Scenario2Vector: Scenario Description Language Based Embeddings for Traffic Situations

Aron Harder*
ah2ph@virginia.edu
Department of Computer Science,
University of Virginia

Jaspreet Ranjit*
jr4fs@virginia.edu
Department of Computer Science,
University of Virginia

Madhur Behl
madhur.behl@virginia.edu
Department of Computer Science,
University of Virginia

ABSTRACT

A popular metric for measuring progress in autonomous driving has been the "miles per intervention". This is nowhere near a sufficient metric and it does not allow for a fair comparison between the capabilities of two autonomous vehicles (AVs). In this paper we propose Scenario2Vector - a Scenario Description Language (SDL) based embedding for traffic situations that allows us to automatically search for similar traffic situations from large AV data-sets. Our SDL embedding distills a traffic situation experienced by an AV into its canonical components - actors, actions, and the traffic scene. We can then use this embedding to evaluate similarity of different traffic situations in vector space. We have also created a first of its kind, Traffic Scenario Similarity (TSS) dataset which contains human ranking annotations for the similarity between traffic scenarios. Using the TSS data, we compare our SDL embedding - with textual caption based search methods such as Sentence2Vector. We find that Scenario2Vector outperforms Sentence2Vector by 13% ; and is a promising step towards enabling fair comparisons among AVs by inspecting how they perform in similar traffic situations. We hope that Scenario2Vector can have a similar impact to the AV community that Word2Vec/Sent2Vec have had in Natural Language Processing datasets.

ACM Reference Format:
Aron Harder, Jaspreet Ranjit, and Madhur Behl. 2021. Scenario2Vector: Scenario Description Language Based Embeddings for Traffic Situations. In *ACM/IEEE 12th International Conference on Cyber-Physical Systems (with CPS-IoT Week 2021) (ICCPS '21), May 19–21, 2021, Nashville, TN, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3450267.3450544

1 INTRODUCTION

A subtle but influential enemy is jamming the wheels of autonomous vehicles (AVs) - public distrust. According to a May 2020 [1] survey, nearly three in four Americans say that autonomous vehicle technology is not yet ready for prime-time. About half of the respondents said they would never get inside a vehicle that is being driven autonomously. Fatal self-driving-car accidents [2, 3] have cast further doubt in the general public on whether AVs can become the future of mobility. Autonomous vehicles are an example of autonomous Cyber-Physical Systems (CPS) - in that they combine the physics of motion with advanced perception, planning, and control (cyber) algorithms to act on their own without (or with limited) close human supervision. The present approach towards developing such automated systems is that reasonable levels of autonomy can be reached with the help of advances in artificial intelligence, formal methods, and control algorithms that can cope with the uncertainties of the real world, and provide assurances.

Due to the rarity of unexpected situations, real-world testing cannot provide high confidence in the safety of automated driving systems [4, 5]. This leads to a challenging issue today for automated vehicle manufacturers and suppliers who are determined to incorporate machine learning for automated driving. It is difficult, and maybe impossible, to characterize all of the behaviors of these components under all circumstances. While the principle of safety by design (verification) is useful, it remains insufficient for AI-enabled autonomous vehicles, because of unknown scenarios that cannot be directly verified, and the incompatibility of formal methods with statistical deep neural network models.

The AV industry is in a trust race. It's certainly convincing to go on a ride where it seems the human is just there for show, or on rides where there's no human present at all. Therefore, companies carefully curate demo routes, avoid urban areas with cyclists and pedestrians, constrain geofences and pickup/drop-off locations, and limit the kinds of maneuvers the AV will attempt during the ride — all in order to limit the *number of disengagements*. As a result, disengagement-free driving has started to become a prerequisite for commercial deployment of AVs. Unfortunately, it has also been used by the media and others to compare technology from different AV companies, or as a proxy for commercial readiness.

The idea that disengagements serve as a meaningful signal about whether an AV is ready and safe for commercial deployment is a myth. 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 assessments today are self-reported by the testing companies, in good faith [6–19]. These companies develop different interpretations of what constitutes safe driving behavior. Autonomous miles driven and miles per disengagements are two metrics closely watched by industry observers to provide a high-level view of AV safety. Disengagements happen when either a safety operator detects bad behavior and takes control of an automated vehicle, or the vehicle itself detects something wrong and

*Equal Contribution

ICCPS '21, May 19–21, 2021, Nashville, TN, USA

ACM ISBN 978-1-4503-8353-0/21/05…$15.00
https://doi.org/10.1145/3450267.3450544

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.

Safety assessments for automated vehicles need to evolve beyond the existing voluntary self-reporting. There is no comprehensive measuring stick that can compare how far each AV developer is in terms of safety. Our goal in this research is to answer the following question: *How can we fairly compare two different AV implementations?* In doing so, the aim of this work is to make progress towards an innovative certification method allowing for a fair comparison between AVs by comparing them on similar traffic situations.

To do so, we propose a novel method called Scenario2Vector - which can transform data (video + other sensors) for a given traffic situation into a semantic embedding based on a Scenario Description Language (SDL) [20]. This embedding can then be used to compare AVs operating in similar traffic situations. The first step in designing such a unified safety certification scheme is to develop a unified representation of a traffic scenario. A standard scenario description language to define a traffic scenario will provide a high-level representation of the multi-agent interaction in a given video. Compared to a low level vector embedding, a high level scenario description for a video preserves human readable information about a traffic scenario. 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 traffic situations.

For this work, a scenario is defined as a short video clip captured from the front facing camera (can be extended to other sensing modalities). Figure 1[Left] shows the SDL encoding schema - comprising of a list of actors, actions, and scene elements. We use the BDDX [21] dataset to extract SDL descriptors from the sample videos. We focus on capturing the temporal structures of traffic scenarios to form a vector representation called Scenario2Vector (similar to *word2vec* [22] in natural language processing).

As indicated in Figure 1[Left], the SDL features are extracted from the last fully connected layers from the DNN which takes a sequence of images as inputs. The learned vector representation is an encoding of the video clip (or traffic scenario). As shown in Figure 1[Left], the embedding consists of an Actor-Action matrix, the columns of which denote which actors (ego, light vehicle, pedestrian, bicycle etc.) are performing which actions (slowing down, accelerating, turning, parking, stopped etc.); along with information about the traffic scene (intersection, freeway, stop sign etc.).

