SCENARIO2VEC: A SCENARIO DESCRIPTION LANGUAGE TO CHARACTERIZE TRAFFIC SCENARIOS FOR THE DEVELOPMENT OF A CERTIFICATION SCHEME

A Technical Report submitted to the Department of Computer Science

Presented to the Faculty of the School of Engineering and Applied Science
University of Virginia • Charlottesville, Virginia

In Partial Fulfillment of the Requirements of the Degree
Bachelor of Science, School of Engineering

Jaspreet Ranjit
Summer, 2020

Technical Project Team Members
Aron Harder

On my honor as a University Student, I have neither given nor received unauthorized aid on this assignment as defined by the Honor Guidelines for Thesis-Related Assignments

Signature Jaspreet Ranjit Date 08/06/20
Jaspreet Ranjit

ADVISOR
Madhur Behl, Department of Computer Science

Scenario2Vec: Scenario Description Language to Characterize Traffic Scenarios

Jaspreet Ranjit, Aron Harder and Madhur Behl
Computer Science
University of Virginia

1 Introduction

Autonomous driving systems have machine learning components, such as deep neural networks, for which formal properties are difficult to characterize. It is difficult, and maybe even impossible, to characterize all of the behaviors of these components under all circumstances.

Due to the rarity of failure events, real-world test driving alone cannot provide high confidence in the safety of automated driving systems with respect to injuries and fatalities [1, 2] This leads to a challenging issue today for automated vehicle manufacturers and suppliers who are determined to incorporate machine learning for automated driving. While the principle of safety by design (verification) is useful, it remains insufficient for automated driving systems, because of the existence of unknown scenarios that cannot be directly designed for, or verified. The aim of this research thrust is to propose a new innovative certification scheme allowing to demonstrate the level of safety and reliability which allows for safe market introduction of automated/autonomous vehicles. Our goal is to answer the following questions:
Q1: How can we fairly compare two different AV software stacks on a given safety metric?
Q2: How can we leverage simulation to find edge cases and failures for a given AV system?

The kind of closed-loop verification, likely to be required for AV component testing, is beyond the reach of traditional test methodologies and discrete verification. There will remain some small risk of crashes. The concept of residual risk has already been accepted for a long time now (see the rollout of airbags or new medicines). Validation puts the verified system to the test in scenarios or situations that the system would likely encounter in everyday driving after its release. These scenarios can either be controlled directly in a physical (closed-course proving ground) or virtual (simulation of pre-defined scenarios) environment, or they can arise spontaneously during operation in the real world (open-road testing or simulation of randomly

generated scenarios).

1.1 Current safety standards for AVs

The meaning of safety in regard to AVs is surprisingly unclear—and no standard definition exists. The regulators rely on automotive companies to present a view of safety, while the companies themselves, each having a different interpretation of what constitutes as safe driving behavior, in turn seek input from the regulators. The majority of safety assessment today is self-reported by the testing companies, in good faith [3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16]. These companies develop different interpretations of what constitutes safe driving behavior. Autonomous miles driven and miles per disengagements/intervention are two metrics closely watched by industry observers to provide a high-level view of AV safety. Interventions happen when either a safety operator detects bad behavior and takes control of an automated vehicle, or the vehicle itself detects something wrong and calls for a human to take over. Low rates of intervention do not necessarily indicate higher safety, they indicate only high agreement between drivers and automated systems. Humans can sometime fail to detect hazards and if the automated vehicle fails too, they will agree without being safe. Therefore, disengagement is only an appropriate safety metric if the goal is to make AVs as safe (but not safer) than human drivers. It is no secret that safety assessments for automated vehicles need to evolve beyond the existing voluntary self-reporting. The hurdle is that no comprehensive common measuring stick to compare how far along each AV developer is in terms of safety exists today.

2 Problem Statement

The first step in designing a unified safety certification scheme is to develop a unified representation of a traffic scenario. A standard dictionary to define a traffic scenario will provide a high-level representation of the multi-agent interaction in a given video. This description would serve as a label for a video that captures information about the scenario. Compared to a low level vector embedding, a high level scenario description for a video preserves human readable information about a traffic scenario. These descriptions will be used as labels for video clips to train a network that can generate a scenario descriptor for an unseen video. The ability to automatically label a traffic scenario using a standardized language can help identify whether or not a vehicle is complying by road safety standards. For example, if a

pedestrian is crossing the street, it will be important for the AV to capture this information in a standardized way and take the appropriate course of action to ensure the safety of the pedestrian. This action will require quick labeling of the traffic scenario provided by the scenario descriptor. Another important end goal of this research will be the task of similar video retrieval using a reference video.

