DEPARTMENT OF INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Master´s Thesis in Informatics: Robotics, Cognition, Intelligence

Anomaly Detection for the behavior of drivers based on Structural Temporal Graph Neural Networks

Hanxi Jiang

DEPARTMENT OF INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Thesis in Informatics: Robotics, Cognition, Intelligence

Anomaly Detection for the behavior of drivers based on Structural Temporal Graph Neural Networks

Erkennung von Anomalien im Verhalten von Autofahrern auf der Grundlage struktureller temporaler neuronaler Netze

Author:	Hanxi Jiang
Supervisor:	Supervisor
Advisor:	Advisor
Submission Date:	Submission date

I confirm that this master’s thesis in informatics: robotics, cognition, intelligence is my own work and I have documented all sources and material used.

Munich, Submission date Hanxi Jiang

Acknowledgments

Abstract

Kurzfassung

Contents

1. Introduction

Despite the fact that High Driving Automation(SAE Level 4) is achievable in the foreseeable future[1], the majority of drivers nowadays would still prefer comprehensive control over their own vehicles. Therefore, increasingly more studies have been focusing on driving safety[2, 3]. According to these papers, driver behavior represents the majority cause of accidents while driving. In that case, several methods have been developed to avoid potential danger. Such methods involve either improving monitoring of the vehicle's inside situation, which analyzes the driving parameters as well as the driver him or herself to determine whether there's no abnormality[4], or proposing a vehicle detection and tracking system from an outer view that estimates time-to-collision (TTC) and warn the driver for a possible collision[5].

On the other hand, only few research focus on driving behavior prediction, which may be due to the lack of application for the undetermined decision model in behavior prediction, specifically under the topic of autonomous driving. Scientists have succeeded in years of generating dynamic graphs based on videos. However, there is still a gap in the rational use of these forms for behavioral prediction.

In this work, we would like to focus on constructing a comprehensive behavior prediction model based on graph neural networks (GNNs). A Graph Neural Network is a novel type of neural network architecture that can be applied to graph-like inputs. As we are now expecting to train the behavior model for the participators, the training data would contain interaction between several objects and participators while they are driving. GNN is, therefore, prioritized due to its unique structure. Along with that, we would specifically focus on building dynamic graphs as training datasets, as our model would be trained based on sequences of behaviour descriptions extracted from videos. Besides, to ensure the compatibility of the dataset and training model, we made some adaptions based on JODIE[6], which is the model based on the dynamic evolution of users and items. In order to make the predictions as detailed as possible, we expanded the output catalog to ensure that not only the interaction itself but also the type of it would be described.

In a nutshell, We would first gather all the necessary information with the help of the Large language model (LLM), convert it into dynamic graphs, and insert these graphs into the model we have adapted from model JODIE. These graphs could contain either one specific participant or all the concerned people. By learning how dynamic graphs change under time series, like the appearance and vanish of all these edges and nodes in the graph, the model should be able to absorb the feature behind it and come up with the prediction for behaviour in the future. Therefore, anomaly detection could also be achieved by comparing the predicted graph with the real one and alerting once when any unsuitable behaviour during driving is detected. we believe that this model provides a new perspective for the prediction of driving behavior detected from videos and allows for more diversified and targeted forecasting.

1.1. contribution

The main contributions of this thesis are summarized as:

Dataset Collecton From the Dataset *drive & act*, we acquire the hierarchical activity labels of given video data and rewrite them in the form of time sequences. We also cluster the behavior types into several categories with the help of the Large Language Model.

Dynamic Graph Construction By reassembling the nodes and edges in the video data, we construct time-sequenced dynamic graphs that could be used as training data for the model. Such graph captures complex dependencies and offers hierarchical representation abstracted from the raw data.

Learning Model Adaption To enrich the diversity of the prediction, we expand the output of the dynamic graph based learning model from binary to catalog description, to ensure that not only the interaction itself but also the type of it would be described.

anormaly detection By comparing the predicted model with the real one, we could detect any unsuitable behavior during the driving and alert the driver. Differ from the research before, our

1.2. Structure

This thesis is structured as follows. In Chapter2 we would introduce and explain concepts and definitions concerned this document. Chapter3 reviews related work in the field of anomaly detection and dynamic link prediction models, the dataset this work refers to and the model we have adapted. Chapter4 describes the methodology of this work, including the dataset collection, dynamic graph construction and model adaption. Chapter5 presents the evaluation of the model and the results of the prediction. Chapter6 discusses the potential future work that could be done based on this work. Chapter7 concludes the thesis and gives a summary of the work done.

2. Background

2.1. Large Language Model

Large Language Models (LLMs) are highly complex artificial intelligence systems that can learn from the vast amounts of available text data[7]. Thanks to the attending of *Transformer*[8], a deep learning architecture, these language models which employed self-supervised pre-training have demonstrated improved efficiency and scalability in many fields.