We can then use this encoding in the following manner. As shown in Figure 1[Right], consider a (fictional) AV company called "Driftic" which releases a database of its autonomous vehicle's operation. We would like to compare the performance of Driftic's AV with two other AV companies - Goodrive, and Idlewagon - who have released their own datasets from their AVs operating in different cities. Let's say we choose a specific traffic situation from Driftic's dataset as a reference - e.g. how the AV handles navigating a tunnel. Using the Scenario2Vec embedding of the reference video clip, our goal is to search Goodrive and Idlewagon's datasets for similar traffic situations (navigating a tunnel). To do so we compare the distance between the vector embedding of the reference clip to the samples from the other datasets, thereby allowing us to compare the performance of different AVs in similar traffic conditions.

This paper has the following research contributions:

(1) We present a first of its kind - **Traffic Scenario Similarity (TSS) dataset**. This dataset contains 100 traffic video samples (scenarios) and for each sample, it contains 6 candidate scenario videos ranked by human participants based on its similarity to the baseline sample.
(2) We then present **Scenario2Vector** - a scenario description language which can encode the actor, action, and scene elements of the traffic scenario video into an embedding which can then be used for similarity search.
(3) Finally, we rigorously compare our method Scenario2Vector with other (text-based) similarity approaches such as Sent2Vec on the TSS dataset using several similarity criteria: SDL one-hot, SDL Matrix, BLEU1, BLEU4, and METEOR.

2 RELATED WORK

The problem of video similarity has received a lot of attention from the computer vision, knowledge discovery and the machine learning research community in recent years. However, little to no work has been done to study this problem in the context of autonomous vehicles and their performance comparison. The related work spans several areas of inquiry and here we present an overview of the most related literature to our problem setting.

2.1 Video Retrieval with Embeddings

One method of performing video retrieval tasks is to model the spatial and temporal aspects of a video with an embedding. [23] uses this approach to learn a spatio-temporal embedding of a video that incorporates appearance, motion, and geometry using a causal convolutional network and a monocular self-supervised depth loss. The embedding space encourages video pixels of the same instance to be clustered together for video instance segmentation. This paper introduces a new spatio-temporal loss for video instance segmentation that maps video pixels to a high dimensional space, a temporal model that improves embedding consistency over time, and a depth loss. In contrast to embedding based approaches, region proposal based methods rely on a region of interest proposal network that predicts bounding boxes and then estimates the mask of the object in the box. However, embedding based approaches are becoming more popular due to their ability to better represent inter-relations of objects for segmentation tasks, and their granularity in mapping each pixel to a high dimensional space to avoid overlapping bounding boxes as is the case in region proposal based networks. The model in [23] consists of an encoder that encodes each input frame as a compact feature, a temporal model that learns a rich spatio-temporal representation of the video, and a decoder that outputs the instance embedding and depth prediction. Although an embedding based approach can be successful for segmentation tasks, it does not preserve high level, human readable information about the scenario, as a caption would. As a result, it can become difficult to interpret multi-agent interaction in a video from a low level embedding in a high dimensional space. A similar issue occurs in [24]. The Video2Vec approach encodes the videos in a global temporal encoding of the frame-level CNN features and combines this encoding with a Word2Vec model to embed the spatio-temporal features in a semantic embedding space where diverse videos sharing similar

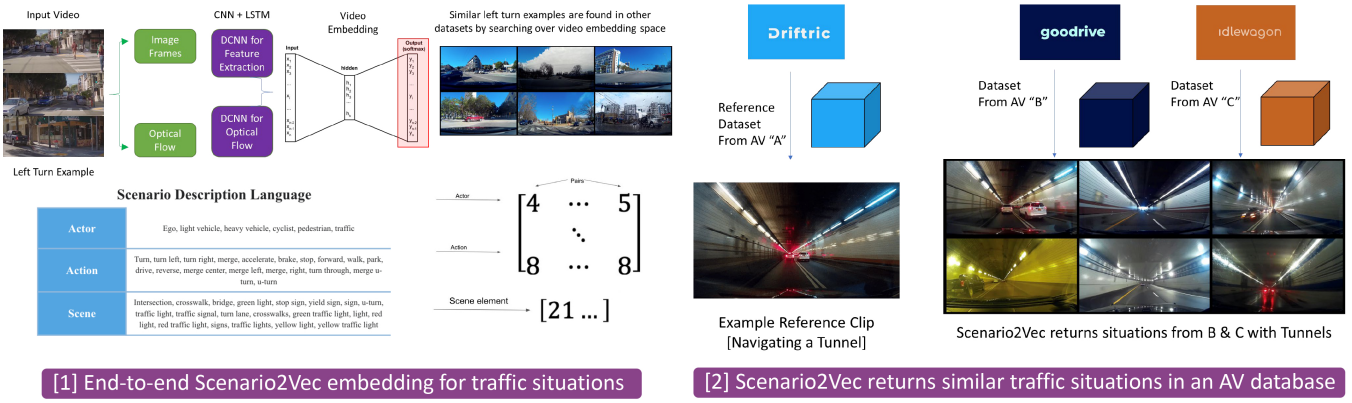


Figure 1: [Left] Framework of our Scenario2Vec representation architecture, and the scenario description language descriptors. [Right] The embedding is then used to search across multiple AV databases to output similar traffic situations. This allows for a fair comparison between different AVs (from different databases) based on the reference traffic scenario.

semantics are clustered together. However, the semantic portion of the embedding uses single words to describe the action taking place in a video and thus would not be suitable to capture a more detailed, high level description of a the scenario. Furthermore, video retrieval tasks based on video similarity in a semantic embedding space would not be as accurate for longer, more complex videos that only differ slightly in content. However, this case scenario is important to consider for applications in the AV field because although a scene with a car stopped at a red light without pedestrians compared to a scene with a car stopped at a red light with pedestrians might look very similar and get placed close together in a semantic embedding space, it's critical to identify pedestrians in the scene for safety reasons. As a result, using low-level embeddings to represent videos, particularly traffic scenarios, may not be the most desirable route. Instead, we need a representation that accounts for the importance of specific actors, actions and scene elements in a scenario to accurately model the multi-agent interaction in a video.

2.2 Characterization of a Traffic Scenario

As discussed in papers [24] and [23], it is difficult to provide a complex, high level representation of a video based on a low level spatio-temporal embedding. [24] suffered from a lack of complexity of the caption generated for a video and [23] did not preserve a high level, human readable representation 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. 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 [25]. This scenario description language outlines road segments, number of lanes, stop signs, actions, 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 [26]. 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.

