A very first Glance on the Safety Analysis of Self-learning Algorithms for Autonomous Cars

Tim Gonschorek, Marco Filax, Frank Ortmeier

A very first Glance on the Safety Analysis of Self-learning Algorithms for Autonomous Cars

(Fast Abstract)

Tim Gonschorek, Marco Filax, and Frank Ortmeier,
Chair of Software Engineering, Faculty of Computer Science, Otto-von-Guericke-University, Magdeburg, Germany
{tim.gonschorek, frank.ortmeier, marco.filax}@ovgu.de

I. Introduction

In this fast abstract, we aim at presenting our first ideas on the verification of autonomous self-driving cars with, in particular, the Artificial Intelligence (AI) algorithm parts and corresponding implementations. We derive our ideas with a special consideration of the system development to given standards and their assessability.

We focus, therefore, on learning-based algorithms enabling an autonomous car to percept its environment, i.e., learning-based algorithms for the visual perception. This mainly contains Artificial Neural Networks [1] with different manifestations, e.g., Generative Adversarial Nets [2] or Convolutional Neural Networks (CNN) [3].

Their analysis is in particular essential for the certification of future autonomous cars since they define the perceptive machine interface and therefore generate relevant input for most following control and steering algorithms. As a consequence, these perception AI-algorithms become safety-relevant functions. Especially when dealing with autonomous systems of autonomy level three and above, the criticality of those components gets ASIL-D, the highest criticality level implying several specific requirements for the system development process.

Fig. 1 shows the schema of a Convolutional Neural Network, one of the currently most important algorithms for visual perception. The basic idea is to define a set of classes, e.g., cars, trucks, or pedestrians, and training data with several pictures, containing instances of those classes. Based on the training data, the CNN "learns" to classify an input image correlating to its content (e.g., car or pedestrian).

In a first step, several convolution blocks, each representing a set of image filters, are applied to the previous block's output. After the set of convolutions, each entry of the last convolution block is mapped to the input vector of the fully connected layer, i.e., the neural net for the classification. The final output of the neural network is a specificator function containing correspondence values for the input to each class (label). This is a value between zero and one whereas the sum of all values (one per label) is one. The specific weights and threshold are learned by minimizing the error of the value in the specificator function for the respective image $\leftrightarrow$ classification tuple.

When implementing a CNN, the basic architecture is fixed. Specific parameters and design decision, however, can be made. This includes, e.g., the convolutions and filters as well as their order, the number and granularity of the classes, or the training data and its distribution.

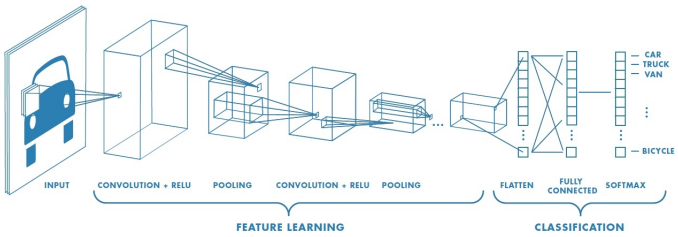

Fig. 1. Example of a Convolutional Neural Network [4].

II. On Ensuring Safety for AI-algorithms

In our point of view, a first step for analyzing such learning-based algorithms would be taken by implementing safety guarantee mechanisms and measures in the following three abstract system (design) layers: *Algorithmic Layer*, *Software/Implementation Layer*, and *Validation Layer*. These different layers correspond to different abstract stages in the system development process.

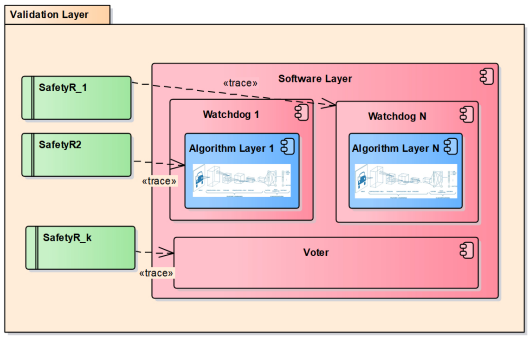

Fig. 2. System and design layers of interest for the analysis.

The *Algorithm Layer* focus on the selection of the algorithm, it's training, and the interpretation of the classification results. In the *Software/Implementation Layer* the AI-algorithm is integrated within the remaining software infrastructure, its results are verified for their plausibility and then compared to those of other algorithms. In contrast to that, the *Validation Layer* focus on the traceability of the safety requirements to the design decisions and implementations.