Based on self-attention mechanisms and feed-forward module, *Transformer* has overwhelming advantages in computing representations and global dependencies.

In concrete terms, the Large Language Models equipped with *Transformer* are capable of diverse tasks raised by Natural Language Processing[9], such as textual entailment, question answering, semantic similarity assessment, and document classification. Take BERT[10] and GPT[7, 11, 12] as two examples, the former utilizes transformer encoder blocks to predict missing words in a given text, and the latter has been enjoying a tremendous reputation for generating diverse and human-like responses, showcasing its potential in various domains.

2.1.1. Text Classification

In our project, a bunch of hierarchical activity labels would be acquired from the dataset *drive & act* to depict every detail of the participant's movements in the driving behavior recorded in the video. These labels, however, are too trivial for the construction of learning data, as considering each activity individually will be tedious in such a vast and complex model training process. Therefore, labels should be classified according to the object on which this behavior operates or the specificity of the moment in which the action takes place. For example, the fastening of a seat belt should occur shortly after entering the vehicle, and all behavior related to eating or drinking should be classified into the same group.

In our task, we use zero-shot text classification. This is a task where a model is trained on a set of labeled examples and then classifies new examples from previously unseen classes. This method, which leverages a pre-trained language model, can be thought of as an instance of transfer learning which generally refers to using a model trained for one task in a different application than what it was originally trained for. This is particularly useful for situations where the amount of labeled data is small, for example, our work with 39 different behaviors to be classified.

The model used in the pipeline is *BART* [13]. According to the paper, BART is trained by corrupting text with an arbitrary noising function and learning a model to reconstruct the original text. It uses a standard Tranformer-based neural machine translation architecture, which, despite its simplicity, can be seen as generalizing BERT (due to the bidirectional encoder), GPT (with the left-to-right decoder), and many other more recent pre-training schemes. With all these prerequisites, it is quite obvious that *BART* is a very practical model for our classification task.

2.2. Graph

In graph theory, a graph is a structure made up of a collection of objects, where certain pairs of these objects are connected in a specific way[14]. As a powerful tool for modeling and analyzing complex systems, researchers have combined graph theory with deep learning to solve various problems in different fields, such as social networks[15], complex physic system[16] and Protein-protein interactions (PPIs)[17].

A Graph is $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is a pair of sets, where collections of *nodes* $\mathcal{V}$ and edges $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ between pairs of nodes. The nodes here are assumed as the nodes to be endowed with s-dimensional *node feature*, denoted by x_u for all $u \in \mathcal{V}$. Taking social network as an example, nodes represent users, and edges correspond to the friendship relations between them. The features of nodes model user properties such as age, likelihood, etc. Also, some models, including this work, would endow the edges with features. In generic setting $\mathcal{E} \neq 0$, the node features are rows of the $x \times d$ matrix $X = (x_1, \ldots, x_n)^T$. To apply some function to update the features in every node, obtaining the set of latent node feature, we would introduce permutation equivariant function $H = F(X)$, where H is the feature matrix whose uth row is the latent feature of node u.

As for the edges $e \in \mathcal{E}$, or the graph connectivity, they can be represented by the $n \times n$ adjacency matrix A defined as

$$a_{uv} = \begin{cases} 1 & (u, v) \in \mathcal{E} \\ 0 & otherwise \end{cases}$$

Here a_{uv} specifies the adjacency information between the nodes described by the uth and the vth rows of X. Most functions acting on graphs can be viewed as the generators for 'local' node-wise output, i.e., whereby the output on node u directly depends on its neighboring nodes in the graph. It is worthwhile formalizing this constraint explicitly in our model construction by defining what it means for a node to be neighboring another. An (undirected) neighborhood of node u is defined as

$$\mathcal{N}_u = \{v : (u, v) \in \mathcal{E} \text{ or } (v, u) \in \mathcal{E}\}$$

and the neighborhood features as the multiset[18].

$$\mathcal{X}_{\mathcal{N}_u} = x_v : v \in N$$

Thus, the features of a node as well as its neighborhood could be collected simultaneously by a local function $\phi(x_u, \mathcal{X}_{\mathcal{N}_u})$. Then the permutation equivariant function F can be constructed by applying ϕ to every node's neighborhood in isolation (Figure 2.1).

$$F(X,A) = \begin{bmatrix} -- & \phi(x_1, \mathcal{X}_{\mathcal{N}_1}) & -- \\ -- & \phi(x_2, \mathcal{X}_{\mathcal{N}_2}) & -- \\ & \vdots & \\ -- & \phi(x_n, \mathcal{X}_{\mathcal{N}_n}) & -- \end{bmatrix}$$

