MFN: A Multi-hop Fusion Network for Traffic Accident Risk Prediction

Hang Gao
Institute of Public Safety Research
Department of Engineering Physics
Tsinghua University
Beijing, 100084, China
School of Safety Science
Tsinghua University
Beijing, 100084, China
mailmeki@163.com

Yiting Wang
Institute of Public Safety Research
Department of Engineering Physics
Tsinghua University
Beijing, 100084, China
School of Safety Science
Tsinghua University
Beijing, 100084, China
wangyiting20000311@163.com

Han Xing
Institute of Public Safety Research
Department of Engineering Physics
Tsinghua University
Beijing, 100084, China
School of Safety Science
Tsinghua University
Beijing, 100084, China
xh1992442@163.com

Yi Liu*
Institute of Public Safety Research
Department of Engineering Physics
Tsinghua University
Beijing, 100084, China
School of Safety Science
Tsinghua University
Beijing, 100084, China
Beijing Key Laboratory of City Integrated Emergency Response Science
Beijing 100084, China
liuyi@tsinghua.edu.cn

ABSTRACT

Traffic accident risk prediction plays a pivotal role in mitigating of road accidents, which often involve a multitude of factors such as traffic events, weather conditions, and road environments. Existing research has struggled to adequately capture and leverage the intricate interrelationships among these attributes for enhancing accident risk prediction. In this work, we introduce a multi-hop fusion network (MFN) for traffic accident risk prediction. The primary motivation behind MFN is that we believe the interrelationships among diverse attributes evolve over time. Our MFN incorporates multiple hops, progressively focusing on interactions between different attributes at each stage. This enhances the comprehension of underlying patterns and amplifies the precision of traffic accident prediction. We conduct extensive experiments on the data of 6 cities on the US-Accidents dataset. Experimental results illustrate significant performance advancements in traffic accident prediction achieved by our MFN.

CCS CONCEPTS

• Computing methodologies → Machine learning → Machine learning approaches • Computing methodologies → Artificial intelligence → Knowledge representation and reasoning

KEYWORDS

Traffic Accident Risk Prediction, Deep Learning, Neural Networks, Multi-hop Fusion.

ACM Reference format:

Hang Gao, Yiting Wang, Han Xing, Yi Liu. 2023. MFN: A Multi-hop Fusion Network for Traffic Accident Risk Prediction. In Proceedings of The 8th ACM SIGSPATIAL International Workshop on Security Response

EM-GIS '23, November 13, 2023, Hamburg, Germany

ACM ISBN 979-8-4007-0346-1/23/11.
https://doi.org/10.1145/3615884.3629425

using GIS 2023 (EM-GIS 2023). ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3615884.3629425

1 Introduction

Traffic safety has always been a matter of significant concern. Traffic accidents have negative impacts on drivers, passengers, and pedestrians, posing a grave threat to public safety. Therefore, scholars have been dedicated to predicting traffic accident risks. In recent years, with the advancement of information technology, extracting correlations and specific patterns among various factors related to historical traffic accident data and using them to predict traffic accident risks [1-2] has become a prominent research focus.

There are numerous factors associated with traffic accidents, such as people, vehicles, roads, the environment, and so on. We believe that there are two key points in delving into these factors. The first one is to construct better representations for heterogeneous factors. For instance, congestion and road construction change over time, and for predicting traffic accident risk at a specific time point, it is necessary to consider a period before that time point. In such cases, data describing congestion levels and road construction are in the form of time series. On the other hand, for buildings like schools or companies, previous work [1] treated them as static factors: their attributes do not change over time. Consequently, a static vector is typically used to represent such factors. Existing research [1], when jointly analyzing dynamic and static factors, aligns dynamic factors with static factors (aligning time series with static vectors), without considering that static factors may exhibit different attributes over time. For example, a school significantly affects traffic flow during peak school hours. Therefore, in this paper, we consider constructing time series for static factors, aligning static factors with dynamic factors, to account for the potential variations of static factors over time.

