Assessing Aggressive Driving Behaviour Using Attention Based Models*

Jonathan Aechtner[1[0009−0005−2781−851X]], Anna Wilbik[1[0000−0002−1989−0301]], and Mirela Popa[1]

Department of Advanced Computing Sciences, Maastricht University, Maastricht, The Netherlands
https://www.maastrichtuniversity.nl/research/department-advanced-computing-sciences
jonathan.aechtner@student.maastrichtuniversity.nl,
{mirela.popa, a.wilbik}@maastrichtuniversity.nl

Abstract. Aggressive driving behavior threatens road safety. In this paper, we utilize attention-based models and feature extraction techniques on the METEOR driving dataset to better understand it. After refining the dataset for our specific research needs, we implemented and evaluated the attention-based models OadTR and Colar, each showing distinct strengths and limitations. Notably, there's a consistent correlation between the frequency of an action in the training data and a model's classification accuracy.
Our research offers two primary contributions. Firstly, we combined the OadTR and Colar models into a novel hybrid that leverages categorical exemplars while predicting future frames. Secondly, we extract salient cues for model interpretability by tracing agent paths across spatial and temporal dimensions. These insights are especially valuable for autonomous vehicle applications where real-time interpretability and efficient computation are vital.

Keywords: Online action detection · Explainable Artificial Intelligence · Temporal Modeling · Driving Behaviour.

1 Introduction

Aggressive driving behavior (ADB) poses a significant threat to road safety. Its roots are tracing back to the early days of modern road infrastructure. J.J. Leeming highlighted the competitive and aggressive attitudes of drivers as early as 1969 [33]. Since then, research in this area has evolved, leading to a distinction between road rage and ADB. While road rage involves the intent to harm, ADB is characterized by a disregard for the safety and well-being of other road users, often motivated by impatience, annoyance, or time-saving [48].

With the advent of autonomous vehicles, understanding and predicting ADB becomes even more critical. Autonomous vehicles interact with human drivers, who may exhibit unpredictable and aggressive behavior. The ability of autonomous vehicles to recognize and respond to such behavior is essential for ensuring road safety.

This research pursues two primary objectives: firstly, to determine the most suitable attention-based model architecture for identifying Aggressive Driving Behavior (ADB), and secondly, to enhance the model's explainability by identifying salient cues essential for driving behavior assessment.

Historically, online action detection primarily relied on Recurrent Neural Networks (RNNs) and Long Short Term Memory networks (LSTMs). However, these models grappled with challenges, most notably their limited computational parallelizability and challenges in effectively retrieving significant information over extended sequences [16]. To circumvent these limitations, more recent research has embraced the attention mechanism. Notably, the OadTR leverages the transformative power of the transformer architecture, particularly using its decoder layers to anticipate future frames. While this methodology has shown promising results, it demands substantial computational resources for both training and inference [57]. In contrast, Colar innovatively bypasses the need to anticipate future frames. Instead, it identifies characteristic exemplars for each action category and expeditiously compares current frames to these pre-established exemplars [61], a strategy somewhat analogous to contemporary advances in information retrieval for language models. The fusion of these divergent strategies into a singular model architecture represents one of the salient contributions presented in this paper.

* Maastricht University

The second contribution made by this research regards the identification of salient cues. Prior works are often times tailored towards image data [11,19,36,40,44,46], designed for specific model architectures [11] and/or require extensive optimization [19,36]. To address these shortcomings, we present an occlusion based, optimization free and model agnostic approach, to generate local explanations, by harvesting prior information about agents.

This research holds significance in several aspects. It advances road safety by creating models that can identify and predict aggressive driving, especially vital for autonomous vehicles navigating alongside human drivers. By leveraging attention-based models, the study employs state-of-the-art technology to tackle real-world challenges. Additionally, the introduction of a real-time optimization-free approach provides insights into the relevance of each agent. The findings also bear significance for shaping policies and regulations concerning autonomous vehicles and road safety, especially as such vehicles become more common on roads.

2 Related Work

The task of detecting aggressive driving behavior in video data intersects several distinct research areas. To facilitate clarity and comprehension, this Section is structured according to the architecture of a typical deep learning-based model for online action detection. Consequently, the discussion begins with an examination of feature extraction methods, followed by a comprehensive overview of various model designs and approaches. Additionally, the psychological and behavioral aspects of aggressive driving behavior are explored, providing a broader context for understanding the problem at hand. A subsection dedicated to explainable artificial intelligence (XAI) methods for online action detection models is also included, highlighting their significance in ensuring the interpretability and accountability of the developed models. This organization allows for a coherent presentation of the diverse aspects involved in addressing the challenge of aggressive driving behavior detection.