2.3 Similarity Metrics

More recently, there have been accelerated developments in the field of video captioning. However, one of the major challenges in judging captioning models is evaluating their performance on video captioning tasks. Due to the lack of human generated ground truth data, we are left to rely on automatic evaluation metrics such as Bilingual Evaluation Understudy (BLEU) [27], Recall Oriented Understudy for Gisting Evaluation (ROUGE) [28] Metric for Evaluation of Translation with Explicit Ordering (METEOR) [29], Consensus based Image Description Evaluation (CIDEr) [30], Semantic Propositional Image Captioning Evaluation (SPICE) [31], and Word Mover's Distance (WMD) [32]. Due to the automatic and quantitative nature of these metrics, they are not always in perfect alignment with human judgement. Furthermore, the metrics are not always robust to the comparison of synonyms and different length sentences making it difficult to rely on them for the evaluation of a model. A video can be described in multiple different ways, which highlights another limitation of these evaluation metrics - they do not adequately account for multiple correct reference sentences in comparison to a candidate. As a result, these metrics still require human oversight to verify their reliability and validity. There exists a need for a systematic approach to reliably evaluate captions for similarity [33]. More specifically, what constitutes similarity in the autonomous vehicle domain with traffic scenarios may differ from more general purpose video captioning. For example, a bird captured in a traffic scenario matters less than a bird captured in a scene at the zoo. These subjective differences in evaluation make it

difficult to use automatic evaluation metrics to determine similarity between two scenarios. There exists a need for a weighted similarity score that takes into account the relevant parts of a traffic scenario for similarity analysis.

3 PROBLEM FORMULATION

3.1 Autonomous Vehicle Comparison Problem

The goal of our research is to provide a common metric that will facilitate the comparison of different autonomous vehicle algorithms. Consider a set $A = (a_1, a_2, \ldots, a_n)$ containing n different AV algorithms. Each algorithm takes a scenario v as its input and produces an action for the vehicle to take as its output. In order to compare the different AVs together, we need to observe them under similar traffic conditions or scenarios. Our goal therefore is to find similar traffic scenarios from the datasets generated by different AVs. Having found similar traffic situations, we can then observe if the output of one AV is more safe/optimal compared to another.

The first step in this process is to look through the recorded data from each AV algorithm and identify the individual scenarios. Let V be the set of recorded sensor data. For some given AV algorithm $a_i \in A$ we have sensor data $v_i \in V$ that contains all of the sensor data recorded by vehicle i. This sensor data v_i contains a sequence of scenarios that together comprise the activity of the vehicle: $v_i = (v_{i1}, v_{i2}, \ldots, v_{in})$. Therefore, a specific scenario can be defined as $v_{ij} \in V$, which is the jth scenario driven by vehicle i.

The next step in this process is to take these scenarios and convert them into high level embeddings. Given a scenario defined by a sequence of AV sensor data v_{ij}, we aim to create an embedding that preserves the relevant information about the scenario. Rather than learning the semantic representation, we define our own high level embedding S such that each value $s \in S$ is a tuple recording the information that defines a scenario. Thus our goal is to learn a function $f : V \rightarrow S$ that maps the AV sensor data $v_{ij} \in V$ defining a scenario into the equivalent embedding $s_{ij} \in S$. The main benefit is to use the embedding to enable us to compare the similarity of different traffic scenarios. This, in turn, will allow us to perform a similarity search between traffic scenarios.

3.2 Traffic Scenario Similarity Search Problem

In this paper we focus on the problem of performing similarity search between traffic scenarios. Given a scenario defined by a sequence of AV sensor data $v_{input} \in V$ as input, our goal is to find the scenarios in V that are most similar to our input. In order to determine the similarity of two scenarios, we define some distance function $d(\cdot)$. This distance function takes two scenarios as inputs, and returns the distance between them as a single number, with higher values indicating that the scenarios are less similar.

Note that the input to $d(\cdot)$ need not be sensor data from V. In fact, sensor data is particularly difficult to compare directly. Instead, we use the high level embedding returned from our mapping function as the input to our distance function. Since our high level embedding represents the scenario as a whole, a similarity between two high level embeddings indicates the similarity between the scenarios.

Therefore, given this information, our similarity search has the following steps. First, for each scenario in our dataset $v_{ij} \in V$, use the mapping function to compute the high level embedding $s_{ij} \in S$. Similarly, for an input scenario v_{input}, compute the high level embedding $s_{input} \in S$. Next, compute the distance between the input and the scenarios in our dataset $d_{ij} = d(s_{input}, s) \forall s \in S$. The scenario that is most similar to the input will be the scenario that provides the lowest value for d_{ij}.

4 SCENARIO2VECTOR

4.1 Video Dataset

In order to develop the mapping $f : V \rightarrow S$ as described in section 3, we use the Berkeley Deep Drive-X (eXplanation) dataset (BDDX) [34] as the primary source of data. This dataset contains 77 hours of driving footage, taken from a dashboard camera, which is split into 6,970 videos each of which is an average of 40 seconds long. The dataset also provides start and end times within each video to further divide the videos into multiple clips. In addition to the timestamps, each clip contains a human annotated textual explanation describing the action taking place. For this paper, we consider each of the sample clips from the BDDX data as an example of a traffic scenario v_{ij}, with a text caption represented by t_{ij}.

4.2 Scenario Description Language

Using the BDDX dataset containing dashboard footage of the scenarios $v_{ij} \in V$ where V is the set of scenarios, an equivalent embedding $s_{ij} \in S$ was extracted that represents what occurred in each scenario. Using a Bag of Words analysis, Figure 2 shows the list of actors, actions and scene elements that are present in the BDDX dataset. This elements shown are meant to be a starting point for investigating SDLs, not a comprehensive list of every possible component of a traffic situation. However, this approach is highly modular making it easy to add new elements as necessary.

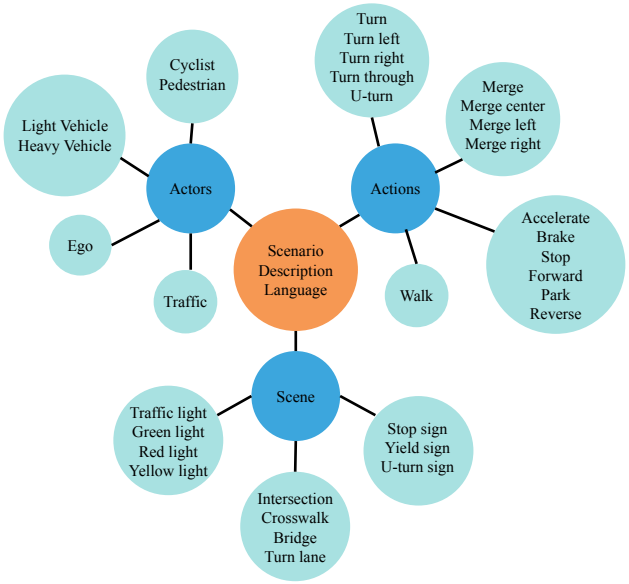


Figure 2: Characterization of scenario description language with the specific actors, actions and scenes that were extracted from the BDDX data.