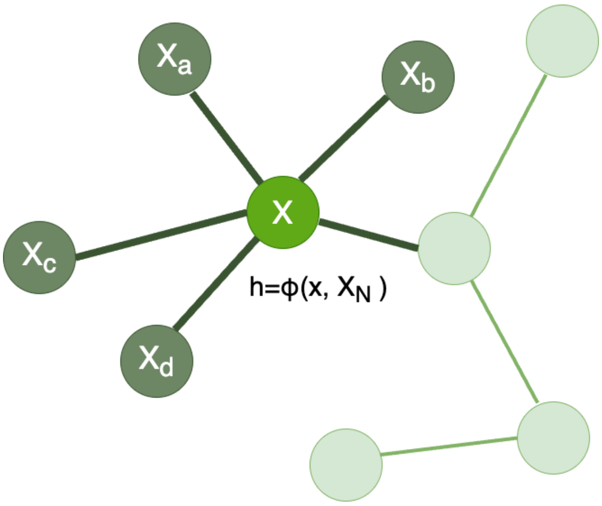


Figure 2.1.: Constructing permutation equivariant functions over graphs.

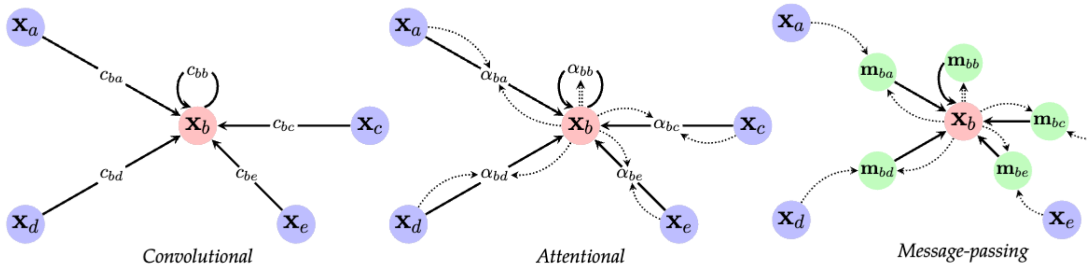
2.2.1. Graph Neural Networks

The intuitive understanding among **Graph Neural Network (GNN)** is that nodes in a graph represent objects or concepts, and edges represent their relationships. Each concept is naturally defined by its features and the related concepts[19]. GNNs are among the most general class of deep learning architectures currently in existence, and most learning architectures can be understood as a special case of the GNN with additional geometric structure.

Given to the mathematic expression we have settled in section 2.2 we consider a graph to be specified with an adjacency matrix A and node feature X. The GNN architectures we study arepermutation equivariant functions $F(X, A)$ construcetd by applying shared permutation invariant function $\phi(x_u, \mathcal{X}_{\mathcal{N}_u})$ over local neighborhood. This function ψ would be referred to as "diffusion", "propagation", or "message passing", and the all over computation of such of such F as a "GNN layer".

The design and study of GNN layers is one of the most active area of deep learning in the recent past few years. And three vast "flavours" of GNN layers are derived from the vast majority of the literatures. These flavours govern the extent to which ψ transforms the neighbourhood features, allowing for varying degrees of complexity when modelling interactions across the graph(Figure 2.2).

In all the three flavours, the feature update function for node u would apply permutation-invariant function $\oplus$ to aggregate features from $\mathcal{X}_{\mathcal{N}_u}$,as well as learnable function ψ. The



Convolutional *Attentional* *Message-passing*

Figure 2.2.: A visualisation of the dataflow for the three flavours of GNNlayers. Reproduced from [18]

function $\oplus$ is potentially transformed, by means of some function ψ and itself is usually realized as a nonparametric operation such as sum, mean, or maximum.

In the **convolution flavour**[20, 21, 22]The feature of the neighborhood nodes are directly aggregated with fixed weights c_{uv}, which often directly depends on the entries in A representingthe structure of the graph. The update function is defined as:

$$\mathbf{h}_u = \phi\left(\mathbf{x}_u, \bigoplus_{v \in \mathcal{N}_u} c_{uv}\psi(\mathbf{x}_v)\right).$$

In the **attention flavour**[23, 24, 25]the weights c_{uv} are replaced by the a learnable self-attention mechanism that computes the importancecoefficients. And the update function is defined as:

$$\mathbf{h}_u = \phi\left(\mathbf{x}_u, \bigoplus_{v \in \mathcal{N}_u} a(x_u, x_v)\psi(\mathbf{x}_v)\right).$$

And the **message-passing flavour**[26, 27] aims at computing arbitrary vectors of neighborhood $\mathcal{N}_u$ send to u, where vector-based messages are computed based on both the sender and receiver: $m_{uv} = \psi(x_u, x_v)$.

$$\mathbf{h}_u = \phi\left(\mathbf{x}_u, \bigoplus_{v \in \mathcal{N}_u} \psi(x_u, x_v)\right).$$

What matters is that a representational containment between these approaches exists: $convolution \in attention \in message - passing$. This, however, does not mean that message-passing is always the most useful choice. Indeed, the vector-based messages always contains most oriented information across the edges, but they are harder to train due to the enormous requirement of memories. And here, in our work, we have such a specific network that none of the approaches above is the best choice.

