Towards cognitive automotive environment modelling: reasoning based on vector representations

Florian Mirus[1,3], Terrence C. Stewart[2] and Jörg Conradt[3]

1- BMW Group - Research, New Technologies, Innovations, Garching, Germany
2- University of Waterloo - Centre for Theoretical Neuroscience, Waterloo ON, Canada
3- Technical University of Munich - Neuroscientific System Theory Group, Munich, Germany

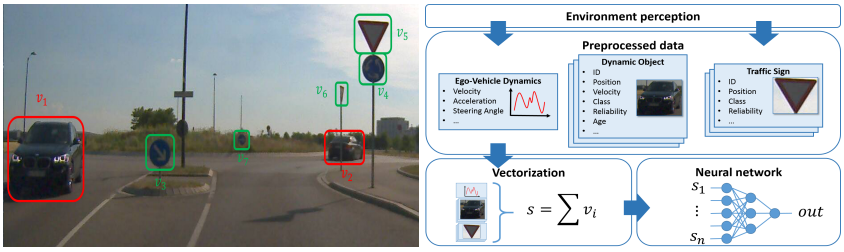
Abstract. In this paper, we propose a novel approach to knowledge representation for automotive environment modelling based on Vector Symbolic Architectures (VSAs). We build a vector representation describing structured information and relations within the current scene based on high-level object-lists perceived by individual sensors. Such a representation can be applied to different tasks with little modifications. In a sample instantiation, we focus on two example tasks, namely driving context classification and simple behavior prediction, to demonstrate the general applicability of our approach. Allowing efficient implementation in Spiking Neural Networks (SNNs), we envision to improve task performance of our approach through online-learning.

1 Introduction

Precise knowledge about the current environment state and its future development is essential for an autonomous agent to plan a secure path for navigation and to safely interact with the world. In case of highly automated vehicles, perception of the outside world usually happens through a variety of different sensory systems like cameras, RADAR and LIDAR sensors [1]. This observed information needs to be collected and combined into a central model of the environment, which is the basis for further reasoning and decisions.

In this paper, we outline a first step in the direction of a cognitive approach to automotive environment modelling based on Vector Symbolic Architectures (VSAs) [2]. We build a vector description of the current scene from high-level object-lists provided by individual sensor units. This rather generic representation can be applied to various different tasks like driving context classification, anomaly detection, behavior analysis and prediction with little modifications to the representation itself. Furthermore, VSAs are suitable as inputs to Spiking Neural Networks (SNNs) [3], which support efficient learning algorithms and future deployment on dedicated neuromorphic hardware. In this paper, we demonstrate the general applicability of our approach on two example tasks, namely driving context classification and behaviour prediction. For the classification task, we present initial results while we only outline our approach for behaviour prediction and postpone an in-depth analysis to future work.

Vector Symbolic Architectures (VSAs) is a term coined by Ross W. Gayler [2] to cover a family of modelling approaches that represent symbols and structures by mapping them to (high-dimensional) vectors. Beside the numerical structure underlying the vectors, the core components of a VSA are a measure of similarity and typically two algebraic operations, namely superposition $\oplus$ and binding $\otimes$, which create a vector similar resp. highly dissimilar to both input vectors. For our purposes, we will



(a) Example scene with labeled object vectors based on environment perception

(b) System architecture

Fig. 1: Schematic system overview with one example scene.

adopt Plate's Holographic Reduced Representations (HRRs) [4]. In this model, basis vectors are picked from the real-valued unit sphere, the angle between vectors (or equivalently the dot product) serves as a measure of similarity, superposition is realized as component-wise vector addition and the binding operation is circular convolution. In our work, basis vectors take the role of atomic ingredients meaning that all structured representations are composed from combinations of basis vectors using the VSA's algebraic operations.

Related Work: Since highly automated vehicles are multi-sensory systems, current automotive world models use mostly probabilistic approaches and Bayesian filtering. The main difference is the level, at which sensory data is combined [5]. Low-level resp. high-level fusion systems combine raw sensor data e.g. in an occupancy-grid [6] or fuse preprocessed object-lists [5] respectively. The work closest to our approach is [7], where they use symbolization, a language modelling technique, to obtain semantic descriptions of driving scenes. For the specific task of recognizing the current driving context, there have been early approaches using statistical pattern recognition based solely on the ego-vehicle's dynamics [8] or on fusion with camera data [9]. The key difference to previous work is that our approach employs cognitive modelling techniques, which allows us to use symbolization as well, but also to combine it with the benefits of neural networks.