2.1 Feature Extraction

Feature extraction is crucial in computer vision, with early methods relying on hand-crafted descriptors such as SIFT. However, trainable architectures have become the dominant approach, capable of automatically learning hierarchical feature representations from image and video data, thus enhancing performance across various tasks, including action detection [31,35,51]. The term "backbone" in deep learning literature denotes the feature extraction part of a larger neural network, which can be designed using convolutional neural networks (CNNs), attention mechanisms, or other methods [17,26,30,34]. The choice of backbone often depends on the dataset it was pre-trained on, with datasets such as UCF101 and Kinetics being a popular choice for action detection. However, they are more aligned with human action recognition, posing challenges for other action detection problems like driving behaviour [15,21,28,45]. While convolutional backbones, have limitations regarding kernel size and complexity with video data [32,43,58,62], they still achieve state-of-the-art performance on numerous tasks [22,29,55,57,63]. On the other hand, approaches based on the attention mechanisms are gaining traction, as they efficiently handle long-range dependencies and global context, especially in video data with a temporal dimension [17,22,34,50,62].

2.2 Online Action Detection

The task of online action detection can be regarded as a specialized variant of action detection, where the model is restricted to using only past and present frames to make predictions. In more formal terms, given a sequence of video frames $F = f_1, f_2, \ldots, f_t, f_{t+1}$, the objective is to predict the action label y_t at time step t using only the information available in the frames up to time step t, i.e., $F_t = f_1, f_2, \ldots, f_t$. The model is not permitted to use any information from future frames, meaning that y_t can only be inferred from F_t and not from any f_i where $i > t$.

This constraint limits the model's ability to leverage the full temporal context of the action, which often contain crucial information about the action's progression and outcome [13]. This makes the task of online action detection more demanding, as the model must be able to accurately predict actions based on potentially incomplete information [20].

For the detection of ADB, this constitutes a major challenge, because the appearance of aggression may be at the very end of the action. An exemplary situation would be an overtaking maneuver that seems non-aggressive until the final lane change. At that point, if the overtaking driver cuts in too closely, it might force the overtaken vehicle to brake abruptly. However, to provide valuable predictions it would be desirable to label instances as aggressive, before the real aggression takes place. On the other hand, aggressive drivers demonstrate clues constantly, implying that even an detection after an initial offence may be valuable.

In the early stages of online action detection, the focus was on extracting handcrafted features, such as dense trajectories [52] and improved dense trajectories [53], to represent and analyze video data. With the emergence of deep learning, researchers explored more advanced models, including 3D CNNs [49] and two-stream CNNs [42], which effectively leveraged spatial and temporal information from video sequences. The adoption of recurrent neural networks (RNNs), particularly long short-term memory (LSTM) networks [16], further facilitated the modeling of temporal dependencies in video data. However, LSTMs faced limitations such as difficulty in capturing long-range dependencies and lack of parallelizability.

A method to address the limitation of inaccessible future frames is to *anticipate future frame representations*, employing these predictions to enhance the classification accuracy at the current frame f_t. This idea is inspired by the fact that humans consistently think about present actions by anticipating the future [12]. Typically, this approach employs an encoder-decoder architecture [20,57]. In this architecture, the encoder is set up to learn frame representations that are most informative to the task. The decoder, on the other hand, is learned to provide predictions about future frame representations. Earlier works rely on LSTM and/or TRN cells [20,59] to model temporal dependencies. The introduction of the online action detection transformer (OadTR) constitutes a shift towards the transformer architecture, where encoder and decoder both employ the multi-headed self-attention mechanism [57]. While this yields good results, computing MSA across all input frames presents a computational challenge, that can be elevated by using a different strategy called *consulting exemplars*. The idea of this approach is to utilise the fact, that examples of a category share characteristics and can therefore be described by representatives. This presents the problem of finding these representatives, which the authors of the colar paper (derived from consulting exemplars) solve by clustering all examples of a single category and subsequently defining the example(s) closest to the cluster center as exemplars. This yields representative exemplars per category [61]. The differences between the implementations however, exceed aforementioned conceptual differences. A more detailed overview of differences between these two models will be provided in Section 3.

2.3 Explainability and Interpretability in Video Data

Deep learning models have gained widespread adoption in the field of computer vision due to their remarkable performance and ability to automatically learn complex features from large-scale data. However, these models are often referred to as "black-box" models, as their internal workings and decision-making processes can be challenging to interpret and understand [8]. This lack of explainability and interpretability poses significant concerns for stakeholders, particularly in high-stakes applications where transparency, trust, and accountability are crucial [23]. Moreover, the opaqueness of deep learning models can hinder the identification and resolution of biases or errors in predictions, which may have far-reaching consequences [25]. Consequently, there is a growing interest in developing Explainable Artificial Intelligence (XAI) techniques that can enhance the interpretability of deep learning models in computer vision and facilitate human understanding of their underlying mechanisms [4]. In order to optimize the efficacy of the explanations generated, it is imperative to consider not only the specific problem at hand but also the characteristics and needs of the intended user [2].