The second key point is how to capture the complex relationships among factors. It is well known that there is a wide variety of factors related to traffic accidents. Understanding the relationships between these factors and identifying which ones influence each other is crucial for predicting traffic accident risks. In this paper, we introduce a multi-hop fusion structure to achieve this goal. We believe that at different moments, the relationships between different factors are different. Therefore, we introduce the fusion structure to allow different factors to be fused at different time step based on their relationships. Additionally, multi-hop fusion is an incremental modeling approach, where each hop's fusion is based on the fusion from the previous hop, allowing it to capture more complex relationships and achieve multi-step reasoning. We name the model based on the above idea as Multi-hop Fusion Network (MFN). We conducted experiments on the US-Accidents dataset, and the experimental results indicate that our model outperforms previous research, while the above design achieved our expectations. In summary, the main contributions of this paper are as follows:

1. We propose constructing time series for static factors, allowing them to align with dynamic factors over time. This helps capture the varying attributes of static factors at different time steps.
2. We introduce MFN, a multi-hop fusion network, where the fusion structure enables the fusion of different factors at different time steps based on their relationships. Furthermore, as each hop builds upon the results of the previous hop, this allows the model to capture more intricate interrelationships among factors.
3. We conduct experiments on the US-Accidents dataset, experimental results illustrate that our MFN can achieve better performance on the task of traffic accident risk prediction.

2 Related Work

Many researchers have conducted research on traffic risk prediction tasks. In previous studies, a seminal work that laid the foundation is [1]. On one hand, it constructed a large-scale dataset using historical data of traffic accidents in the United States. On the other hand, it extended the relevant factors of traffic accidents, including weather, points-of-interest features (amenities, traffic signals, etc.) and so on. Subsequent research [2-5] has conducted based on this dataset.

Additionally, since historical traffic accident data is essentially tabular data, some studies [2] have assessed classic models for tabular data, including random forests [6], XGBoost [7], deep forests [8]. In this paper, we aim to enhance the performance of traffic accident risk prediction by focusing on constructing better representations and capturing more complex interrelationships among factors.

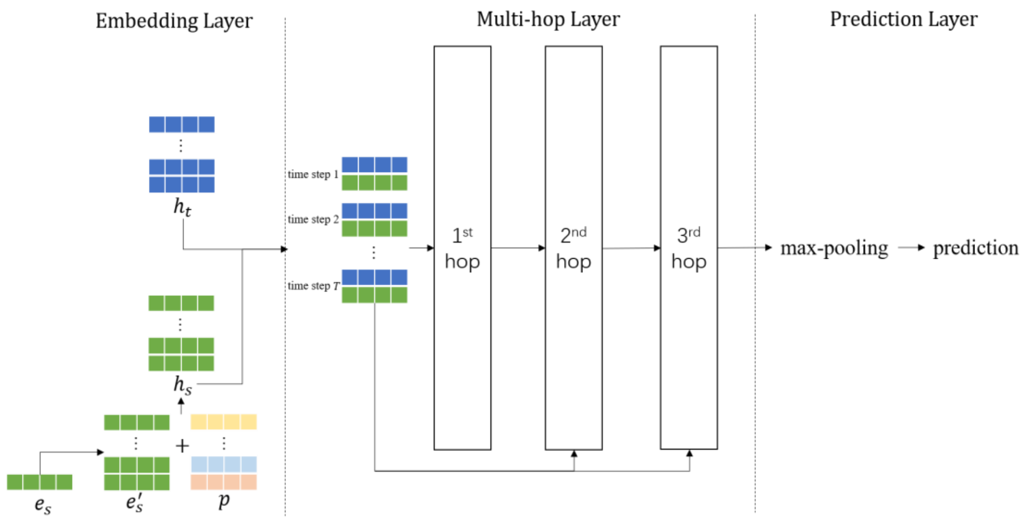


Figure 1 The overall architecture of our MFN. There are totally three layers, which are the embedding layer, the multi-hop layer, and the prediction layer (from left to right). The multi-hop layer is responsible for capturing more intricate interrelationships among factors

3 Methodology

In this section, we describe our multi-hop fusion network (MFN). It takes multiple features as input, including time-series features and static features, and generates a probability distribution that represents the prediction of traffic accident risk based on the current input.