2.2.2. Dynamic graph

Our discussion so far has focused solely on input that exhibits spatial variations across a given domain; however, the input may also change over time, for instance, video, text or speech. Assume that the input consists of arbitrary steps, and an input signal will be provided at each step t, we represent as $X^{(t)\in\mathcal{X}(\omega^{(t)})}$

While in many cases the domain in kept fixed across time t, we notice that exceptions also happens. For example in our work, we many find tremendous changes in the videos from our dataset. Such domain as changing over time is referred to as *dynamic graph*[28, 29].

Assume an encoder function $f(X^{(}t))$ offering latent presentation for such dynamic graph learning problem. In the video analysis, ω is a fixed grid, and signals are a sequence of frames. To have a view of the entire frame, one of the options is to implement f as a translation invariant CNN, outputting a k-dimensional representation $z(t) = f(X^{(}t))$ of the frame at time-step t. In our work, however, we use an alternative approaches, which we would introduce in chapter3.

To have a canonical analysis for dynamically aggregate the sequence of vectors $z(t)$, we introduce *Recurrent Neural Network* (RNN)[30] due its advantages in the field of temporal progression of inputs and online arrival of novel data-points. A clear view of analyzing video data with RNN is given in Figure 2.3.

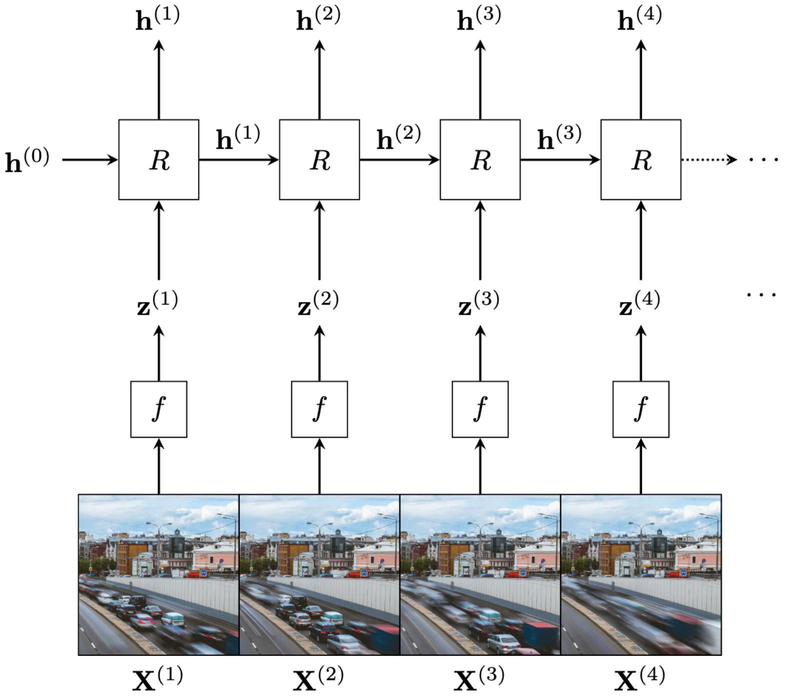


Figure 2.3.: Illustration of processing video input with RNNs. Reproduced from [18]

A simple RNN model is defined as the following way: The model holds with an mdimensional summary vector $h(t)$ of all the input steps up to and including t. This will summarize the current step's features as well as the information collected from the past, through a shared function $R : \mathbb{R}^k \times \mathbb{R}^m \to \mathbb{R}^m$, as follows:

$$h(t) = R(z(t), h(t-1))$$

As both $z(t)$ and $h(t-1)$ are vectors, the function R is usually realized as a simple feed-forward neural network(often known as *Simple RNN*[31, 32]):

$$h(t) = \sigma(W_z z(t) + Uh(t-1) + b)$$

Where W_z and U are the weight matrices, b is the bias, and σ is the activation function.

The summary vectors may then be appropriately leveraged for the down-stream taskif a prediction is required at every step of the sequence, then a shared predictor may be applied to each $h(t)$ individually. For classifying entire sequences, typically the final summary, $h(T)$, is passed to a classifier, just like many works including ours.

2.3. behavior prediction for human driver

The earliest research regarding the prediction of human driver behavior could be dated back to the 1990s, when hidden Markov models are first introduced to model the driver's behavior[33]. An HMM is a statistical model that represents a system with hidden states, where observable outputs are generated by these underlying, unobservable states. Each state transitions to another with a certain probability and emits observations based on specific probability distributions.

The paper is constructed on the ground truth that the intended actions, like to turn or change lanes, are modeled as a sequence of internal states. By observing the temporal patterns of the driver's behavior or even intention. Although the attempt then is to capture and predict driver eye movements in the context of lane keeping/curve negotiation and car following to determine. It still proves that characteristic patterns in driver behavior could be identified and even represent by mathematical models.