Two distinct approaches define the XAI landscape in computer vision today. While attribution methods and gradient based approaches focus on the model itself, perturbation approaches try to modify the input data to generate explanations.

Attribution methods and gradient based approaches have shown remarkable results. Their explanations are usually visualised as heatmaps, where the most influential regions of the image are highlighted [11]. Gradient-based methods leverage the backpropagation process, essential for training neural networks, to obtain gradients concerning each layer's input [41,44,46]. In contrast, attribution methods generate explanations by recursively decomposing a network's decisions into the contributions from preceding layers, with the theoretical foundation laid by Deep Taylor Decomposition (DTD) [38].

Notable contributions include Layer-wise Relevance Propagation (LRP) [5], Contrastive-LRP [24], Softmax-Gradient-LRP [27], and TIBAV [11].

Perturbation approaches [19], introduce minor alterations to the data, while monitoring the model's varying behavior. However, if cast as an optimisation problem, these methods are computationally demanding due to the requirement for separate model inference with each data perturbation. Shapley value methods [36] encounter similar issues.

The majority of explainable AI (XAI) approaches in computer vision are tailored to image input [11,19,36,40,44,46]. Adapting these methods to video data is not a trivial task. Partially, because computational overhead challenges are exacerbated when incorporating a temporal dimension.

2.4 Aggressive Driving Behavior

Driving behavior is influenced by a multitude of factors such as personality traits, age, experience, gender, distraction, weather conditions, and the influence of alcohol and drugs. Studies have shown that personality traits such as aggression, sensation-seeking, and anxiety can affect driving behavior [54]. Various methodologies have been proposed to assess the aggressiveness of driving behavior, many of which have emerged from the fields of sociology and traffic modeling [3,56]. The vast majority of these metrics rely on survey data, such as the Driving Anger Scale, which employs a 5-point Likert scale to measure the extent of anger experienced in a given situation [14]. This scale has been expanded by the Driving Anger Expression Inventory, which identifies four distinct modes of driver anger that can be aggregated to yield the Total Aggressive Expression Index [9]. A similar research direction was adopted by [60] and [47], where machine learning based approaches are used to identify the aggressiveness of a driver. While these approaches offer valuable insights into the sociological and psychological underpinnings of aggressive driving, their applicability to autonomous driving is limited, as they primarily focus on the individual driver. In the case of AVs, the challenge becomes identifying aggressive driving through the vehicle's driving style, rather than attempting to understand, avoid or mitigate driver aggressiveness. A notable contribution to this was the introduction of CMetric [9], a distance based approach where a Dynamic Geometric Graph is utilized to measure the distance between road users. The authors use a Dynamic Geometric Graph to measure the closeness to other road users. The information embedded in this graph is subsequently used to compute how likely a given road user is to drive aggressively, as well as how pronounced this behaviour is. However, due to the reliance of this approach on distance it is not easily adaptable to behaviors such as *Rule Breaking*. Additionally, the distance based measures need to be computed for every agent at every time-step, making it questionable if the real-time execution can be maintained for large number of agents. CMetric has been successfully applied to generate traffic simulations featuring aggressive drivers [37]. Another important aspect of understanding aggressive driving behavior is to establish a comprehensive list of behaviors, which are considered aggressive. Drawing from the literature, a variety of studies and sources have provided definitions of aggressive driving behaviors that can serve as a foundation for further research. In particular, [48] and [18] propose the following behaviors as indicative of aggressive driving:

- Tailgating
- Weaving in and out of traffic
- Improper passing (e.g., cutting in too close in front of a vehicle being overtaken)
- Passing on the road shoulder
- Improper lane changes (failure to signal)
- Failure to yield the right of way to other road users
- Preventing other drivers from passing
- Unwillingness to extend cooperation to motorists unable to merge or change lanes due to traffic conditions
- Driving at speeds far in excess of the norm which results in frequent tailgating, frequent and abrupt lane changes
- Running stop signs
- Running red lights
- Not respecting traffic regulations

By incorporating these behaviors into the analysis, researchers aim to develop a more robust understanding of aggressive driving that can be applied to the detection and assessment of such behaviors in autonomous vehicles.

3 Methodology

3.1 Research Design

The primary goal of this research is to determine if and how aggressive driving can be assessed using attention-based deep learning models.

The findings of this study will serve as a stepping stone for the automotive industry, particularly for autonomous vehicle development, as such vehicles could use information about aggressive driving agents to alter their future interactions with these agents.

The dataset used in this research is approximately 100GB of video data from the METEOR Dataset. It contains specific types of aggressive driving behavior, which were determined by refining the initial seven labels, present in the data, to four. Apart from the RGB-videos provided in the METEOR Dataset, no additional features or data sources were incorporated into the research.