3.1 Overall Architecture

Figure 1 illustrates the overall architecture of our MFN. It consists of 3 layers, which are the embedding layer, the multi-hop fusion layer and the prediction layer. In the following sections, we will provide a detailed description of each layer.

3.2 Embedding Layer

The embedding layer takes in the raw inputs and encodes them into embedding sequences. Here, we separately describe the encoding process for two types of inputs: time-series features and static features.

Let $x_t \in \mathbb{R}^{T \times d_t}$ denotes a time-series feature, where T is the number of time steps, and d_t is the dimension of each time step. We utilize a BiLSTM to encode x_t, and the output sequence of the BiLSTM is taken as the embedding sequence of x_t. Formally:

$$h_t = \text{BiLSTM}(x_t)$$

where $h_t \in \mathbb{R}^{T \times d}$ is the embedding sequence, and d is the dimension of each time step in h_t. Note that each time step in all embedding sequences constructed in the embedding layer has the same dimension, which is d.

Let $x_s \in \mathbb{R}^{d_s}$ denotes a static feature, where d_s is the dimension of the static feature. We first utilize a fully-connected feed-forward (FF) layer followed by a Tanh to map x_s to a d-dimensional vector, which is denoted as $e_s \in \mathbb{R}^d$. After that, we duplicate e_s T times, and stack them along a new axis to obtain $e_s' \in \mathbb{R}^{T \times d}$. That is to say, we allocate the current static features for each time step. In Section 1, we discussed that static features may have different semantics at different time steps. To learn these semantics, we construct a set of trainable position embeddings, which are denoted as $p \in \mathbb{R}^{T \times d}$. The embedding sequence of x_s can be calculated as follow:

$$h_s = p + e_s'$$

where $h_s \in \mathbb{R}^{T \times d}$. It's worth noting that these position embeddings differ from the position embeddings in the Transformer: we do not use them to emphasize the sequence order, but instead employ a set of trainable position embeddings to highlight the different semantics expressed by the current static features at different time step. Besides, all static features share the same position embeddings. So far, for M time-series features and N static features, we construct $M + N$ embedding sequences. We stack them along a new axis and obtain $h \in \mathbb{R}^{T \times (M+N) \times d}$, which is the output of the embedding layer.

3.3 Multi-hop Layer

In aligning static features with each time step, each time step will contain multiple features. In the multi-hop fusion layer, our primary objective is to establish the interrelations among the multiple features within each time step and subsequently fuse them based on these relationships. That is to say, the fusion of multiple features is conducted at each time step. This aligns with our intention, which is to capture varying interrelationships among multiple features at different time points. Technically, the fusion utilized in our MFN depicts the phenomenon where multiple features interact with each other in distinct ways at different moments.

Besides, it is well known that the relationships among traffic accident risk-related features are highly complex. Therefore, we introduce a multi-hop structure to enhance the model's capability to capture intricate interdependencies. The multi-hop layer consists of multiple fusion layer, known as hops. Except for the first hop, each subsequent hop is based on the results of the fusion in the previous hop. Here, we introduce the first hop, and then, as an example, explain the subsequent hops using the second hop. It should be noted that the total number of hops, denoted as H, is treated as a hyperparameter.

For the first hop, we utilize a single head attention to capture the interrelations among the multiple features within each time step. Let $h_i \in \mathbb{R}^{(M+N)\times d}$ denotes the i-th step in h, the alignment of h_i can be firstly calculated as follow:

$$a_{1,i} = \mathrm{softmax}\left(h_i h_i^T\right) h_i$$

where $a_{1,i} \in \mathbb{R}^{(M+N)\times d}$. After that, a max-pooling layer is applied to squeeze $a_{1,i}$, and $v_{1,i} \in \mathbb{R}^{d}$ can be obtained. For the total T steps, we can obtain the fused representation $v_1 \in \mathbb{R}^{T\times d}$. Then, a BiLSTM is utilized to encode v_1. Here, we adopt the residual connection:

$$h_1 = \mathrm{BiLSTM}(v_1 + h_{max})$$

where $h_1 \in \mathbb{R}^{T\times d}$, and $h_{max} \in \mathbb{R}^{T\times d}$, which is the max-pooled h.