AV manufacturers have big datasets of videos with traffic scenarios making it difficult to find similar scenarios in two different datasets. In order to provide a fair, and equitable comparison of two AV designs, it will be necessary to use a reference traffic scenario to query datasets for similar scenarios. Similar traffic situations can then be compared on the basis of a standardized certification scheme to identify where one AV design outperforms another. In order to accomplish these goals, this paper will focus on developing a scenario description language that extracts a high level description from a captioned dataset with traffic scenarios.

3 Related Work

Prior research has focused on generating a video embedding that is composed by extracting features of videos from successive frames with the ultimate goal of being able to compare two videos based on temporal information. Each frame is represented as a matrix embedding containing spatial information about the scene. These frames are stacked on top of each other to form a video matrix that is transformed into a vector. As a result, the video vector represents information about the video restricted to a frame level analysis. A convolutional neural network is trained using video clips to generate these video vectors. The video vectors can then be used for tasks such as video classification. The code that performs this tasks is in the following github repository. https://github.com/fengyoung/video_embedding. One of the biggest limitations of this approach is its inability to encode temporal information about the video. This limits the network's ability to differentiate between similar traffic scenarios. For example, videos that contained similar temporal content but were different spatially were still recognized by the network as different.

This method of generating video embeddings was further investigated by generating video vectors for a small subset of the moments in time dataset. [17]. The goal of the experiment was to investigate whether the video embeddings can be used to query an unseen database of videos to find a similar video based on a reference video. The moments in time dataset is very di-

verse and for some videos that are classified in a certain category, it would be difficult, even for a human, to recognize that it belonged to that category. Since the vector embedding only encodes pixel wise information, two videos that are similar temporally but different spatially would still be classified as different. The goal of the experiment was to test and verify this hypothesis. There were four categories: Driving, Bowling, Handwriting, and Walking. There were 24 videos in each category for the train set, and 10 videos from each category in the test set. The videos were each 3 seconds long, and for each video, a vector embedding was generated and a norm value was extracted. As a result, each vector was represented as a single number. The norms from each category of videos in the train set were averaged to get one number representative of the entire category. For each video in the test set, a vector was generated along with a norm value. This number was then used to predict which category the video belonged; the video was placed in the category that was closest to its norm value. As predicted, this approach failed in instances where the videos were different spatially. It also failed to predict two temporally similar videos in the same category. The results are shown in Figure INSERT FIGURE where the train set column represents the vector norm values that were used to represent each category. The categories from the test set were initially unlabeled and were assigned to one of the 4 categories based soley on their video embedding vector norm. A video was placed in a category if it was within +/- 0.01 of the category norm. The test set column shows how many videos were correctly identified. Furthermore, the norms generated for each category were a very small distance apart from each other making it difficult to differentiate between categories.

Moments in Time Experiment

Category	Train Set (Vector norms)	Test Set (Percentage correctly identified)
Handwriting	39.6937	10%
Driving	38.8466	10%
Bowling	42.9000	20%
Walking	42.7116	0%

Figure 1: Results from the moment in time experiment. The train set column shows the vector embedding norms for each category. These numbers were used as identifiers for each category. For each of the 10 videos in the test set, a vector norm was extracted and used to place a video in a category closest to its norm. However, this experiment showed that temporally different videos could not be classified properly

As shown in the experiment, encoding only spatial information in an embedding does not serve as a powerful representation of the contents of the video. As a result, a higher level embedding would be required to capture both spatial and temporal aspects of a scenario. These descriptions would need to account for multi-agent interaction in the video and extract information from a scenario that would be useful in differentiating two traffic scenarios. For example, Open Autonomous Safety outlines scenarios that an AV could encounter and defines a detailed scenario description language to capture the behavioral requirements that must be followed by an AV in order to maintain the highest standard of safety at all times [18]. This scenario description language outlines road segments, number of lanes, stop signs, ego actions, other actors, and start and end positions along with scene elements such as intersections, pedestrians, and speed limits. The definition of an SDL enables the development of a comprehensive list of different scenarios to define the various situations an AV might encounter. This language can be used to quickly parametrize a traffic scenario an AV might encounter and evaluate whether the AV is following safety standards. Similarly, M-SDL is an open source, human readable high level language that captures

information about a scenario [19]. This allows for easy reuse and sharing of scenarios between companies to compare two AVs on similar, standardized data. However, both Open Autonomous Safety and M-SDL require manual labeling which can be a time consuming, error prone process.