Deep learning models were chosen for this research due to their widespread adoption in contemporary related work and their ability to handle large datasets, and learn complex patterns.

The first research objective is investigated by testing two attention based model architectures and evaluating their performance. Additionally, these two distinct architectures were aggregated, resulting in a third model that was evaluated on the online action detection task.

To address the second research objective, an occlusion based approach was employed. This approach excludes one agent at a time from the video and compares the resulting classification output to the classification of the original video. This strategy results in local explanations, providing an understanding of the influence of individual agents on the overall classification.

To alleviate the computational burden of the training process, this research uses a *ResNet-50 i3D* to extract spatio-temporal features. This model was pre-trained on the Kinetics dataset [7]. Even though this fine-tuning is not driving-specific, it was chosen for its wide adoption in literature, including the original implementation of the **Colar** online action detection model [61]. The model configuration used for feature extraction returns one feature vector for every 8 frames. Implying that the feature extraction f, which is defined as $\mathbb{R}^{t\times h\times w\times c} \rightarrow \mathbb{R}^{\lfloor \frac{t}{8} \rfloor \times 2048}$.

3.2 Dataset and Label Refinement

The METEOR driving dataset offers annotations for a large selection of rare and unusual behaviors, that are rarely featured in other driving datasets. Figure 4 in Appendix A provides a comprehensive overview on which behaviors are present in the data. While some of these behaviors can hardly be considered aggressive, others are explicitly described as aggressive in the accompanying publication [10].

The final decision to use the METEOR driving dataset for this paper was primarily driven, by its representation of rare and aggressive behaviors, as well as it being underrepresented in scientific publications.

Deriving Labels Encoding ADB from Meteor Dataset. To train models capable of detecting aggressive driving behaviors, we mapped the behaviors established as aggressive (based on the definitions from related work) to the labels native to the Meteor dataset [10].

To best utilize the given dataset, it was decided that labels explicitly affirming the existence of aggression should be considered alongside labels matching the definition from [48]. Lastly, the METEOR label *cutting* was approved for the final list of labels. The METEOR paper states that cutting represents an agent of interest interrupting the road crossing of a slower entity (e.g., bicycle, pedestrian, etc.)

To convert agent-level labels to frame-level, each frame is assigned a binary label based on the actions of all agents within it.

It is noted that these actions are independent of one another, implying that an agent can show multiple aggressive actions at the same time. This constitutes a multilabel case, where different labels can co-exist at the same time. This is noted, as it has far reaching consequences for model implementation and training.

Label Refinement. Initial experiments were conducted with all available agents and actions in the dataset. However, due to poor results, the scope of the experiments was iteratively narrowed. At first the number of actions was reduced based on their occurrence in the dataset. As can be seen

in Table 2, the number of frames featuring interesting behaviors as well as agents varies greatly. Following the intuition that aggressive action are typically performed in relation to another agent (Overspeeding being the exception to the rule), the initial configuration had an upper bound of $N^2 * A$ combinations of aggressive behavior where N is the number of agents and A is the number of aggressive actions. Following the findings from Table 2 only the agents *Car, MotorBike* and *Scooter* were included into the final configuration based on their frequent involvement in ADB. Additionally, the labels *LaneChanging(m)* and *LaneChanging* were merged into a single label, for their distinctive characteristic being irrelevant to the task of identifying ADB [10]. The ADB labels in the final configuration are: *OverTaking, LaneChanging, WrongLane* and *Cutting*

Lastly, a background label was added, as is standard practice in almost all recent works on online action detection [6]. Background is defined as the absence of all considered actions.

3.3 Models for Online Action Detection

OadTR. As outlined in Section 2, the Online Action Detection Transformer (OadTR) represents a significant advancement over previous designs, incorporating the transformer architecture into the online action detection task [57]. This architecture is primarily composed of two key components: the encoder and the decoder.

OadTR consists of two main components, the encoder and the decoder, both of which are inspired by the standard encoder-decoder architecture of current transformer models. As such they employ multi-headed self attention.

The encoder incorporates a class token to learn global features pertinent to the online action detection task. This is done to ensure that the final feature representation is not disproportionately influenced by the initial value of a specific feature.

Conversely, the class token, via its adaptive interplay with other tokens within the encoder, can yield a semantic embedding more fitting for feature representation and for capturing the global context of the video sequence. Empirical support for these theoretical postulates is provided by experiments showing that the feature similarity for $t = 0$ dramatically diminishes before and after a pass through the network when a class token is employed.

While the encoder's responsibility lies in accurately capturing temporal dependencies and relations, the decoder's function can be unerstood, as predicting the future of $t = 0$, achieved via learnable predictive queries. To optimally leverage semantic information from the encoder, the decoder incorporates the encoder-decoder cross-attention mechanism.