For the second hop, we also utilize a single head attention to fuse the features of each time step. Note that in the single head attention of this hop, we take h_1 as the query, and h as the key and value. Formally, for the i-th step, we calculate the alignment in the second hop:

$$a_{2,i} = \mathrm{softmax}\left(h_{1,i} h_i^T\right) h_i$$

where $h_i \in \mathbb{R}^{(M+N)\times d}$, and $h_{1,i} \in \mathbb{R}^{1\times d}$. Similar to the first hop, we also adopt a max-pooling and a BiLSTM to obtain the encoded sequence $h_2 \in \mathbb{R}^{T\times d}$.

3.4 Prediction Layer

Suppose there are total H hops in the multi-hop fusion layer, we adopt a max-pooling to squeeze the encoded sequence $h_H \in \mathbb{R}^{T\times d}$ and obtain a representation $r \in \mathbb{R}^{d}$. After that, we feed r into a fully-connected classifier for prediction. The standard cross-entropy loss is adopted for training.

4 Experiments

In this section, we first introduce the datasets used in the experiment. Then, the experimental settings are introduced in detail. Finally, we analyze the experimental results and validate our methods.

4.1 Datasets

We evaluate our MFN on the US-Accidents Dataset [1]. Experiments are conducted on six cities, which are Atlanta, Austin, Charlotte, Dallas, Houston, and Los Angeles. Following the previous study, there are one time-series feature (including time-traffic-weather information), and three static features (including point-of-interest information, historical traffic events, and the index of the current region which contains information such as spatial heterogeneity and traffic characteristics). More details can be founded in [1]. All dataset configurations are identical to those in [1].

4.2 Settings

We implement our MFN with Pytorch (version 1.1.0) and train on one Tesla P100 GPU. All results reported in this paper are the average of 3 runs. We adopt the Adam [9] optimizer for training, with a learning rate set to 0.01 and a batch size of 128. We train the MFN for 50 epochs. We tune the number of hops among [1,2,3].

4.3 Experimental Analysis

Table 1 Comparison between MFN and previous methods in terms of F1-score for class of accidents (Acc). The numbers in parentheses indicate the number of hops that yielded the best performance. Best results are marked in **bold**.

Methods	Atlanta	Austin	Charlotte	Dallas	Houston	Los Angeles
LR	0.54	0.58	0.56	0.30	0.49	0.41
GBC	0.57	0.61	0.60	0.32	0.51	0.45
DNN	0.62	0.62	0.61	0.36	0.59	0.53
DAP-NoEmbed	0.62	0.62	0.61	0.43	0.58	0.53
DAP	**0.65**	0.64	0.63	0.50	0.58	0.56
MFN	**0.65(3)**	**0.67(2)**	**0.65(2)**	**0.51(2)**	**0.60(2)**	**0.57(3)**

Table 3 Comparison between MFN and previous methods in terms of F1-score for non-accidents (Non-Acc). The numbers in parentheses indicate the number of hops that yielded the best performance. Best results are marked in **bold**.

Methods	Atlanta	Austin	Charlotte	Dallas	Houston	Los Angeles
LR	**0.91**	**0.93**	**0.91**	**0.94**	**0.94**	**0.88**
GBC	**0.91**	**0.93**	**0.91**	**0.94**	**0.94**	**0.88**
DNN	0.89	0.92	0.87	0.94	0.93	0.81
DAP-NoEmbed	**0.91**	**0.93**	0.87	0.88	0.92	0.77
DAP	0.89	0.91	0.87	0.93	0.93	0.84
MFN	**0.91(3)**	0.92(2)	0.90(2)	0.93(2)	0.93(2)	0.85(3)

[1] https://github.com/mhsamavatian/DAP

Table 4 Comparison between MFN and previous methods in terms of F1-score for weighted average (W-avg). The numbers in parentheses indicate the number of hops that yielded the best performance. Best results are marked in **bold**.