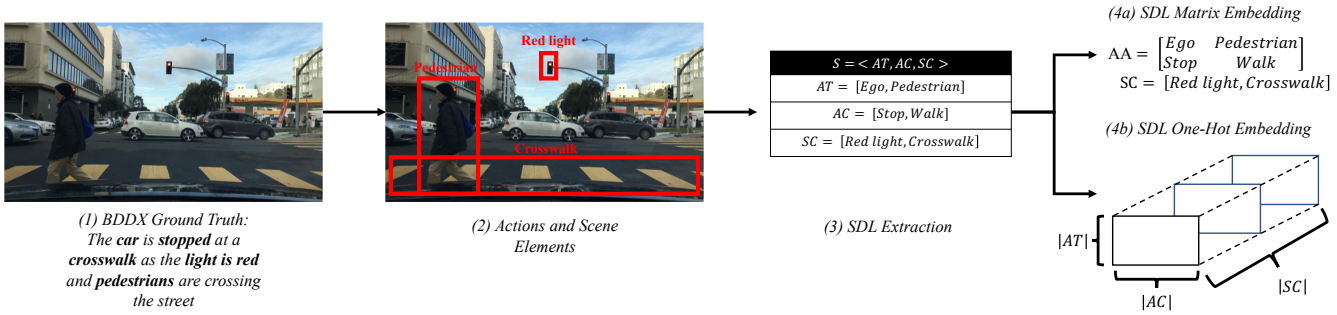


Figure 3: SDL Extraction. The actors are the car and the pedestrian where the car is stopped and the pedestrian is walking. There is also a red light and a crosswalk in this scene. The SDL is composed of three fields: actors, actions and scene elements. So the scenario descriptor fills in the ego vehicle (car), and pedestrian as actors, stop and walk as respective actions, and red light and crosswalk as scene elements. The automated SDL extraction process produces two embeddings: SDL Matrix embedding with actor-action matrix and a scene matrix, and a one-hot embedding of the actors, actions and scenes.

We used the captions from the BDDX data together with the elements described in Figure 2 as the basis for constructing our SDL. The Scenario2Vector SDL object is a tuple $S = \langle AT, AC, SC \rangle$ with each element of S representing a list that contains information about the content of the scenario. The list AT contains the Actors present in the scenario, with elements such as "Ego" and "Light Vehicle". Similarly, AC contains temporal Actions such as "Accelerate", "Brake", "Merge", while SC contains Scene Elements such as "Stop Sign", "Intersection", "Green Light". Including Undefined values and variations of elements listed in Figure 2, there are 6 possible actor elements, 19 possible action elements, and 20 possible scene elements that are valid entries to these lists. These three lists taken together encapsulate the relevant information about the scenario.

Figure 3 illustrates an example of a scene $v_{ij} \in V$ and its corresponding scenario embedding $s_{ij} \in S$ as returned by our SDL extraction method. The SDL extraction method returns two types of scenario embeddings: SDL one-hot embedding and SDL matrix embedding. Given a scenario $v_{ij} \in V$, the SDL extraction process identifies the actors, actions and scene elements from a ground truth caption t_{ij} provided in the BDDX dataset, as shown in Figure 2. The extraction process starts by searching the caption to find words that are associated with elements of our language. Each word that represents an actor, action, or scene element is added to the corresponding list in our SDL. Actions require extra analysis during this process in order to associate them with the specific actor performing said action. Each actor must perform some action, or else have a "No action" element associated with it. Similarly, for each action identified in the caption, there must be an actor performing that action, meaning that the number of actors and actions will always be equal. The scene elements, however, are independent from the actor-action pairs, and may differ in length. The actor-action pairs and scene elements are converted into the high level embedding s_{ij}. Thus, the SDL extraction process serves as a function that maps a scenario $v_{ij} \in V$ to an equivalent embedding $s_{ij} \in S$.

4.3 Scenario2Vector Embeddings

4.3.1 One-Hot Embedding. The SDL tuple $S = \langle AT, AC, SC \rangle$ encodes high level information about the actors and their associated actions as well as scene elements. In order to calculate similarity between two SDLs, a one-hot embedding was developed to represent the actors, associated actions, and scene elements for any given scenario. The one-hot embedding consists of a tensor with three dimensions: actor, action and scene. This embedding is illustrated in Figure 3[4b]. Each SDL tuple has a 7x20x22 one-hot embedding representing its action, actor and scene elements where each dimension has an extra element to account for NaN values. These one-hot embeddings are stacked on top of each other to produce a 4D tensor with dimensions [samples x actors x actions x scenes]. The advantage of this embedding method is that it makes it straightforward to compute similarity between different embeddings using Euclidean distance. However, the embedding is very sparse, with most SDLs containing two or fewer actors, and three or fewer scene elements. Furthermore, although there is no connection between the scene element and actors/actions in our representation of the SDL, the one hot embedding forces the scene to be linked to an actor-action pair. As a result, if there is more than one scene element, there is no deterministic method for embedding multiple scene elements, so the scenes will be intrinsically linked to actor-action pairs. This could provide misleading information when computing the Euclidean distance to determine similarity between two embeddings.

4.3.2 Matrix Embedding. To address some of the limitations of one-hot embedding, and also provide a more robust representation that could be used for further evaluation and comparison, we also considered an approach where the tuple structure was converted to an actor-action matrix and a scene matrix. This separation better depicted the information captured by the SDL since scene elements are considered to be separate from the actor-action pairs. The actor-action (AA) matrix is a [2 x n] matrix where n represents the number of actor-action pairs in each SDL tuple. A single SDL tuple always contains at least one actor-action pair because each video will always have an ego actor. The scene elements (SC) are encoded in a list and are independent of the actor-action matrix.

$$\text{AA} = \begin{bmatrix} at_0 & \dots & at_n \\ ac_0 & \dots & ac_n \end{bmatrix} \qquad \text{SC} = \begin{bmatrix} sc_0 & \dots & sc_m \end{bmatrix}$$

To determine similarity in the SDL space, a distance function was constructed to compare two SDLs using two assumptions. The first assumption is that the scenario follows a hierarchy with actors being the most important measure of similarity, then actions, and lastly the scene elements. For this reason, we punish SDLs more for having differing actors than for having differing scene elements. The second assumption is that, relative to a baseline scenario, a comparison scenario missing one element from the baseline is less similar than a comparison scenario with an extra element over the baseline. For this reason, we punish SDLs more for lacking elements than having extra elements. Under this assumption, our distance function is not a distance metric due to a lack of symmetry, but it still allows for comparison between SDLs. Each list from the SDL tuples is compared independently using an intermediate function. The Actor List uses an intermediate function $d'_{MR}(AT_{base}, AT_{comp}, w_{AT}, w_{missing})$ which returns two values - the number of elements present in the comparison list that are not present in the baseline list (calculated by $|AT_{comp} \setminus AT_{base}| \times w_{AT}$), and the number of elements present in the baseline list that are not present in the comparison list (calculated by $|AT_{base} \setminus AT_{comp}| \times w_{AT} \times w_{missing}$). The other lists from the SDL tuple are compared in similar fashion. After calculating these six values from our representation, we use them to form a vector, and define the distance function $d_{MR}(s_i, s_j)$ as the length of this vector.