An important difference to the original Vision Transformer is that the OadTR implementation permits the parallelization of the prediction queries' decoding at each decoding layer [57].

The initial implementation of OadTR is tailored to a multiclass problem. To adapt it to multilabel problems, a few modifications were required. The most significant adjustment involved updating the loss function. Following initial trials, the PyTorch implementation of BCEWithLogitsLoss was employed [39]. As can be seen in Equation 1, this loss function amalgamates binary cross-entropy loss with a sigmoid layer, offering an additional advantage of eliminating the necessity of the softmax operation implemented by OadTR for the final classification outputs and the supervised training of the decoder.

$$\ell(x, y) = \frac{1}{N} \sum_{n=1}^{N} -w_n \left[y_n \cdot \log \sigma(x_n) + (1 - y_n) \cdot \log(1 - \sigma(x_n)) \right] \tag{1}$$

We used Adam as implemented by PyTorch, to optimize the model. A notable divergence from the OadTR implementation was the use of a CosineAnnealingLR scheduler, which allows for dynamic updates of the learning rate during training. This modification was considered necessary as the original OadTR implementation employed a StepLR scheduler, which is significantly influenced by the step-size. This is a hyper-parameter that would have necessitated substantial effort to optimize. The revised approach thus aimed to make the training process more efficient, without compromising on model performance.

In terms of regularization, two distinct strategies were employed. The first involved the use of the *weight_ decay* parameter within the Adam optimizer, and the second incorporated dropout layers dispersed throughout the network. Furthermore, the size of the model itself served as an additional regularization factor. This multi-faceted approach to regularization aimed to prevent overfitting and enhance the generalization capabilities of the model.

Colar. The second model employed for online action detection is the Colar Model [61]. This model was chosen for its novelty, its lower computational footprint compared to the OadTR model and its innovative approach to action detection by consulting representative examples of each action category. An overview of the conceptual differences between the OadTR model and the Colar model can be found in Figure 6 in Appendix B

The Colar Model operates by comparing each frame in a video sequence with both previous frames and category exemplars. This dual comparison is facilitated in distinct branches, named dynamic branch and static branch.

The dynamic branch models the temporal process of actions. It does this by comparing each frame with its predecessors. During this process, local features are aggregated to obtain a combined feature representation of the entire input sequence.

The static branch, on the other hand, is designed to capture the unique characteristics of each action category. It does this by comparing each frame with a set of representative exemplars for each category. These exemplars are selected based on a simple clustering approach where instances closest to the cluster centers are used as exemplars.

The original Colar implementation fuses predictions from dynamic and static branches according to the following formula: $s = \beta\hat{s}^s + (1-\beta)\hat{s}^d$, where $\hat{s}^s$ are the predictions from the static branch and $\hat{s}^d$ are the predictions from the dynamic branch. Ablation studies found the configuration $\beta = 0.3$ to be optimal. However, in order to make the fusion of the final predictions learnable as well, our modification concatenates predictions from static and dynamic branches along the first dimension. The subsequent prediction vector is then fed into a MLP to aggregate a final prediction score per class.

Since the Colar model was implemented for a multiclass problem and does not natively support the multilabel case, some modifications were necessary. Similarly to the OadTR model, the most significant change was the implementation of the BCEWithLogitsLoss [39]. Other modifications regarded the replacement of Softmax Layers with Sigmoid layers, for both the final classification and the measure of similarity within the static branch.

We used Adam as implemented by PyTorch, to optimize the model. Regularization was facilitated, by utilizing Adam's *weight_ decay* parameter. Following the same intuition as with OadTR, a CosineAnnealingLR scheduler was used to reduce the learning rate over the course of the experiments.

Combining OadTR and Colar. The exploration of a comprehensive approach to online action detection in this study involves the integration of the static branch of the Colar model with the OadTR model. The static branch of the Colar model provides a robust mechanism for capturing category-specific features, thereby enhancing the model's ability to distinguish between different action categories. The OadTR model contributes its efficient and effective handling of temporal dependencies, enhancing the model's ability to track the evolution of actions over time.

The integration of these models is designed to harness the unique strengths of each. It is expected to yield a model with only slightly increased computational demands (compared to OadTR baseline), yet significantly improved classification performance.

OadTR already aggregates features right before computing the final classification scores. Extending this aggregation to also include the output of the static branch of the Colar model emerges as a straightforward implementation.