Methods	Atlanta	Austin	Charlotte	Dallas	Houston	Los Angeles
LR	0.83	0.87	0.83	0.87	**0.88**	0.78
GBC	0.84	0.87	**0.84**	0.87	**0.88**	**0.79**
DNN	0.83	0.87	0.82	0.87	**0.88**	0.75
DAP-NoEmbed	0.84	0.87	0.81	0.83	**0.88**	0.72
DAP	0.84	0.87	0.82	**0.88**	**0.88**	0.78
MFN	**0.85(3)**	**0.88(2)**	**0.84(2)**	**0.88(2)**	**0.88(2)**	0.78(3)

4.3.1 Effectiveness analysis of MFN

The experimental results, including the F1-score for class of accidents (Acc), non-accidents (Non-Acc), and weighted average (W-avg) are illustrated in Table 2,3, and 4. The following conclusions can be drawn:

(1) MFN achieves the best performance in terms of Acc. as mentioned in previous studies, the class of accident deserves more attention [1]. Our MFN outperforms previous research significantly in the Acc metric across all six cities. This supports the superiority of our model.

(2) MFN achieves the best performance in terms of W-avg on five cities except for Los Angeles. Its performance surpasses DAP, which is also a neural network and achieves the best performance previously. On the dataset for Los Angeles, MFN's performance is very close to the best model GBC. Overall, MFN's performance on the W-avg metric is satisfactory.

(3) LR and GBC, which are popular and classic machine learning models, achieve the best performance in terms of Non-Acc. Nevertheless, our MFN outperforms DNN, DAP-NoEmbed, and DAP. That is to say, our MFN has achieved the best performance compared to other neural networks.

4.3.2 Ablation analysis

In this section, we focus on the impact of the number of hops on performance. As the most critical hyperparameters in MFN, the number of hops directly affects the model's ability to capture complex relationships. While keeping all other parameters fixed, we vary the number of hops and evaluate the model's performance. Figure 2 illustrates the experimental results.

From Figure 2, it can be observed that the sensitivity to the number of hops varies among cities, with some cities being highly sensitive (e.g., Atlanta, Austin, Charlotte, and LosAngeles), while others are less sensitive (e.g., Dallas, and Houston). In future work, we will attempt to quantify the criteria for selecting the number of hops and explore automated methods for determining the optimal number of hops.

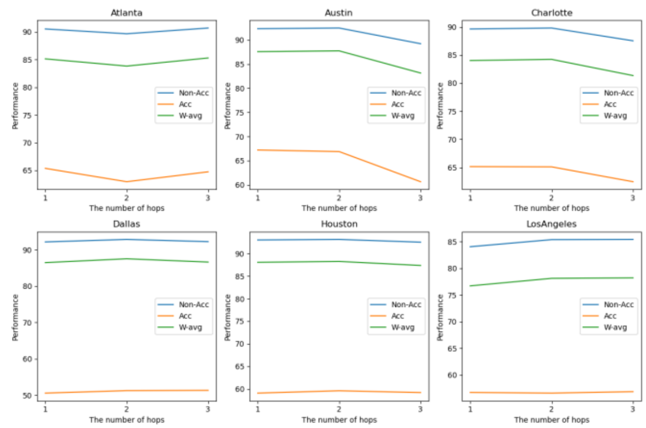
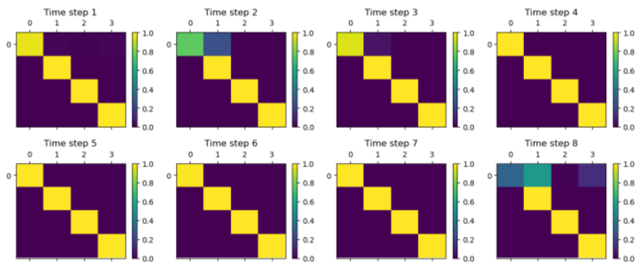


Figure 2 Performance with different numbers of hops.