4 Methodology

The primary motivation of this research project is to develop a scenario description language that can be parsed using textual caption data. Using the Berkeley Deep Drive-X Dataset which contains textual descriptions and explanations for dashboard camera videos, a standardized scenario descriptor was parsed and extracted from the accompanying textual description. However, the first step in extracting the scenario descriptor was developing a set of descriptors that could be extracted from the given dataset. After performing data analysis, it was found that the most reasonable information to encode were the actors, their associated actions, and scene elements. Figure 2 provides a detailed overview of the actors, actions and scenes present in the BDDX data.

Scenario Description Language

Actor	Ego, light vehicle, heavy vehicle, cyclist, pedestrian, traffic
Action	Turn, turn left, turn right, merge, accelerate, brake, stop, forward, walk, park, drive, reverse, merge center, merge left, merge, right, turn through, merge u-turn, u-turn
Scene	Intersection, crosswalk, bridge, green light, stop sign, yield sign, sign, u-turn, traffic light, traffic signal, turn lane, crosswalks, green traffic light, light, red light, red traffic light, signs, traffic lights, yellow light, yellow traffic light

Figure 2: Characterization of scenario description language with the specific actors, actions and scenes that were extracted from the BDDX data.

Given this dictionary of descriptors, the following step used the captions to parse an SDL string for each video. This string encoded high-level information about the actors and their associated actions as well as scene elements. To provide a more robust representation that could be used for further evaluation and comparison, the string structure was converted to an

actor-action matrix and a scene matrix. The actor-action matrix is a [2 x n] matrix where n represents the number of actor-action pairs in each SDL object. A single SDL object has at least one or more actor-action pairs given that each video has an ego actor. The scene elements are encoded in a list where each number is mapped to a scene object. These matrices are illustrated in Figure 3. Before training a network with SDLs as labels for videos to automatically tag unseen traffic scenarios with an SDL, it is imperative to show similarity in the SDL space. To illustrate similarity in the SDL space, a similarity metric was constructed to compare two SDLs. The similarity metric was developed on the assumption that the scenario follows a hierarchy with actors being the most important measure of similarity, then actions, followed by the scene. More specifically, given a reference SDL, similar SDLs are found by first comparing actors to ensure all actors are present in a given SDL, then actions, and then the scene elements. Sent2Vec was used as the baseline approach for comparing how well similarity can be characterized in the SDL space.

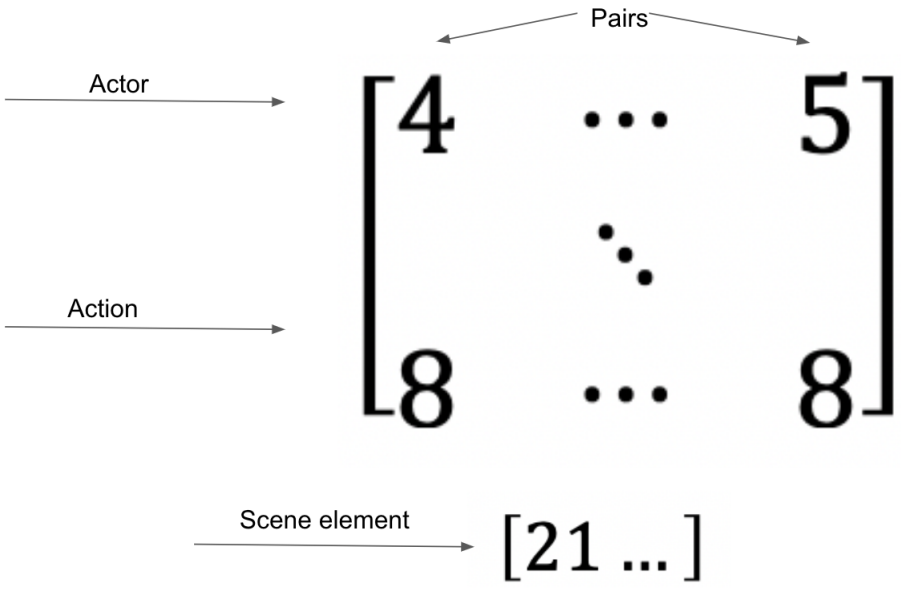

Figure 3: SDL matrix embedding. The first matrix is the actor-action matrix where each column represents an actor-action pair. The second vector is a list of scene elements. Each number is mapped to a specfic SDL element as detailed in Figure 2.