2 Implementation

System Architecture: Fig. 1b shows a schematic overview of our system architecture. *Environment perception* happens through a variety of different sensors [1] providing *preprocessed data* in the form of object-lists or raw sensory data. We build our representation from this preprocessed data through *vectorization* of sensory data and high-level object lists. We use a *neural network* depending on the given task for predictions based on the current scene vector. In an example instantiation we use supervised learning to distinguish three different driving contexts (city, interurban, highway).

Input Data: The input data used for training and evaluating the system is real-world data gathered during test drives in the region of Munich, Germany. Depending on the test vehicle's sensor setup [1], a subset of the following sensor systems is available: camera, RADAR, LIDAR as well as the dynamics of the ego-vehicle (e.g. velocity, acceleration, steering angle) through introspective sensors. While lists of dynamic objects

(i.e. cars, pedestrians, etc.) are available from all extrospective sensors, the camera-based perception system additionally provides lists of static objects like traffic signs. In this work, we focus on the ego-vehicle's dynamics and the information provided by the camera-system as the only extrospective sensor. The camera-system is present in all available test traces and furthermore, its data is most informative regarding categories of dynamic objects while being the only system that provides information about traffic signs. Beside the object's classification, the camera systems provides estimations (with variance) of entities like relative position, orientation and velocity for each object. The data is divided into three different sets: one for training and two test sets containing roughly 27 min, 18 min and 7 min respectively of driving data.

Vector Representation: In this work, we encapsulate three types of information in our vector-based scene representation: ego-vehicle dynamics , dynamic objects and traffic signs (provided as preprocessed object-lists). For each category, we describe the process of converting the input data into a vector representation. We obtain the final vector describing the current scene by superposition of all vectors created in each category. For all our vectors in this work we have chosen a dimension of $D = 512$, a reasonable trade-off between informational capacity and computational complexity.

Ego-vehicle dynamics: For ego-vehicle dynamics, we use the current velocity, acceleration in *x*/*y*-direction (ego-vehicle coordinate system), the angle of the steering wheel as well as the steering angle of the front axle. For all values except acceleration, we randomly choose one normalized ID-vector representing the respective value, e.g. $\mathbf{VELOCITY} = (v_1, \cdots, v_D)$ with $v_i \in \mathbb{R}$ for velocity, and multiply this ID-vector with the current scalar value x, e.g. $x \cdot \mathbf{VELOCITY}$ for velocity. Furthermore, we normalize all scalar values to the range $[-2, 2]$ to keep the length of our vectors limited. For vectorization of two-dimensional values, we use an encoding with sine and cosine functions with different spatial frequencies and offsets. Therefore, we define the following helper functions

$$f_{(m,i)} : \mathbb{R}^2 \longrightarrow \mathbb{R}^4, (x,y) \longmapsto \left(\cos \frac{m \cdot \pi + x}{i+1}, \sin \frac{m \cdot \pi + x}{i+1}, \cos \frac{m \cdot \pi + y}{i+1}, \sin \frac{m \cdot \pi + y}{i+1} \right),$$

$$\psi_i : \mathbb{R}^2 \longrightarrow \mathbb{R}^4, (x,y) \longmapsto \left(f_{(0,i)}(x,y), f_{\left(\frac{1}{2},i\right)}(x,y), f_{(1,i)}(x,y), f_{\left(\frac{3}{2},i\right)}(x,y) \right)$$

and obtain the final vector representation of acceleration in *x*/*y*-direction via the function

$$\lambda : \mathbb{R}^2 \longrightarrow \mathbb{R}^D, (x,y) \longmapsto \frac{1}{\sqrt{\frac{D}{2}}} \left(\psi_0(x,y), \cdots, \psi_{\frac{D}{16}-1}(x,y) \right).$$

This encoding $\lambda(a_x, a_y)$ leads to normalized, nonzero, similar vectors with information distributed over all elements (in contrast to a simple encoding like $(a_x, a_y, 0 \cdots, 0)$).

Dynamic objects: The camera-based classification system is able to distinguish seven different object categories, namely bicycle, car, motorcycle, pedestrian, stationary, truck and unknown. In the simplest form of our vector representation, we assign one random vector to each of those categories and add it to the current scene representation once for each category's occurrence in the object-list. However, this representation just encodes that there are certain objects present somewhere in the current scene without any additional information. Enhancing this simple encoding, we use the function λ to map each

dynamic object's position in x/y-direction (relative to the ego-vehicle) to vector form and bind the result to the vector representing the object's category. One quite unique feature of e.g. highway driving is the fact that almost all other traffic participants drive in similar direction as the ego-vehicle. Therefore, we create additional random vectors encoding the orientation of dynamic objects relative to the ego-vehicle for three discretized categories, namely **SAME**, **OPPOSITE** and **LATERAL**. If we want to jointly bind those two pieces of information to one object, we need to introduce two additional ID-vectors **POSITION** and **ORIENTATION** to impose structure. For example, a car detected at position (p_x, p_y) with approximately the same orientation as the ego-vehicle would lead to the following vector representation