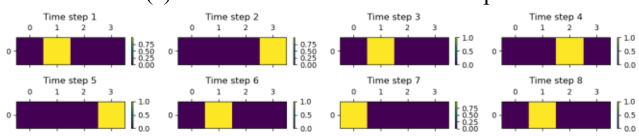
4.3.3 Visualization

In this section, we provide insights into how MFN captures intricate interrelationships among factors through visualization. We input a record from the dataset of Austin into the trained MFN and visualize the attention matrices computed by the model for the first hop and the second hop. Figure 3 displays the visual results. In each attention matrix, feature indexed as 0 is the dynamic feature about time-traffic-weather information, and features indexed as 1, 2, and 3 are all static features, which are point-of-interest, historical traffic events, and the index of the current region. For simplicity, we label them as F0, F1, F2 and F3.

As shown in Figure 3. Firstly, in the first hop, for most time steps (except for time steps 2 and 8), each factor has the highest correlation score with itself. In other words, the model cannot find interrelationships among factors in the first hop. This is reasonable because F1, F2 and F3 have all been defined as static features in previous studies [1], and they are indeed challenging to relate to dynamic features for the majority of time steps. However, in the attention matrices' first row at time steps 2 and 8, F0 establish associations with F1, F2, and F3, respectively. This partly supports our proposition that there are varying interrelationships among features at different time points. On the other hand, it indicates that static features do exhibit different properties at specific moments. Furthermore, in the second hop, different attention patterns are displayed at different time steps. This suggests the effectiveness of our multi-hop structure, as it can indeed capture complex interdependencies among factors, especially in the temporal dimension.



(a) attention matrices in the first hop



(b) attention matrices in the second hop

Figure 3 Visualization.

5 Conclusion

In this work, we introduce a novel multi-hop fusion network (MFN) for the task of traffic accident risk prediction. By analyzing the characteristics of traffic accident-related factors, we design a multi-hop fusion structure. This structure allows different factors to be fused at different time steps based on their relationships. Furthermore, within the multi-hop structure, each hop's fusion is based on the fusion results from the previous hop, enabling it to capture more complex relationships. Experiments are conducted on the data of 6 cities on the US-Accidents dataset. Experimental results demonstrate the superiority of our MFN.

ACKNOWLEDGMENTS

Funded by National Key R&D Program of China (No. 2021YFC3001500) and National Natural Science Foundation of China (No.72174102, No.91646101).

REFERENCES

[1] Moosavi, S., Samavatian, M. H., Parthasarathy, S., Teodorescu, R., & Ramnath, R. (2019, November). Accident risk prediction based on heterogeneous sparse data: New dataset and insights. In Proceedings of the 27th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (pp. 33-42).

[2] Gan, J., Li, L., Zhang, D., Yi, Z., & Xiang, Q. (2020). An alternative method for traffic accident severity prediction: using deep forests algorithm. Journal of advanced transportation, 2020, 1-13.

[3] Zhao, Y., & Deng, W. (2022). Prediction in traffic accident duration based on heterogeneous ensemble learning. Applied Artificial Intelligence, 36(1), 2018643.

[4] Liu, Z., Chen, Y., Xia, F., Bian, J., Zhu, B., Shen, G., & Kong, X. (2023). TAP: Traffic Accident Profiling via Multi-Task Spatio-Temporal Graph Representation Learning. ACM Transactions on Knowledge Discovery from Data, 17(4), 1-25.

[5] Wu, M., Jia, H., Luo, D., Luo, H., Zhao, F., & Li, G. (2023). A multi‐attention dynamic graph convolution network with cost‐sensitive learning approach to road‐level and minute‐level traffic accident prediction. IET Intelligent Transport Systems, 17(2), 270-284.

[6] Breiman, L. (2001). Random forests. Machine learning, 45, 5-32.

[7] Chen, T., & Guestrin, C. (2016, August). Xgboost: A scalable tree boosting system. In Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining (pp. 785-794).

[8] Zhou, Z. H., & Feng, J. (2019). Deep forest. National science review, 6(1), 74-86.

[9] Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.