 21.6    | 34.9     | 63.7          |
| PretrainIRNN | 28.2    | 44.6     | 70.4          |
| IRNN         | 34.7    | 49.7     | 76.9          |
| StackedIRNN  | 34.8    | 52.3     | 77.4          |
| GBDT         | 18.3    | 29.1     | 55.9          |
| TripGBDT     | -       | 51.2     | 74.3          |

C. Experiment on 1000 Drivers’ Data

We further conduct a large scale test using 1000 drivers’ trip data. We include CNN, StackedIRNN, and TripGBDT in comparison since they are representative methods in Table I. The neural nets’ parameters keep unchanged. Parameters of TripGBDT are further tuned with max tree depth 20. Results are reported in Table II. We can see that deep neural networks exhibit significantly better scalability: the performance does not decrease much considering the problem becomes a harder 1000-class problem (random guess accuracy 0.1%). Contrarily, TripGBDT’s performance becomes dramatically worse. This indicates that if the problem becomes harder, manually defined features are no longer powerful as the ones extracted by deep neural networks. Still, StackedIRNN performs the best. But again, it cost much more computational time to converge.

| Method        | Seg (%) | Trip (%) | Trip Top-5 (%) |
|---------------|---------|----------|---------------|
| CNN           | 15.4    | 26.7     | 46.7          |
| StackedIRNN   | 27.5    | 40.5     | 60.4          |
| TripGBDT      | -       | 7.2      | 15.8          |

D. Interpretation of Learned Features

It is interesting to investigate what kind of features are learned by the deep networks. Due to space limit, here we only report if using the 2nd recurrent layer of the StackedIRNN trained from the 1000 drivers’ data, among all the training samples, which ones result in the maximum activations on the 100 hidden neurons. For each neuron, we visualize the training data that lead to the top five activations. By observing the common patterns among the data, we can analyze what features have been learned. In Figure II, we show the results of three selected neurons. Interestingly, some neuron seems to have learned driving behaviors such as slowdown at hard turns, high speed driving along straight roads, and even those GPS failures that caused sudden huge jumps, i.e., outliers. The extracted features are fairly interpretable, which partially explains why deep learning performs well.

IV. DISCUSSIONS

The proposed deep learning approach also has another advantage over the methods such as TripGBDT that rely on trip level features. Because only segment level data is needed, the deep learning approach can be used in real-time prediction scenarios where segment data can be available online as the car moves. In contrast, trip level features such as trip length and trip time duration can only be available after the trip ends. This makes TripGBDT cannot be used for online prediction, whereas deep learning approaches are far more flexible.

Privacy is often a key concern in analyzing telematics data. The proposed data transformation in Section II-A has a merit of not revealing any location or time specific information to the learning phase, even if the GPS trip is not anonymized. This is because the basic features only describe movements with relative location and time information. In a real system, if the data transformation can be done, e.g., on the vehicle side, data privacy can be well preserved even if the learning is performed in a centralized manner such as on the cloud side, which requires data to be uploaded for analyses.

Driving contexts, e.g., road level, road shape, traffic, and weather, can also influence driving behaviors. Additional car sensor data, such as OBD (On-Board Diagnostic) monitoring the vehicle status, are also helpful to model driving behaviors. Such contextual and sensor data inputs can be further plugged into our framework to enrich the statistical feature matrix. As long as the data are in the format of sequences or time series, similar transformation can be designed so that the calibrated inputs to deep neural networks can encode richer information. Deep neural networks should be able to discover the correlations and learn even better driving style representations.
In this paper, we proposed a deep learning approach for characterizing driving styles, which could be the first attempt of studying deep learning on GPS data analyses and especially driving feature extraction. First, we developed a data transformation method to construct an easily consumable input format (the statistical feature matrix) from raw GPS sequences for deep learning. Second, we proposed several CNN and RNN architectures and studied their power on learning a good representation of driving styles from the transformed data inputs. Taking driver identification as a sample task, the experiments on a large real dataset showed that the proposed deep learning approach significantly outperforms traditional machine learning methods as well as the state-of-the-art feature engineering methods that mostly rely on handcrafted driving behavior features. More importantly, the driving style features learned by the deep neural networks were fairly interpretable, explaining the effectiveness of deep learning. In a word, deep learning can be a powerful tool for learning driving style features and many other driving behavior analysis problems.