4.4 Caption Evaluation Metrics

4.4.1 Sentence2Vector. Sent2Vec [35] is an extension of Word2Vec that provides an efficient model for learning sentence embeddings. The embeddings have useful properties for computing similarity between two sentences as the embeddings are derived from a pretrained model on a large dataset. As a result, Sent2Vec can serve as a powerful model for downstream tasks such as sentiment analysis, and calculating similarity between two sentences. Sent2Vec was used to compute a similarity score between two captions from the BDDX dataset. After computing sentence level embeddings for each of the captions in the BDDX dataset, cosine similarity is used to calculate the similarity score for each embedding. This served as one of the baseline scores for comparing how well the SDL metric performed in computing similarity between two SDL tuples.

4.4.2 BLEU and METEOR. Two other automatic evaluation methods were also used in addition to Sent2Vec to perform the same downstream sentence comparison task. Automatic evaluation of captions can be performed by a variety of metrics including Bilingual Evaluation Understudy (BLEU)[27], Metric for Evaluation of Translation with Expicit Ordering (METEOR)[29], Recall Oriented Understudy for Gisting Evaluation (ROUGE) [28] and several others as described in [33]. In this paper, the BLEU and METEOR similarity metrics were investigated. One of the advantages of using BLEU is that it has been widely adopted, and tends to correlate fairly well with human evaluation. The BLEU metric ranges from 0 to 1 where 1 implies a sentence is identical to a reference sentence. BLEU compares n-grams in the candidate sentence to n-grams in the reference text and calculates the score based on the overlap, regardless of the position of the n-grams. For example, n-grams can be unigrams where each word in the sentence serves as a token and the overlap of unigrams are compared between sentences. METEOR is an extension of the BLEU metric but first compares the reference sentence and candidate sentence using unigrams. However, METEOR also takes into account the stemmed forms of words, and their meanings using a WordNet model.

5 EXPERIMENTS AND RESULTS

5.1 Traffic Scenario Similarity Dataset

Our goal was to find a distance metric that would accurately determine the similarity of two input videos. In order to evaluate our distance metrics, we needed a ground truth measure of similarity. However, the BDD-X dataset does not have any measure of similarity between videos. Therefore, we created our own Traffic Scenario Similarity (TSS) Dataset out of the videos in the BDDX dataset.

The Traffic Scenario Similarity Dataset consists of 100 scenario samples. Each sample v'_k in the Traffic Scenario Similarity Dataset represents a scenario v_{ij} randomly sampled from the BDD-X dataset. From this scenario v_{ij}, we extracted the camera footage as a baseline video against which other scenarios could be compared. We chose six candidate videos in total to compare against the baseline video for each sample. For each scenario v'_k in our dataset, we have one baseline video b_k and six candidate videos: $C_k = [c_{k,1}, \dots, c_{k,6}]$

In addition to the baseline video and the six candidate videos, the TSS Dataset also includes a list of the ranking of how similar each candidate video is compared to the baseline video. This ranking list is represented as $\mathbf{r}_{gt,k} = [\mathbf{r}_{k,1}, \dots, \mathbf{r}_{k,6}]$. To obtain these ranking lists, we asked participants watch the baseline video and rank the candidate videos from most similar to least similar. When ranking the videos, participants were instructed to pay attention to aspects such as the actions of the vehicles in the video or the presence of road signs. Figure 4 shows the videos and the corresponding human evaluated similarity ranking of two samples taken from the Traffic Scenario Similarity Dataset. As can be seen from the image, the candidate videos can vary dramatically in context from the base video, but a human annotator has little trouble ranking them from most to least similar. Looking at the sample on the left, we can see that the highest ranked video (the bottom video) shows a vehicle waiting at a red light, the same scenario shown in the base clip.

The TSS dataset is different from the existing video similarity datasets described in Section 2. These other datasets do not have a subjective, human evaluated ground truth as the TSS dataset does. They are typically concerned with finding videos that look similar, and often perform modifications on existing videos to generate ground truth data. Instead of looking at visual similarity, our dataset explores conceptual similarity with a focus on scene understanding. Our ground truth data is not automatically generated, but created by human annotators. Even within the autonomous vehicle domain, this dataset is unique. There are many existing datasets for traffic situations, but none that look at comparing video similarity. Our dataset extends the BDD-100k dataset in a previously unexplored direction in this area. The dataset will be made publicly available at a later date.

5.2 Similarity Ranking Experiment Setup

Figure 5 provides an overview of the similarity ranking experiment. For each sample v'_k in the TSS dataset, we used the distance metrics defined in section 4 to calculate the distance between the baseline and each of the candidate videos. For the one-hot distance metric, we calculated list of distances $d_{OH} = [d_1 \dots d_6]$, where d_i is the distance between the baseline and candidate video c_i. These distances were then ranked from smallest to largest to provide a list of rankings $r_{OH} = [r_1 \dots r_6]$, where r_i is the rank of candidate video c_i. When there was a tie between two distances, we assigned those rankings the rank that most closely matched the ground truth data if possible. The rankings for each of the other metrics (r_{MR}, r_{S2V}, r_{BLEU1}, r_{BLEU4}, and r_{METEOR}) were all generated in a similar manner. The list of rankings generated by each metric distance function was then compared against the ground truth rankings for that sample, to assess how well the metric performed on the task of ranking the videos.