Following the notations from the Colar and OadTR papers, Equation 2 is derived, explaining how the final prediction p_0 is made. A visual representation of the proposed architecture can be found in Figure 1.

$$p_0 = \text{sigmoid}(\text{Concat}[m_N^{\text{token}}, \tilde{Q}, \hat{s}^s]W_c) \tag{2}$$

In Equation 2, the parameter m_N^{token} represents the task-related features in the OadTR encoder, $\tilde{Q}$ depicts the pooled-predicted features from the OadTR decoder and $\hat{s}^s$ is the output of the static branch of the Colar model. W_c are the parameters of a fully connected layer that is used to combine features.

3.4 Identifying Salient Cues

The identification of salient cues in video data is a crucial aspect of this study. To facilitate this, a masking technique is employed, which sets parts of the input video to zero, effectively excluding certain

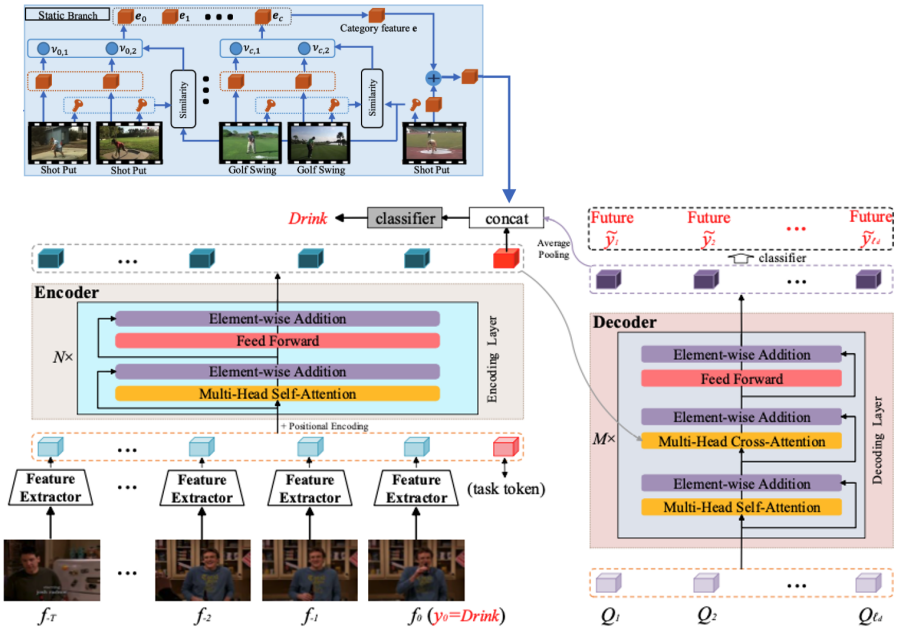


Fig. 1: Proposed new architecture for combination of OadTR and Colar. Adapted from the original figures in [57] and [61]

agents from the frame. Methodological, it therefore belongs to the perturbation based approaches. However, as we harvest prior information about the bounding boxes of agents, no optimizations are required, significantly reducing the computational overhead.

Essentially this approach only runs $n + 1$ inferences, where n is the number of agents in the video sequence (for simplicity, we only consider agents in the most recent frame). To establish a baseline we run inference on the unmodified video. For each of the following n inference runs, we mask a unique agent in the video sequence by setting all pixels in the bounding box defining that agent to 0. The prediction results are then compared to the established baseline indicating the influence of that agent for the baseline prediction. This is formalized in Equation 3.

$$\text{Influence}_{\text{agent}_i} = \text{classification}(\text{baseline}) - \text{classification}(\text{video excluding agent}_i) \tag{3}$$

While this approach provides a mechanism to identify salient cues in the video data, it should be noted that this method may produce misleading results in cases where agents occlude each other, as can occur in the dataset.

An example of this approach can be seen in Figure 2. The original prediction for this scene has LaneChange at 0.996. However, after occluding the red agent, this prediction drops to 0.012, which leads to $\text{Influence}_{\text{agent}_i} = 0.984$.

While the use of occlusion or masking to provide insights into video data and online action detection is not a common approach, it is related to other techniques in the field of computer vision and machine learning. For example, the use of key-frames in action recognition and the use of 3D and 4D data for facial expression recognition aim to capture salient features or cues in the data. This study extends these concepts by applying a masking technique to identify the influence of specific agents on the classification outcome. Nonetheless, it's worth noting, that elements like background can influence classification. Due to the inherent design of our approach, such effects aren't captured in the resulting explanations.

In conclusion, the local explanations provided by this occlusion-based approach offer tangible insights for targeted improvement of the model's performance, aligning with the primary goal of building robust and interpretable models for aggressive driving behavior assessment. Additionally, these explanations hold significant potential for applications in the domain of autonomous vehicles, specifically in identifying agents that are responsible for a positive classification.