5 Preliminary Results

To evaluate how well similarity can be classified in the SDL space, a similarity metric was used to extract similar SDLs given a reference caption from the BDDX data. The same reference caption was put through a Sent2Vec model to extract similar sentences. The results from both the SDL model and the Sent2Vec model were compared for similarities and differences to investigate which one performed better in capturing similar videos. Both the Sent2Vec and the SDL Methods have the pairwise distance computed between all samples. For the Sent2Vec method, the Euclidean distance between the vectors was used as the distance metric. For the SDL method, a hierarchical method was used, wherein the samples that contained the most-similar actors were found. These results were then further narrowed down to the samples containing the most-similar actions, and then only the samples containing the most-similar scene elements. Using this hierarchical method, the nearest neighbor was found for each sample in the SDL. If there were multiple samples that were equally close, all such samples were returned as being nearest. For each sample in the Sent2Vec, the nearest neighbors were found until the number of neighbors matched that of SDL. If there were multiple samples that were equally close such that adding them all would cause Sent2Vec to have more samples than SDL, those samples were ignored and Sent2Vec simply had fewer samples than SDL. Once the n nearest neighbors for both of our methods had been computed, the f1 score was computed, any sample that was in both datasets was considered true positive, any sample that was in SDL but not Sent2Vec was considered false positive, any sample that was in Sent2Vec but not SDL was considered false negative. The average f1 score (used to measure a test's accuracy) across all samples was 0.246.

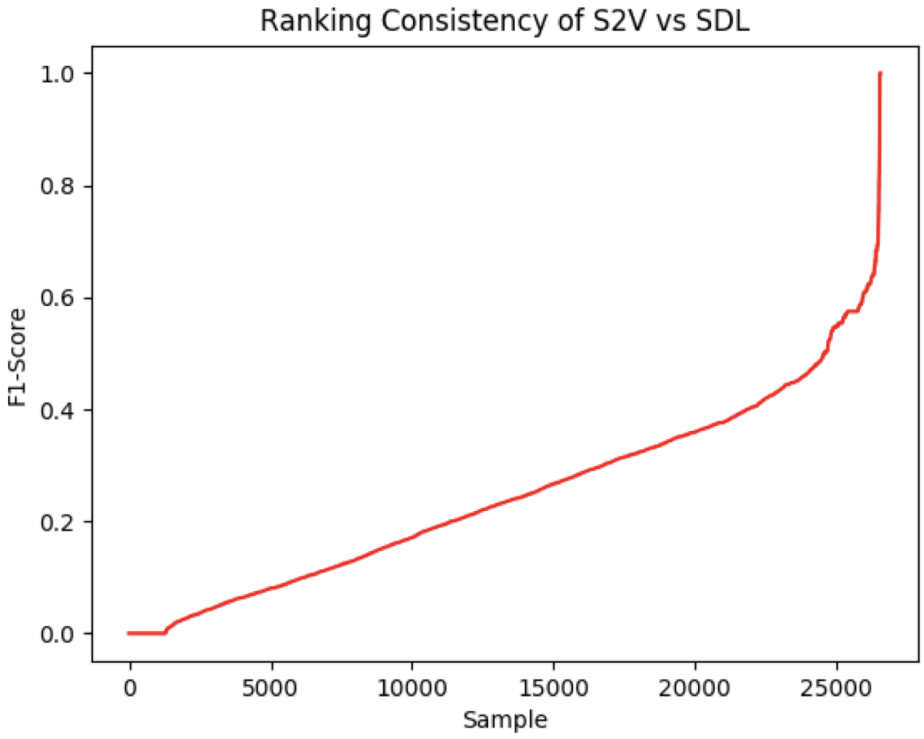

Figure 4: F1 score from Sent2Vec and SDL experiment. The F1 score is 0.246.

For a given caption: "The car slows slightly because cars ahead are making a turn", the SDL parser returned: "ego-brake, light vehicle-turn, no scene element". The video frames are shown in Figure 5.