A. Validation Layer

In the validation layer, one major task would be the definition of new traceability associations from system and safety requirements to architecture elements and corresponding code. The important part here is that we neither want to develop the architecture nor implement the neural network, but use a specific architecture and implementation. However, we have to choose the, in most cases predefined, AI-algorithm architecture, defining the corresponding distribution for the input function and training data, specify a loss function, i.e., a quantification to map the prediction values to specific classes as well as corresponding guarantees. So, we must focus on arguing the design decision with traceable associations back to the initial system or safety requirements (*cf.* [5]).

An example of an important design decision is given in Fig. 3. This is the specificator function, mapping the values of the last layer of the neural net to the predefined classes. Here we can see, that in many cases there exists a level of uncertainty in the decision since often we do not have a 100% decision. An essential step in the validation layer would therefore be to argue about the level of guarantee required and how to achieve this, e.g., in combination with other algorithms and distinct training data for different algorithms.

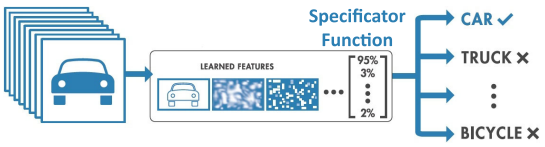

Fig. 3. Classification of an image with corresponding specificator funtion [4].

B. Implementation Layer

For the safety assessment, we think especially two parts are important in the implementation layer: the application of several independent algorithms (e.g., differently trained and different architecture) and the comprehensibility of the algorithm as an understandable plausibility check.

As given in the previous section, it is possible that the perception algorithms has a specific level of uncertainty. Therefore, it is essential to define some voting process weighting the outputs of the networks and decide whether the reliability of the computed output is high enough. Taking into account the single specificator function values, we could derive specific weights in combination with a so-called ensemble voting approach, where results from different networks, trained with different data or implementing different architectures are included in the classification process [6].

Another important fact is that, even though we understand the general behavior of a training-based algorithm, it is hard to validate the specific learned values. Especially since images can be manipulated with noise that is not visible to the human but can totally mislead the neural network [7]. Therefore, we propose to use parallel, dynamically assembling watchdog components implementing a behavior redundant to the learning-based algorithm but in a comprehensible way. It is not necessary that the redundant component is as good as the neural network, but it should be able to identify completely outlying computed values and make the decision procedure of the net more comprehensible to a human engineer.

C. Algorithmic Layer

Even though we do not implement the basic network, we must choose several parameters for its implementation.

It is, for example, quite significant in which order we apply with convolution and how are the input-output constraints for each convolution layer. From this, we can abstract particular formulae specifying, e.g., input requirements and output assurances (comparable to software code contract). By encoding these contracts into a SAT problem, it should be possible to verify whether these convolutions can be applied together in the specific order and combination as well as whether they fit the overall score function.

Another critical point is the definition of classification sets and the corresponding training data since it has an impact on the neural net if, e.g., the sets of images are sufficiently different in their specific features and the classes have sharp distinctions based on recognizable image features. Therefore, specific feature distributions must be defined stating the training images are sufficiently specific for the recognition [8] but also cover all important possible external environmental influences (e.g., rain, cloudy weather, or sunshine).

III. Conclusion

Summarizing, we think that there are several stages on the different layers for ensuring safety for learning-based artificial intelligent algorithms within autonomous driving cars. This does not necessarily correspond to formal verification of the implemented algorithm itself, but also to providing a traceable and comprehensible system description and behavior which can be assessed by an external expert or increasing the reliability of the output by diversifying the decision strategies.

References

[1] R. J. Schalkoff, *Artificial neural networks*. McGraw-Hill New York, 1997.

[2] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014.

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012.

[4] [Online]. Available: https://de.mathworks.com/solutions/deep-learning/convolutional-neural-network.html

[5] M. Filax, T. Gonschorek, and F. Ortmeier, "Building models we can rely on: Requirements traceability for model-based verification techniques," in *Proceedings of IMBSA 2017*, 2017.

[6] M. Woźniak, M. G. na, and E. Corchado, "A survey of multiple classifier systems as hybrid systems," *Information Fusion*, 2014.

[7] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," *CoRR*, 2013. [Online]. Available: http://arxiv.org/abs/1312.6199

[8] K. Grosse, P. Manoharan, N. Papernot, M. Backes, and P. D. McDaniel, "On the (statistical) detection of adversarial examples," *CoRR*, vol. abs/1702.06280, 2017. [Online]. Available: http://arxiv.org/abs/1702.06280