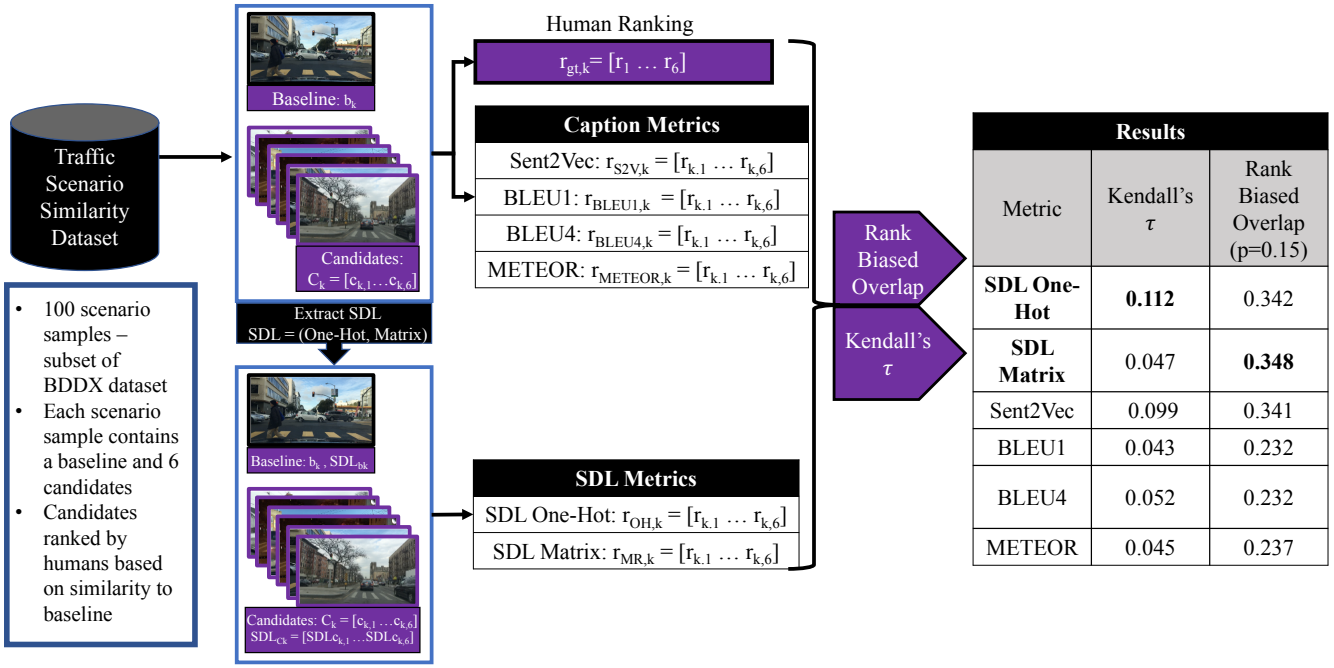
Figure 4: Dataset Samples. The Traffic Scenario Similarity Dataset Contains 100 samples. Each sample consists of a baseline video, and six candidate videos. For each candidate video, a human annotator has ranked its similarity to the baseline video to create a ground truth similarity ranking.

5.2.1 Kendall's Tau. In order to determine which distance metric was most accurate in predicting the ranks of the candidate videos, we looked at two different list similarity metrics. The first list similarity metric is Kendall's Tau. Kendall's Tau returns the correlation between two lists based on the number of items that are similarly ranked. A score of 1 indicates a perfect correlation, while a score of 0 indicates no correlation. The advantage of using Kendall's Tau is that it does not require any specific ordering of the data, as long as both lists have the same order of candidate videos when they are uniformly weighted. Kendall's Tau shows how accurate the metrics are in predicting the order of the candidate videos. It provides the same score for getting the two most similar videos wrong as getting the two least similar videos wrong.

5.2.2 Rank-Biased Overlap. Instead of a uniform similarity metric, an alternative method of measuring similarity between two ranked lists could be to use a weighted similarity measure. For instance, it makes sense to use a list similarity metric that places more emphasis on getting the most similar video (as reported by the human) correct. Therefore, in addition to Kendall's Tau we have chosen to also compare our ranked lists from the SDL and the Sent2Vec criteria to the human ground truth rankings using Rank-Biased Overlap (RBO) [36]. RBO assigns more importance to the first few elements of the ranked list by using a weight vector defined by a geometric series based on a hyper-parameter p. Thus, in order to give the most weight to the video that is most similar, it is necessary to order the lists from most similar to least similar. The amount of weight given to the most similar video as opposed to less similar videos varies depending on the value used for p. At $p = 0$, only the first element of the list is considered, and RBO returns either a 0 if the first element is not the same, or a 1 it is. As p approaches 1, the weights become flatter (although it never truly becomes uniformly weighted) to allow later elements to have more of an impact, at the cost of early elements being weighted less. As a side effect of RBO, a high value for p also implies that there are more elements for comparison, approaching infinite elements as p approaches 1. For a list with few elements, such as in our experiments, this has the effect of providing a very low RBO score at high values of p.

5.3 Similarity Ranking Results

5.3.1 Choice of p. Since there is no specific value of p for the RBO score that is guaranteed to work best for a list of depth six, we performed a hyper-parameter search of p. For each of our distance metrics, we compared the ranking list to the human labels using RBO as we varied the value of p. We set p to values ranging from



Results		
Metric	Kendall's τ	Rank Biased Overlap (p=0.15)
SDL One-Hot	**0.112**	0.342
SDL Matrix	0.047	**0.348**
Sent2Vec	0.099	0.341
BLEU1	0.043	0.232
BLEU4	0.052	0.232
METEOR	0.045	0.237

Figure 5: Experiment Setup. Each sample of the Traffic Scenario Similarity dataset contains a baseline video along with a set of 6 candidates. Each sample goes through the SDL extraction process which outputs a one-hot and matrix embedding for the baseline video and the 6 candidates. Human evaluation of the videos results in a human ranking of how similar the candidate videos are to the ground truth. Caption based metrics investigated in this study include: Sent2Vec, BLEU1, BLEU4, and METEOR. SDL based metrics include euclidean distance for the one-hot embedding and the metric defined in section 4.3.2 for the SDL matrix: $d_{MR}(s_i, s_j)$. Each of these metrics results in a ranked list of the candidate video with respect to the baseline video. The rank biased overlap and Kendall Tau's metrics take as input, the human evaluation and one of the other seven metrics and output a similarity score based on rank order.

0.01 to 0.99 in increments of 0.01. The results of these tests can be seen in Figure 6 [Top]. In a similarity search, it can be assumed that there are far fewer videos that are similar than those that are dissimilar. For this reason, we choose to focus on low values of p for RBO, values from 0.01 to 0.20 (the shaded region of Figure 6 [Top]). Figure 6 [Bottom] shows an example of the RBO weights for a low value of $p = 0.15$. These weights place the majority of the emphasis on correctly predicting the top ranked human video for each sample. Only a small amount of weight is given to correctly predicting the second highest ranked human video, and the other four videos are given almost no weight.