V. CONCLUSION

In this paper, we proposed a deep learning approach for characterizing driving styles, which could be the first attempt of studying deep learning on GPS data analyses and especially driving feature extraction. First, we developed a data transformation method to construct an easily consumable input format (the statistical feature matrix) from raw GPS sequences for deep learning. Second, we proposed several CNN and RNN architectures and studied their power on learning a good representation of driving styles from the transformed data inputs. Taking driver identification as a sample task, the experiments on a large real dataset showed that the proposed deep learning approach significantly outperforms traditional machine learning methods as well as the state-of-the-art feature engineering methods that mostly rely on handcrafted driving behavior features. More importantly, the driving style features learned by the deep neural networks were fairly interpretable, explaining the effectiveness of deep learning. In a word, deep learning can be a powerful tool for learning driving style features and many other driving behavior analysis problems.

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in Advances in neural information processing systems, 2012, pp. 1097–1105.

[2] O. Abdel-Hamid, A.-r. Mohamed, H. Jiang, L. Deng, G. Penn, and D. Yu, “Convolutional neural networks for speech recognition,” Audio, Speech, and Language Processing, IEEE/ACM Transactions on, vol. 22, no. 10, pp. 1533–1545, 2014.

[3] R. Collobert and J. Weston, “A unified architecture for natural language processing: Deep neural networks with multitask learning,” in Proceedings of the 25th international conference on Machine learning. ACM, 2008, pp. 160–167.

[4] A. Laurie, “Telematics: the new auto insurance,” Towers Watson, 2011.

[5] A. Graves, A.-r. Mohamed, and G. Hinton, “Speech recognition with deep recurrent neural networks,” in Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on. IEEE, 2013, pp. 6645–6649.

[6] A. Hannun, C. Case, J. Casper, B. Catanzaro, G. Diamos, E. Elsen, R. Prenger, S. Satheesh, S. Sengupta, A. Coates et al., “Deep-speech: Scaling up end-to-end speech recognition,” arXiv preprint arXiv:1412.5567, 2014.

[7] P. F. Brown, P. V. Desouza, R. L. Mercer, V. J. D. Pietra, and J. C. Lai, “Class-based n-gram models of natural language,” Computational linguistics, vol. 18, no. 4, pp. 467–479, 1992.

[8] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” Proceedings of the IEEE, vol. 86, no. 11, pp. 2278–2324, 1998.

[9] H. Lee, P. Pham, Y. Largman, and A. Y. Ng, “Unsupervised feature learning for audio classification using convolutional deep belief networks,” in Advances in neural information processing systems, 2009, pp. 1096–1104.

[10] J. L. Elman, “Finding structure in time,” Cognitive science, vol. 14, no. 2, pp. 179–211, 1990.

[11] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” Neural computation, vol. 9, no. 8, pp. 1735–1780, 1997.

[12] M. Schuster and K. K. Paliwal, “Bidirectional recurrent neural networks,” Signal Processing, IEEE Transactions on, vol. 45, no. 11, pp. 2673–2681, 1997.

[13] Z. C. Lipton, “A critical review of recurrent neural networks for sequence learning,” arXiv preprint arXiv:1506.00019, 2015.

[14] R. Pascanu, T. Mikolov, and Y. Bengio, “On the difficulty of training recurrent neural networks,” arXiv preprint arXiv:1211.5063, 2012.

[15] Q. V. Le, N. Jaitly, and G. E. Hinton, “A simple way to initialize recurrent networks of rectified linear units,” arXiv preprint arXiv:1504.00941, 2015.

[16] J. H. Friedman, “Greedy function approximation: a gradient boosting machine,” Annals of statistics, pp. 1189–1232, 2001.