Figure 5: Frames for a reference video.

The SDL similarity metric and Sent2Vec each returned a list of "similar" objects. These lists were compared for similarities and differences and are shown in Figure 6.

Similar Results Returned by Sent2Vec and SDL

#	Sent2Vec	SDL
1	The car slows down because the car ahead is slowing to make a right turn.	The car is slowing down because there is a slow-moving car in the intersection trying to turn right.
2	The car slows down because the car in front is making a turn.	The car slows to avoid hitting a car that's turning in its lane.
3	The car slows slightly because it's making a right turn	The car decelerates because there is a stop sign and a car turning into the road ahead.

Figure 6: Results from SDL and Sent2Vec similarity experiment. Each method returned a list of most "similar" captions based on its similarity metric.

The Sent2Vec results all involve a car making a turn. The SDL results all involve a car other than the ego a turn. The Sent2Vec results all contain the same clause, "The car slows". The SDL results ignore having the same exact phrasing in favor of having the same actor/action pairs. Overall, both are returning results that are similar to the base sentence. However, since the F1 score is .268, this indicates that further research will need to be done in investigating where Sent2Vec and the SDL method fail. Furthermore, this F1 score shows that Sent2Vec may not be the best baseline score to compare to and measure the validity of the SDL.

6 Conclusions and Future Work

Future research will involve further investigation into designing a similarity metric to compare similarity in the SDL space. This will also involve investigating and comparing how well the current similarity scheme compares to Sent2Vec. Further investigation into where one method fails over another will also be required to evaluate the validity of the SDL similarity scheme. More specifically, future efforts will focus on capturing the temporal structures of traffic scenarios to form fixed-length vector representation. Using the SDLs as labels for these video vector representations, a network will be trained to automatically tag a traffic scenario based on its spatial and temporal features. The testing phase will test the certification scheme's ability to compare two AV designs based on the level of safety they provide.

References

[1] Walther Hans Karl Wachenfeld. *How stochastic can help to introduce automated driving*. PhD thesis, Technische Universität Darmstadt, 2017.

[2] Rick Salay and Krzysztof Czarnecki. Using machine learning safely in automotive software: An assessment and adaption of software process requirements in iso 26262. *arXiv preprint arXiv:1808.01614*, 2018.

[3] Apple Inc. Our approach to automated driving system safety. *Apple ADS Safety*. https://www.apple.com/ads/ADS-Safety.pdf.

[4] Aurora safety report: The new era of mobility. *Aurora*. https://aurora.tech/vssa/index.html.

[5] The autox safety factor. *autox*. https://autox.ai/safety.html.

[6] A matter of trust : Ford's approach to developing self driving vehicles. *ford*. https://media.ford.com/content/dam/fordmedia/pdf/Ford_AV_LLC_FINAL_HR_2.pdf.

[7] 2018 self driving safety report. *General Motors*. https://www.gm.com/our-stories/self-driving-cars.html.

[8] Reinventing safety: a joint approach to automated driving systems. *Daimler and Bosch*. https://www.daimler.com/innovation/case/autonomous/reinventing-safety-2.html.

[9] Navya safety report the autonom era. *Navya*. https://navya.tech/safety-report/.

[10] Delivering safety: Nuro's approach. *Nuro*.

[11] Safety is what drives us: Introducing the nvidia self-driving safety report. *Nvidia*, Oct 2018. https://blogs.nvidia.com/blog/2018/10/23/introducing-self-driving-safety-report/.

[12] Starsky robotics: Voluntary safety self-assessment. *Starsky Robotics*.

[13] 2019 self-driving safety report. https://www.tusimple.com/wp-content/uploads/2019/05/TuSimple-2019-Self-Driving-Safety-Report.pdf.

[14] Uber advanced technologies group a principled approach to safety. *Uber*, 2018. https://uber.app.box.com/v/UberATGSafetyReport.

[15] Safety report and first responders waymo safety report. *Waymo*. https://waymo.com/safety/.

[16] Safety is foundational to our mission. *Zoox*.

[17] Jinkyu Kim, Anna Rohrbach, Trevor Darrell, John Canny, and Zeynep Akata. Textual explanations for self-driving vehicles. *Lecture Notes in Computer Science*, page 577–593, 2018.

[18] Open Autonomous Safety. Oas scenarios, 2018.

[19] Foretellix. Open m-sdl, 2019.