5.3.2 Interpretation of Results. The results of our experiments for the TSS data are summarized in Table 1. For each metric, we report the Kendall's Tau result as well as the RBO score at $p = 0.15$. We chose $p = 0.15$ as the point to report our RBO scores because at this value of p almost all of the weight is placed on the highest ranked sample, and the remaining samples have barely any weight as shown in Figure 6 [Bottom]. Out of all of the metrics, the SDL One-Hot embedding achieves the highest Kendall's Tau score, while SDL matrix embedding achieves the highest RBO score, with a 13% improvement over Sent2Vec. One-hot embedding doing well on the Kendall's Tau metric implies that it is the best metric for creating an overall ranking. Sent2Vec performs second best here, while matrix embedding and the other caption-based metrics lag behind. On the RBO metric, for $p = 0.15$ we see matrix embedding performing slightly better comparative to Sent2Vec, with a 2% improvement. One-hot embedding achieves nearly the same accuracy as Sent2Vec. *This implies that the SDL metrics perform the best at finding the single most similar video which is agreement with the highest similar reported video tagged by the human annotator.* As p increases to about 0.65, we see Sent2Vec begins to outperform the SDL metrics slightly, and the caption-based metric begin to close the gap. After $p = 0.65$, the RBO values begin to drop off as the weights become smaller and smaller. Despite the values becoming smaller, the metrics remain in the same relative order with Sent2Vec performing best and METEOR performing worst. The biggest improvement from Kendall's Tau to RBO is matrix embedding, which has the highest RBO score just above one-hot embedding. This implies that matrix embedding performs much better on the top few videos than it does on those that are less similar. These results show SDL metrics just barely outperforming caption-based approaches. The improvements made were incremental rather than significant, however it still shows that working in a high-level embedding space is as useful as working in text space.

Table 1: For each metric, the average value of Kendall's τ and RBO at p=0.15

Similarity metric	Kendall's τ	Rank Biased Overlap [RBO] (p=0.15)
SDL One-Hot	**0.112**	0.342
SDL Matrix	0.047	**0.348**
Sent2Vec	0.099	0.341
BLEU1	0.043	0.232
BLEU4	0.052	0.232
METEOR	0.045	0.237

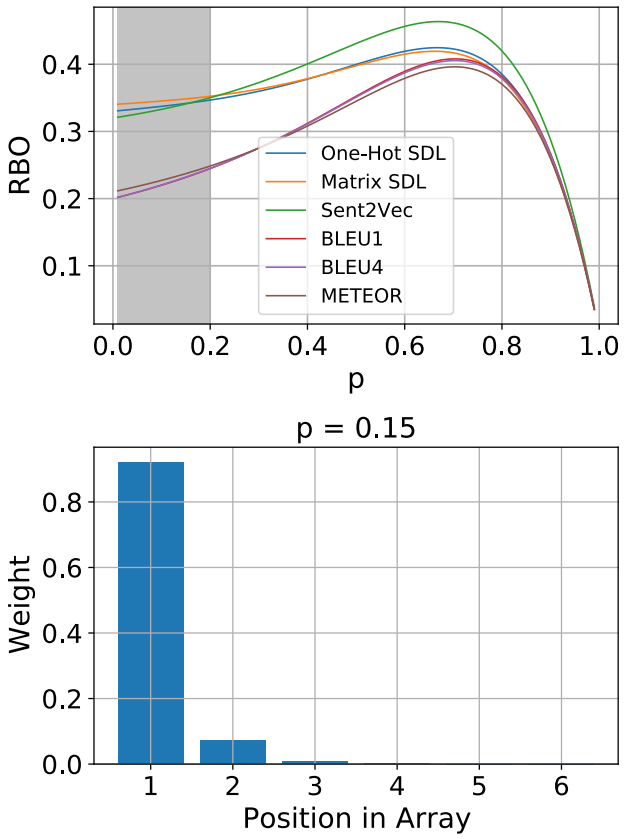


Figure 6: [Top] The plot shows the RBO for each metric as a value of p. [Bottom] The plot shows the distribution of weights with a p-value of 0.15 for the RBO metric.

5.4 Qualitative Analysis

Figure 7 Shows examples taken from the TSS dataset along with the highest ranked video from the one-hot embedding and the Sent2Vec metrics. Let us do a more in-depth analysis of example 1 from this figure. The video from this example shows the ego vehicle speeding up after a stop and following a yellow van. Along the sidewalk there are many pedestrians. The caption for this video is *The car accelerates because traffic is speeding up.* Of the six candidate videos, the one-hot embedding metric ranks the first video as being most similar. The first candidate video shows the ego vehicle speeding up after a stop and following another car. Unlike the baseline video, this video does not have any pedestrians, and there are many other vehicles on the road. The caption for this video is *The car is picking up speed and driving forward because traffic is picking up speed ahead.* The ground truth also ranked the first video as being the most similar. In contrast, the Sent2Vec metric ranks the second candidate video as being the most similar. The second candidate video shows the ego vehicle speeding up after a stop, with a truck in front of it. Again, there are no pedestrians and more traffic compared to the baseline video. Also, in this clip the ego does not match the speed of the truck in front of it, allowing the truck to get further away throughout the clip. The caption for this video is *The car accelerates because traffic is moving again.* The ground truth ranked the second video as being the second most similar.

The one-hot embedding metric rates the first candidate video as being very similar to the baseline because they contains the same information in the actor-action matrix. In the first candidate video, the ego vehicle is accelerating and traffic is also accelerating. These are the same actor-action pairs as the baseline video. Therefore, one-hot embedding metric gives these pair of videos a distance of 0. In the second candidate video, the ego vehicle is accelerating while traffic is moving. This simple difference in the action associated with the traffic causes the one-hot embedding metric to rank this video as much less similar, with a final ranking of fifth out of six.

The Sent2Vec metric rates the second candidate video as being very similar to the baseline because the caption for these videos are very similar. Both captions start with the same phrase, *The car accelerates because traffic is....* Since these captions are word-for-word identical up until the last two words, Sent2Vec determines that they must be very similar. In the first candidate video, the captions do not match each other as closely. Even though the meaning of the sentence is the same, the exact words used are different, which causes Sent2Vec to rank this video as much less similar, with a final ranking of fourth out of six.

Clearly, both metrics have their flaws. Since the SDLs have been extracted from the captions, an error in the caption will cause both both metrics to fail. If the captioning data fails to accurately represent the video, the metrics will rank them poorly. Similarly, if two different people provide a caption for the same video, they are likely to describe the video differently. With two different captions, Sent2Vec is likely to underestimate their similarity. The SDL extraction process aims to avoid this problem by identifying synonyms and placing the elements within predefined categories. However, this comes with a lack of nuance such that even a small difference become a large distance. A more robust approach to extracting the SDL information would likely increase its ranking accuracy.