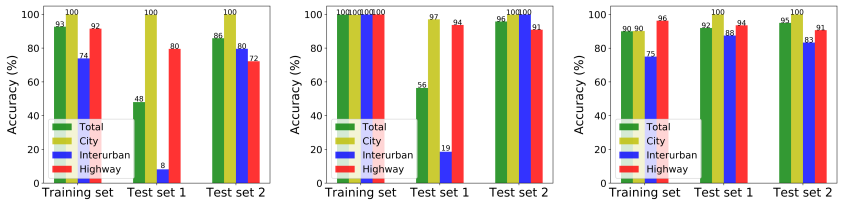
$$\mathbf{CAR} + \mathbf{CAR} \otimes \mathbf{POSITION} \otimes \lambda\,(p_x, p_y) + \mathbf{CAR} \otimes \mathbf{ORIENTATION} \otimes \mathbf{SAME}.$$

Traffic signs: The ego-vehicle's camera-system [1] is able to recognize a significant amount and variety of traffic signs. Again, we assign a random vector to each possible traffic sign label and add it to the current scene representation. However, in contrast to dynamic objects, most traffic signs are not only valid while being visible but stay relevant for the current driving context until withdrawn by another sign. Therefore, we implemented a simple form of memory for a certain subset of traffic signs relevant to the task of driving context classification even after disappearing. Due to the fact, that the camera system is not immune to false detections, we implemented a decaying memory, to avoid relying too much on false detections and to allow the system to consider other cues. Furthermore, we included a simple withdraw logic, e.g. new speed limit signs overwrites previously seen ones and a sign indicating a city entrance withdraws a memorized highway sign.

Training: To enable automated training of any supervised learning system, the training data needs to be labeled. In this work, we hand-labeled our data sets by visually inspecting the images provided by a reference camera-system and labelling the intervals between transitions of driving contexts as indicated by the respective traffic signs.

For actual training of our system, we used the software suite Neural Engineering Objects (Nengo) [10] for neural simulations, an implementation of the Neural Engineering Framework (NEF) [11]. This allows rapid training and testing while at the same time setting up our system in the framework of neural computation. However, we also trained a network using the Keras library [12] for reference. The neural ensemble trained in the Nengo simulator consists of 1500 Leaky-Integrate-and-Fire (LIF) spiking neurons, while we use a simple two-layer network consisting of 1500 and 500 neurons per layer in Keras.

Behaviour Prediction: In a second example task, we use our vector representation as input data for another neural network to predict the future position of one other traffic participant at a time. We use an additional indication vector **THIS** bound to the object we want to predict to tell the network the current focus. To predict all objects of the scene during deployment, we envision to use multiple instantiations of the same network. Thereby, the amount of training data generated per file increases with the number of objects while we only need to train one network. We expect that our structured vector representation will be able to capture relations and mutual influence between traffic participants necessary for reliable prediction when combined with memory.



(a) Nengo network (b) Keras network (c) Human performance

Fig. 2: Comparison of context classification results against human level performance.

3 Results

Human Level Performance: To get a better understanding of the quality of the context classification system's results, we compare it to human level performance. However, neither showing raw camera images nor numerical vectors would yield comparable results. Therefore, we created human-readable versions of our input vectors in text form and presented a subset of 50 random samples for each data set (with the training set always being the first) to two human subjects asking for their classification guess. The accumulated results are shown in Fig. 2c.

Classification Performance: In our experiments, we found that the vector representation with decaying memory for traffic signs as well as structured information (position/orientation of dynamic objects) works best for driving context classification (cf. Fig. 2a, 2b). Given the limited amount of training data, the system is able to reliably distinguish the city and highway categories, even in the absence of traffic signs indicating the start of a highway as well as examples of the interurban category similar to those in the training set (Test set 2). By introducing a more sophisticated network structure and learning algorithm in Keras, we were able to improve performance to a level comparable with human level performance on Test set 2 (cf. Fig. 2b, 2c) and superior to previous work [8]. The interurban category, however, is problematic being the main factor for our system to deliver poor classification results on Test set 1 (cf. Fig. 2). There are two reasons: firstly, the largest part of Test Set 1 labelled as interurban driving is heavy stop-and-go traffic, which is dissimilar to all interurban examples in the training set, and secondly the camera-system misses to detect a traffic sign indicating a city exit.