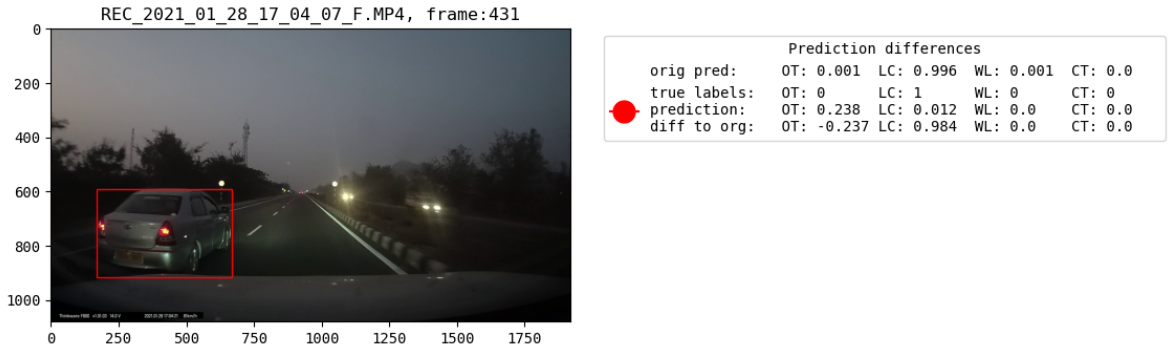


Fig. 2: Example for the identification of Salient Cues

4 Experiments and Results

4.1 Model Training and Evaluation

To improve training speed, the training process was split into two stages, where stage one was dedicated to extracting features from the video data and stage two revolved around model training.

The dataset was split randomly according to video names using 80% for training and 20% for testing. This approach averted scenarios where the model would use time steps for evaluation that were also used for training, thus preventing data leakage. Despite the split being done without regards for the number of frames per category in each video, Figure 5 in Appendix A shows, that the result approximates the intended 80/20 ratio.

The hyperparameters and regularization techniques varied for different models and were discussed in the introduction of the model architectures.

To address the issue of class imbalance among the different labels, a weighted loss function was used, where the weights are calculated based on the Equation 4.

$$\text{Weight}_{\text{Category } i} = \frac{\text{TotalFrames} - \text{Frames}_{\text{Category } i}}{\text{Frames}_{\text{Category } i}} \tag{4}$$

Model performance was evaluated using the ROC-AUC score to assess how much better the model performed compared to a random classifier. Additionally, the metrics F1 score, precision and recall are used to compare the performance of different model configurations. These metrics are used as implemented by sklearn [1]. To reduce the impact of the class imbalance on the metric results, the parameter *average* was set to *weighted*. Based on this, the support of each class is used as a weight, so that the average is calculated according to the class distribution.

Additionally, Mean Average Precision (mAP) is used as well. Mean Average Precision (mAP) is a widely used metric, computed by taking the mean of average precision per class.

Furthermore, multilabel confusion matrices were generated and used to visualize the model performance as well.

This research was made possible, in part, using the Data Science Research Infrastructure (DSRI) hosted at Maastricht University. Model training and evaluation were performed on a single NVIDIA Tesla V100 - 32GB GPU provided by the DSRI.

4.2 Experimental Setup

To maintain uniform experimental conditions, consistent hyperparameters were used. A comprehensive list can be found in Appendix C, where the tables 3 and 4 depict the hyperparameters for Colar and OadTR respectively.

OadTR. As a different learning rate scheduler was used, the parameters *lr_drop* and *lr_drop_size* are of no importance to the experimental setup. In the interest of ensuring a comprehensive understanding and facilitating seamless replication, these parameters have been reported nonetheless. Certain parameters, for example *weight_values* and *thumos_data_path*, are defined at runtime.

To ensure fair comparison with the Colar model, the number of time-steps given to the model was increased from 63 to 64. Based on each feature incorporating information of 8 frames, this implies that the model can utilise information from $8 * 64 = 512$ frames.

Colar. As can be seen in Table 3 in Appendix C, there are much fewer hyperparameters for the Colar model than for the OadTR model. The modifications made are specific to the dataset and the available hardware and bear little importance to the setup of the model.

4.3 Testing Different Model Architectures

In the pursuit to answer the first research objective, three different architectures were explored. As introduced in Section 3, these models are the OadTR model, the Colar model and an integrated model using the OadTR skeleton, but enhancing it by the static branch of the Colar model. Subsequently this integrated model will be dubbed OadTR_v4, complying with naming conventions introduced by OadTR [57].

For fair comparison, all three models were trained for 50 epochs, regardless of wall-clock time. It is noted, that training the Colar model is significantly faster than OadTR.

Table 1: Performance metrics represent the weighted class average of the impact of different model architectures.

Experiment	ROC_AUC	F1	Precision	Recall
OadTR_v3	**0.668**	0.671	0.686	0.659
OadTR_v4	0.661	0.658	0.685	0.633
Colar	0.613	**0.682**	**0.688**	**0.688**

The OadTR_v3 model showcased outstanding results, especially with an ROC_AUC of 0.668 and a precision of 0.686 according to Table 1. While it excelled in the 'Background' category with an F1 score of 0.799, there is potential for enhancement in other categories.