With the advancement of various machine learning models, more sophisticated tools have emerged, expanding the range of options available for predicting human driver behavior. In [34], hidden Markov models are adapted for this topic as well, yielding promising results due to the enhanced computational capabilities available today. Another study [35] proposes an LSTM-based trajectory prediction method for human drivers, which facilitates better decision-making for autonomous vehicles, particularly in urban intersection scenarios. Despite these significant achievements, there remains a noticeable gap in driver behavior prediction for driver-oriented dynamic scene graphs. Such an approach could capture individual driving habits, enabling more precise predictions and even anomaly detection tailored to each driver. Our work aims to address this gap.

3. Related Work

3.1. Datasets for driver behavior analysis

As of the time of writing, numerous datasets are available for driver behavior analysis, each with its own specific focus. For instance, **DR(eye)VE Dataset**[36]capturing drivers' gaze patterns in real-world scenarios, includes ego-centric views and car-centric views, which provide data on where drivers look in different driving contexts. **Naturalistic Driving Study (NDS)**[37] contains extensive data collected from naturalistic driving conditions, including in-vehicle driver behavior such as driver movements, eye gaze, and interactions with vehicle controls during different driving scenarios.

In our work, we utilize the **Drive & Act** dataset[38] as our primary training resource. This dataset includes over twelve hours and 9.6 million frames, capturing individuals engaged in various distractive activities during both manual and automated driving. It is specialized for distinguishing between closely related actions (e.g., opening a bottle vs. closing a bottle) and features a high diversity in action durations and complexities, presenting unique challenges for action recognition models. For instance, brief actions like opening a door from the inside may take less than a second, while prolonged activities such as reading a magazine may last several minutes. This dataset is exceptionally well-suited to our study, as it provides a comprehensive view of driver behavior within the vehicle, with accurately labeled, temporally sequenced annotations for each behavior.

3.2. Dynamic Scene Graph for Video

As is revealed in chapter2, the most common way to represent the latent information in a video is to use CNNs to directly extract the features of the frames and then use RNNs to model the temporal information. However, this method has its limitations as the excessive high-dimensional abstraction may cause the model to misinterpret human learning intentions; in other words, the black box itself is difficult to interpret. In our work we would like to extract the dynamic scene graph from the video, which is a more intuitive way to represent the video content.

The scene graph is a structured representation of a scene that can clearly express the objects, attributes, and relationships between objects in the scene[39]. Accompanied by the development of computer vision technology, simply detecting and recognizing objects in images no longer satisfies the researchers, as they would expect some higher level of understanding and reasoning for image vision tasks. In this way, an intuitive idea comes up about adding up the relationship between the detected objects (See example in figure3.2). The

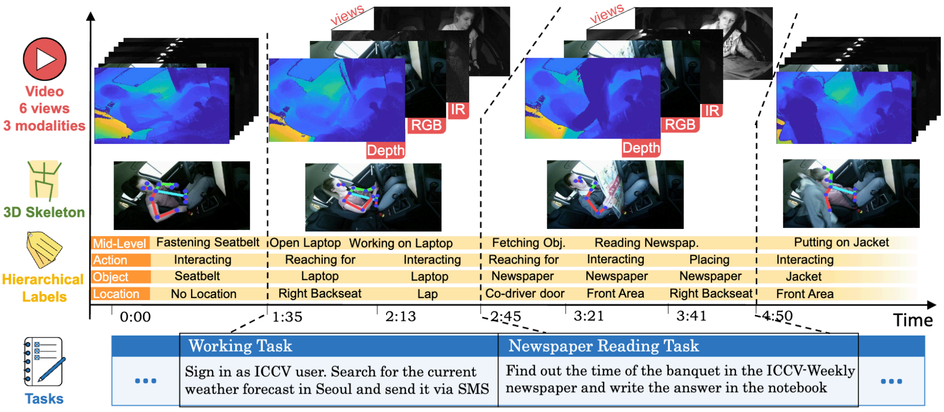


Figure 3.1.: Overview of the Drive&Act dataset for driver behavior recognition. Reproduced from[38]

earliest research could date back to 2017, when some objects and relations of a given image could be inferred, and a scene graph would be produced as a result[40]. Other research like Neural Motifs[41] also shows the possibility of predicting the most frequent relation between object pairs with the given labels and object detections. Later in 2018, videos came into discussion, and both spatial and temporal relations would be concerned in the dynamic graph researchers propose to represent[42]. Meanwhile, the accuracy of Scene Graph Detection tasks has significantly improved thanks to the application of unbiased SGG[42], fueled by the **Detection2** [43], a library that contains various state-of-the-art detection and segmentation algorithms. In a nutshell, representing videos as dynamic scene graphs including the detection of objects and the relations in between has been realized in the past years. And our work would utilize such technique and extract our driver-oriented dynamic scene graphs for further learning.