6 CONCLUSION AND DISCUSSION

This paper lays the foundation for a novel method called Scenario2Vector - for generating a vector embedding for traffic scenarios. The vector embedding is based on a scenario description language which aims to capture the actor, action, and scene elements of a complex traffic situation into an equivalent embedding. The embedding is indicative of both the temporal and static elements of the traffic situation. The SDL based embedding can then

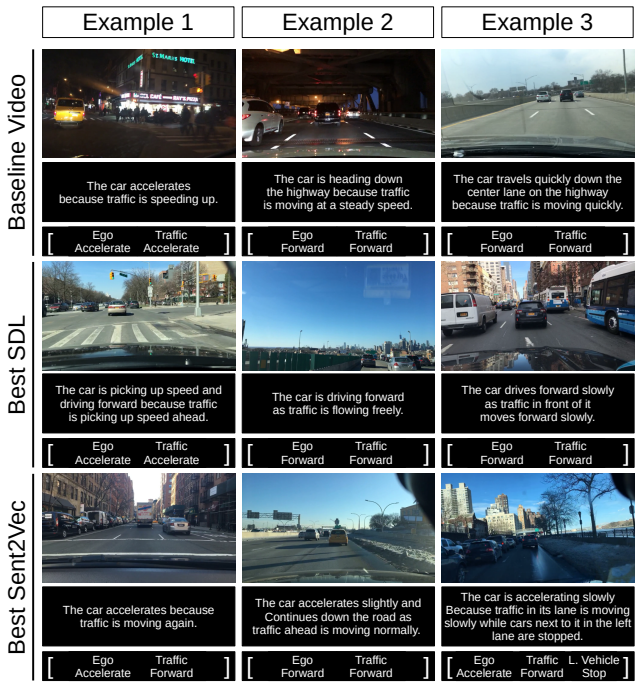


Figure 7: Qualitative results. For each of the chosen examples, we show a video frame, the caption, and the SDL actor-action matrix from the baseline video, the highest ranked SDL video, and the highest ranked Sent2Vec video.

be used to search for traffic situations similar to a baseline reference traffic situation; allowing for an equitable method of comparing two AVs operating under similar traffic scenarios. In doing so, we have also created the first of its kind Traffic Scenario Similarity (TSS) dataset. This dataset contains human rankings for similarity between 100 traffic scenarios and 6 candidate scenarios for each. Such a dataset will form the basis for a fair comparison and certification method for AVs which goes beyond the disengagement per miles used widely today. We compare our SDL based embedding with caption based methods such as Sentence2Vector and report findings which indicate that the SDL based method outperforms caption based methods by 13%.

Our ongoing and future work is focused on extending and releasing our TSS dataset to the CPS, autonomous vehicles, and computer vision communities. In addition, we will explore expanding and automating the SDL capture process to work with other sensor modalities - beyond video.

REFERENCES

[1] Pave poll: Americans wary of avs but say education and experience with technology can build trust. *https://pavecampaign.org/*, May 2020. https://pavecampaign.org/news/pave-poll-americans-wary-of-avs-but-say-education-and-experience-with-technology-can-build-trust/.

[2] Daisuke Wakabayashi. Self-driving uber car kills pedestrian in arizona, where robots roam. *New York Times*. https://www.nytimes.com/2018/03/19/technology/uber-driverless-fatality.html.

[3] Tesla car that crashed and killed driver was running on autopilot, firm says. *The Guardian*, Mar 2018. https://www.theguardian.com/technology/2018/mar/31/tesla-car-crash-autopilot-mountain-view.

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

[5] 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.

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

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

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

[9] 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.

[10] 2018 self driving safety report. *General Motors*. https://www.gm.com/our-stories/self-driving-cars.html.

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

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

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

[14] 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/.

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

[16] 2019 self-driving safety report. https://www.tusimple.com/wp-content/uploads/2019/05/TuSimple-2019-Self-Driving-Safety-Report.pdf.

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

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

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

[20] Marc René Zofka, Sebastian Klemm, Florian Kuhnt, Thomas Schamm, and J Marius Zöllner. Testing and validating high level components for automated driving: simulation framework for traffic scenarios. In *2016 IEEE Intelligent Vehicles Symposium (IV)*, pages 144–150. IEEE, 2016.

[21] Fisher Yu, Wenqi Xian, Yingying Chen, Fangchen Liu, Mike Liao, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving video database with scalable annotation tooling. *arXiv preprint arXiv:1805.04687*, 2(5):6, 2018.

[22] Xin Rong. word2vec parameter learning explained. *arXiv preprint arXiv:1411.2738*, 2014.

[23] Anthony Hu, Alex Kendall, and Roberto Cipolla. Learning a spatio-temporal embedding for video instance segmentation, 2019.

[24] S. Hu, Yikang Li, and Baoxin Li. Video2vec: Learning semantic spatio-temporal embeddings for video representation. In *2016 23rd International Conference on Pattern Recognition (ICPR)*, pages 811–816, 2016.

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

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

[27] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, page 311–318, USA, 2002. Association for Computational Linguistics.

[28] Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics.

[29] Satanjeev Banerjee and Alon Lavie. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan, June 2005. Association for Computational Linguistics.

[30] Ramakrishna Vedantam, C. Lawrence Zitnick, and Devi Parikh. Cider: Consensus-based image description evaluation. *CoRR*, abs/1411.5726, 2014.

[31] Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. SPICE: semantic propositional image caption evaluation. *CoRR*, abs/1607.08822, 2016.

[32] Matt Kusner, Yu Sun, Nicholas Kolkin, and Kilian Weinberger. From word embeddings to document distances. volume 37 of *Proceedings of Machine Learning Research*, pages 957–966, Lille, France, 07–09 Jul 2015. PMLR.

[33] Nayyer Aafaq, Syed Zulqarnain Gilani, Wei Liu, and Ajmal Mian. Video description: A survey of methods, datasets and evaluation metrics. *CoRR*, abs/1806.00186, 2018.

[34] Jinkyu Kim, Anna Rohrbach, Trevor Darrell, John Canny, and Zeynep Akata. Textual explanations for self-driving vehicles. *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[35] Matteo Pagliardini, Prakhar Gupta, and Martin Jaggi. Unsupervised learning of sentence embeddings using compositional n-gram features. *CoRR*, abs/1703.02507, 2017.

[36] William Webber, Alistair Moffat, and Justin Zobel. A similarity measure for indefinite rankings. *ACM Transactions on Information Systems (TOIS)*, 28(4):1–38, 2010.