4 Discussion

Conclusion: We presented a novel approach to knowledge representation for automotive environment modelling and demonstrated applicability on two example tasks. We showed initial results on the task of driving context classification compared to human performance and related work, while we only outlined our approach to behaviour prediction. Given the limited data, we expect our system to improve significantly with increasing number and diversity of training examples. However, we consider the general approach of using cognitive modelling techniques in automotive context the main contribution of this work. With highly automated driving on the horizon, machine learning in general becomes increasingly important in the automotive domain. Our approach

offers a unified representation, which can be the basis for different learning algorithms. Finally, our vector representation allows implementation in the framework of SNNs using the principles of the NEF. This framework allows us to apply (online) learning and to deploy our system on energy-efficient neuromorphic hardware, which could be a promising addition in automotive context, where energy-efficiency is essential. Therefore, we believe that our work is a promising first step in the direction of a cognitive automotive environment model.

Future Work: Although our results are promising, there are several options for future enhancements. Instead of choosing basis vectors at random with no inherent structure or similarity, we envision to build a basis vector vocabulary through a learning system that is able to encapsulate different levels of similarity (e.g. visual, contextual, semantic). We also aim to further investigate our approach on the task of predicting the behaviour of other traffic participants around the ego-vehicle. Furthermore, we intend to apply the developed representation to other driving-related tasks like prediction of traffic signs expected in the current driving context or anomaly detection (e.g. discrepancy between expected and perceived traffic signs).

References

[1] M. Aeberhard, S. Rauch, M. Bahram, G. Tanzmeister, J. Thomas, Y. Pilat, F. Homm, W. Huber, and N. Kaempchen. Experience, Results and Lessons Learned from Automated Driving on Germany's Highways. *IEEE Intelligent Transportation Systems Magazine*, 7(1):42–57, Spring 2015.

[2] Ross Gayler. Vector Symbolic Architectures answer Jackendoff's challenges for cognitive neuroscience. In Peter Slezak, editor, *ICCS/ASCS International Conference on Cognitive Science*, pages 133–138. University of New South Wales, CogPrints, 2003.

[3] Chris Eliasmith. *How to build a brain: A neural architecture for biological cognition*. Oxford University Press, New York, NY, 2013.

[4] Tony A. Plate. *Holographic Reduced Representation: Distributed Representation for Cognitive Structures*. CSLI Publications, Stanford, CA, USA, 2003.

[5] Jos Elfring, Rein Appeldoorn, Sjoerd van den Dries, and Maurice Kwakkernaat. Effective World Modeling: Multisensor Data Fusion Methodology for Automated Driving. *Sensors*, 16(10):1668, 2016.

[6] Georg Tanzmeister, Julian Thomas, Dirk Wollherr, and Martin Buss. Grid-based mapping and tracking in dynamic environments using a uniform evidential environment representation. In *2014 IEEE International Conference on Robotics and Automation, ICRA 2014, Hong Kong, China, May 31 - June 7, 2014*, pages 6090–6095, 2014.

[7] S. Yamazaki, C. Miyajima, E. Yurtsever, K. Takeda, M. Mori, K. Hitomi, and M. Egawa. Integrating driving behavior and traffic context through signal symbolization. In *2016 IEEE Intelligent Vehicles Symposium (IV)*, pages 642–647, June 2016.

[8] J. Engstrom and T. Victor. Real-time recognition of large-scale driving patterns. In *ITSC 2001. 2001 IEEE Intelligent Transportation Systems. Proceedings (Cat. No.01TH8585)*, pages 1018–1023, 2001.

[9] W. Hauptmann, F. Graf, and K. Heesche. Driving environment recognition for adaptive automotive systems. In *Proceedings of IEEE 5th International Fuzzy Systems*, volume 1, pages 387–393 vol.1, Sep 1996.

[10] Trevor Bekolay, James Bergstra, Eric Hunsberger, Travis DeWolf, Terrence C Stewart, Daniel Rasmussen, Xuan Choo, Aaron Voelker, and Chris Eliasmith. Nengo: A Python tool for building large-scale functional brain models. *Frontiers in Neuroinformatics*, 7(48), 2014.

[11] Chris Eliasmith and Charles H. Anderson. *Neural Engineering : Computation, Representation, and Dynamics in Neurobiological Systems*. Computational neuroscience. Cambridge, Mass. MIT Press, 2003.

[12] François Chollet. Keras. `https://github.com/keras-team/keras`, 2015.