3.3. Dynamic Link Prediction

As

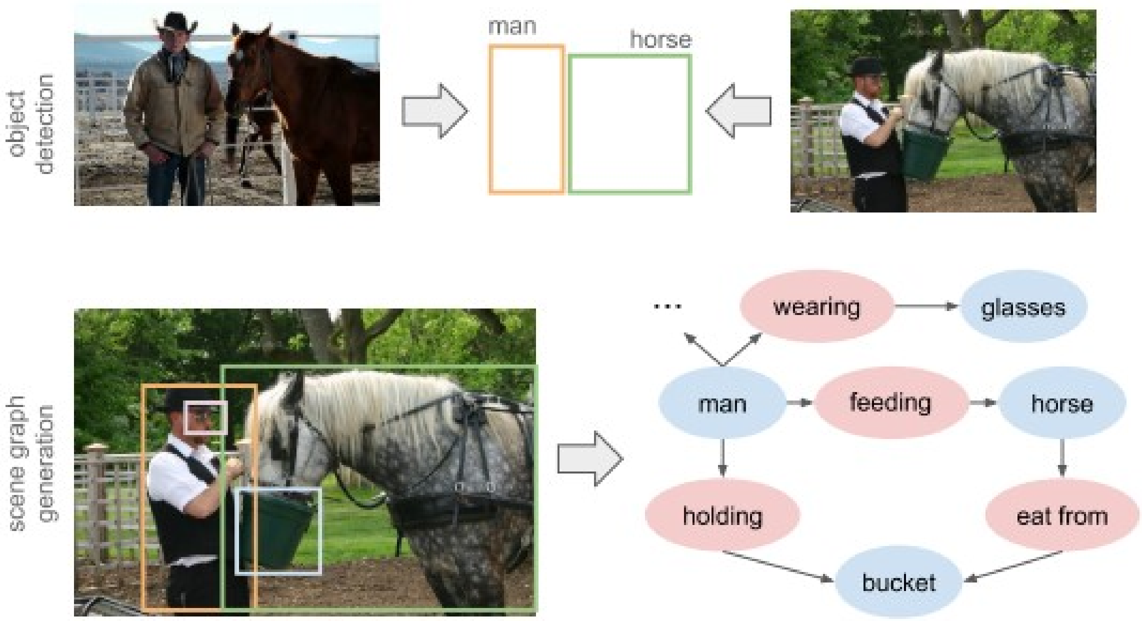


Figure 3.2.: Scene Graph Generation by Iterative Message Passing. Reproduced from[44]

4. Methodology

This work aims to construct a graph neural network-based architecture for predicting, analyzing, and detecting any potentially abnormal behavior regarding the driver during the whole driving process. In particular, The model extracts a description graph, the so-called scene graph, of the driver from the video filmed inside the vehicle and trains itself with these data to learn for future behavior prediction. The result will be used to compare and detect any abnormal behavior. Here we would lay most emphasis on the construction of the training model. To make precious anomaly detection we aim to predict not only if there is a behavior between humans and a specific kind of object but the type of behavior as well, which will cause several adaptions based on existing model *JODIE*.

4.1. Scene graph generation

4.1.1. Video data extracting

4.1.2. Graph generating

4.2. Model architecture

After comparing all the training results of the below models we would find that **JODIE** is one coming up with the best prediction. However, the model jodie still fail to predict the state of the predicted edge. In my masterwork I would like to rewrite the embedding function and the loss function of **JODIE to make the state prediction possible.

- function from **JODIE**:
embedding function

$$\mathbf{u}(\mathbf{t}) = \sigma(W_1^u \mathbf{u}(\mathbf{t}^-) + W_2^u \mathbf{i}(\mathbf{t}^-) + W_3^u f + W_4^u \Delta_u)$$

$$\mathbf{i}(\mathbf{t}) = \sigma(W_1^i \mathbf{i}(\mathbf{t}^-) + W_2^i \mathbf{u}(\mathbf{t}^-) + W_3^i f + W_4^i \Delta_i)$$

loss function(BCE)

$$L = -(j_{pos} \log \tilde{j} + j_{neg} log(1 - \tilde{j}))$$

where

$$\tilde{j}(t + \Delta) = W_1 \hat{u}(t + \delta) + W_2 \bar{u} + W_3 i(t + \Delta^-) + W_4 \bar{i} + B$$

- functions adapted in my work:

embedding function

$$\mathbf{u}(\mathbf{t}) = \sigma(W_1^u \mathbf{u}(\mathbf{t}^-) + W_2^u \mathbf{i}(\mathbf{t}^-) + W_3^u f + W_4^u s + W_5^u \Delta_u)$$

$$\mathbf{i}(\mathbf{t}) = \sigma(W_1^i \mathbf{i}(\mathbf{t}^-) + W_2^i \mathbf{u}(\mathbf{t}^-) + W_3^i f + W_4^i s + W_5^u \Delta_i)$$

we will change it from BCE to CE for predictiing state.

$$\tilde{j}(t + \Delta) = W_1 \hat{u}(t + \delta) + W_2 \bar{u} + W_3 i(t + \Delta^-) + W_4 \bar{i} + W_5 s + B$$

5. Evaluation

6. Future Work

7. Conclusion

A. General Addenda

If there are several additions you want to add, but they do not fit into the thesis itself, they belong here.

A.1. Detailed Addition

Even sections are possible, but usually only used for several elements in, e.g. tables, images, etc.

B. Figures

B.1. Example 1

✓

B.2. Example 2

✗

List of Figures

List of Tables

Bibliography

[1] T. Inagaki and T. B. Sheridan. "A critique of the SAE conditional driving automation definition, and analyses of options for improvement". In: *Cognition, technology & work* 21 (2019), pp. 569–578.

[2] J. D. Lee. "Driving safety". In: *Reviews of human factors and ergonomics* 1.1 (2005), pp. 172–218.

[3] J. D. Lee. "Fifty years of driving safety research". In: *Human factors* 50.3 (2008), pp. 521–528.

[4] M. Karrouchi, I. Nasri, M. Rhiat, I. Atmane, K. Hirech, A. Messaoudi, M. Melhaoui, and K. Kassmi. "Driving behavior assessment: a practical study and technique for detecting a driver's condition and driving style". In: *Transportation Engineering* 14 (2023), p. 100217.

[5] B. Aytekin and E. Altu. "Increasing driving safety with a multiple vehicle detection and tracking system using ongoing vehicle shadow information". In: *2010 IEEE International Conference on Systems, Man and Cybernetics*. IEEE. 2010, pp. 3650–3656.

[6] S. Kumar, X. Zhang, and J. Leskovec. "Predicting dynamic embedding trajectory in temporal interaction networks". In: *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*. 2019, pp. 1269–1278.

[7] A. Radford. "Improving language understanding by generative pre-training". In: (2018).

[8] A. Vaswani. "Attention is all you need". In: *Advances in Neural Information Processing Systems* (2017).

[9] K. Chowdhary and K. Chowdhary. "Natural language processing". In: *Fundamentals of artificial intelligence* (2020), pp. 603–649.

[10] S. Alaparthi and M. Mishra. "Bidirectional Encoder Representations from Transformers (BERT): A sentiment analysis odyssey". In: *arXiv preprint arXiv:2007.01127* (2020).

[11] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, et al. "Language models are unsupervised multitask learners". In: *OpenAI blog* 1.8 (2019), p. 9.

[12] T. B. Brown. "Language models are few-shot learners". In: *arXiv preprint arXiv:2005.14165* (2020).

[13] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer. *BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension*. 2019. arXiv: `1910.13461 [cs.CL]`. URL: `https://arxiv.org/abs/1910.13461`.

[14] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun. “Graph neural networks: A review of methods and applications”. In: *AI open* 1 (2020), pp. 57–81.

[15] Y. Wu, D. Lian, Y. Xu, L. Wu, and E. Chen. “Graph convolutional networks with markov random field reasoning for social spammer detection”. In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 34. 01. 2020, pp. 1054–1061.

[16] A. Sanchez-Gonzalez, N. Heess, J. T. Springenberg, J. Merel, M. Riedmiller, R. Hadsell, and P. Battaglia. “Graph networks as learnable physics engines for inference and control”. In: *International conference on machine learning*. PMLR. 2018, pp. 4470–4479.

[17] A. Fout, J. Byrd, B. Shariat, and A. Ben-Hur. “Protein Interface Prediction using Graph Convolutional Networks”. In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett. Vol. 30. Curran Associates, Inc., 2017. URL: `https://proceedings.neurips.cc/paper_files/paper/2017/file/f507783927f2ec2737ba40afbd17efb5-Paper.pdf`.

[18] M. M. Bronstein, J. Bruna, T. Cohen, and P. Velikovi. “Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges”. In: *arXiv* (2021). DOI: `10.48550/arxiv.2104.13478`. eprint: `2104.13478`.

[19] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini. “The Graph Neural Network Model”. In: *IEEE Transactions on Neural Networks* 20.1 (2009), pp. 61–80. ISSN: 1045-9227. DOI: `10.1109/tnn.2008.2005605`.

[20] T. N. Kipf and M. Welling. “Semi-supervised classification with graph convolutional networks”. In: *arXiv preprint arXiv:1609.02907* (2016).

[21] M. Defferrard, X. Bresson, and P. Vandergheynst. “Convolutional neural networks on graphs with fast localized spectral filtering”. In: *Advances in neural information processing systems* 29 (2016).

[22] F. Wu, A. Souza, T. Zhang, C. Fifty, T. Yu, and K. Weinberger. “Simplifying graph convolutional networks”. In: *International conference on machine learning*. PMLR. 2019, pp. 6861–6871.

[23] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, Y. Bengio, et al. “Graph attention networks”. In: *stat* 1050.20 (2017), pp. 10–48550.

[24] F. Monti, D. Boscaini, J. Masci, E. Rodola, J. Svoboda, and M. M. Bronstein. “Geometric deep learning on graphs and manifolds using mixture model cnns”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 5115–5124.

[25] J. Zhang, X. Shi, J. Xie, H. Ma, I. King, and D.-Y. Yeung. “Gaan: Gated attention networks for learning on large and spatiotemporal graphs”. In: *arXiv preprint arXiv:1803.07294* (2018).

[26] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl. "Neural message passing for quantum chemistry". In: *International conference on machine learning*. PMLR. 2017, pp. 1263–1272.

[27] P. W. Battaglia, J. B. Hamrick, V. Bapst, A. Sanchez-Gonzalez, V. Zambaldi, M. Malinowski, A. Tacchetti, D. Raposo, A. Santoro, R. Faulkner, et al. "Relational inductive biases, deep learning, and graph networks". In: *arXiv preprint arXiv:1806.01261* (2018).

[28] E. Rossi, B. Chamberlain, F. Frasca, D. Eynard, F. Monti, and M. Bronstein. "Temporal graph networks for deep learning on dynamic graphs. arXiv 2020". In: *arXiv preprint arXiv:2006.10637* (2020).

[29] D. Xu, C. Ruan, E. Korpeoglu, S. Kumar, and K. Achan. "Inductive representation learning on temporal graphs". In: *arXiv preprint arXiv:2002.07962* (2020).

[30] R. M. Schmidt. "Recurrent neural networks (rnns): A gentle introduction and overview". In: *arXiv preprint arXiv:1912.05911* (2019).

[31] J. L. Elman. "Finding structure in time". In: *Cognitive science* 14.2 (1990), pp. 179–211.

[32] M. I. Jordan. "Serial order: A parallel distributed processing approach". In: *Advances in psychology*. Vol. 121. Elsevier, 1997, pp. 471–495.

[33] A. Pentland and A. Liu. "Modeling and Prediction of Human Behavior". In: *Neural Computation* 11.1 (1999), pp. 229–242. ISSN: 0899-7667. DOI: `10.1162/089976699300016890`.

[34] Q. Deng, J. Wang, and D. Soffker. "Prediction of human driver behaviors based on an improved HMM approach". In: *2018 IEEE Intelligent Vehicles Symposium (IV)* 00 (2018), pp. 2066–2071. DOI: `10.1109/ivs.2018.8500717`.

[35] Z. Qiao, J. Zhao, J. Zhu, Z. Tyree, P. Mudalige, J. Schneider, and J. M. Dolan. "Human Driver Behavior Prediction based on UrbanFlow*". In: *2020 IEEE International Conference on Robotics and Automation (ICRA)* 00 (2020), pp. 10570–10576. DOI: `10.1109/icra40945.2020.9196918`.

[36] A. Palazzi, D. Abati, F. Solera, R. Cucchiara, et al. "Predicting the driver's focus of attention: the dr (eye) ve project". In: *IEEE transactions on pattern analysis and machine intelligence* 41.7 (2018), pp. 1720–1733.

[37] M. Regan, A. Williamson, R. Grzebieta, and L. Tao. "Naturalistic driving studies: literature review and planning for the Australian naturalistic driving study". In: *Proc. Australasian College Of Road Safety Conference, A Safe System: Expanding The Reach, Sydney*. 2012.

[38] M. Martin, A. Roitberg, M. Haurilet, M. Horne, S. ReiSS, M. Voit, and R. Stiefelhagen. "Drive Act: A Multi-Modal Dataset for Fine-Grained Driver Behavior Recognition in Autonomous Vehicles". In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019, pp. 2801–2810. DOI: `10.1109/ICCV.2019.00289`.

[39] X. Chang, P. Ren, P. Xu, Z. Li, X. Chen, and A. Hauptmann. "A Comprehensive Survey of Scene Graphs: Generation and Application". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45.1 (2023), pp. 1–26. DOI: `10.1109/TPAMI.2021.3137605`.

[40] D. Xu, Y. Zhu, C. B. Choy, and L. Fei-Fei. “Scene graph generation by iterative message passing”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 5410–5419.

[41] R. Zellers, M. Yatskar, S. Thomson, and Y. Choi. “Neural motifs: Scene graph parsing with global context”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 5831–5840.

[42] X. Wang and A. Gupta. “Videos as space-time region graphs”. In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 399–417.

[43] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick. *Detectron2*. `https://github.com/facebookresearch/detectron2`. 2019.

[44] K. Tang, Y. Niu, J. Huang, J. Shi, and H. Zhang. “Unbiased scene graph generation from biased training”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 3716–3725.