In comparison, the OadTR_v4 model had notable precision (0.685) but showed room for growth in the 'WrongLane' and 'Cutting' categories. Noteworthy is its superior performance in the 'LaneChange' category, achieving an F1 score of 0.138.

Differing from the others, the Colar model was balanced in its metrics, boasting the highest F1 and recall scores at 0.682. Specifically in the 'Background' category, it achieved an impressive precision of 0.832. Additionally, its strength in detecting 'OverTaking' instances was evident with a recall of 0.682, and it outperformed in the 'WrongLane' and 'Cutting' categories compared to its counterparts.

Among all models, the Background category had the strongest performance. F1 scores ranged from 0.777 (OadTR_v4) to 0.799 (OadTR_v3). Colar demonstrated superior precision, marking its effectiveness in recognizing true positives. In the *OverTaking* category, performance was slightly lower across all models. However, Colar stood out with a recall of 0.682, indicating its capability in detecting 'OverTaking' instances and reducing false negatives. All models showed sub-par performance in the *LaneChange* category, highlighting the difficulty in predicting this action. OadTR_v4 had a slightly better F1 score of 0.138, showcasing balanced precision and recall. Correctly predicting the *WrongLane* category proved difficult for all models as well. However, the Colar model did indicate a marginally better F1 score, suggesting potential for improvement with further adjustments. Cutting Category: All models faced challenges due to limited representation in the training data. Still, Colar managed to achieve the highest F1 score of 0.041, hinting at its resilience for this category.

In summary, all models demonstrated varying performance across different categories, each excelling in certain areas while requiring improvements in others. The Colar model showed balanced performance, OadTR_v3 outperformed in precision and ROC_AUC, and OadTR_v4 exhibited strength in the 'LaneChange' category. Despite the strong performance in the 'Background' category, all models demonstrated a need for further optimization for the less represented categories such as 'Cutting'.

A more visual representation of the model performance can be found in Figure 7 in Appendix D.

4.4 Generating Salient Cues as Explanations

The methodology employed to identify salient cues for the elucidation of the model's predictions is detailed in Section 3.4. In these experiments, a masking technique was employed which involved setting parts of the input video to zero, thus effectively 'removing' certain agents from the frame.

This masking approach creates a suite of $n + 1$ videos, where n represents the total number of agents in the frame. The prediction results from these masked videos are then compared against a baseline prediction from the unmasked video. The comparison provides a measure of influence of each agent on the classification outcome.

In addition to the primary method, an alternative technique utilizing negative masks was also investigated. This technique deviates from the positive masking approach, in that it preserves only the actor of interest within each frame, with all other regions being set to zero.

However, the use of negative masks led to undesirable outcomes, in that the classification results appeared to be random and yielded no noticeable improvements. In fact, the lack of contextual data within the frames due to negative masking, led to significant declines in classification accuracy.

Hence, following initial exploratory trials, the strategy of using negative masks was abandoned due to its detrimental effects on classification performance. However, it is noteworthy that the current codebase is designed to allow for swift adaptation and potential future exploration of negative masks.

The reported values are generated by applying a sigmoid function to the last layer of the neural network. Given the nature of the multilabel classification problem at hand, the outputs of this operation can be interpreted as probabilities. Each output corresponds to the probability of the associated class label being present, as estimated by the model. This is because each output logit represents a separate binary classification problem, independent of the others. It's worth noting that these 'probabilities' do not sum to 1 across classes, reflecting the fact that multiple class labels can be simultaneously present in this multilabel context.

Generally speaking it would be difficult to incorporate video explanations into this paper. In order to provide some visualization and allow for a mapping of probability differences and agents, it was decided to use the last frame as a reference. Therefore each explanation consists of a visual representation of the last image next to a legend, which maps the colored bounding boxes to probability estimates and the difference between these new differences and the original ones.

5 Discussion

The discussion Section seeks to elaborate on the findings presented earlier, provide possible explanations, and propose future research avenues. To aid further discussion, the first part focuses on the generation of salient cues, which is important as these are used in subsequent parts of this discussion. The second part focuses on the performance of three architecturally different online action detection models, which were evaluated.

5.1 Discussion on Results for Different Model Architectures

The results from the experiment examining various model architectures reveal distinct strengths and weaknesses for each model, highlighting potential areas for improvement.

OadTR_v3. The OadTR_v3 model demonstrated notable performance in terms of ROC_AUC and precision, leading the three models with respective values of 0.668 and 0.686. This model exhibited a particular strength in discerning the 'Background' category, attaining the highest F1 score of 0.799. However, its somewhat subdued performance in other categories suggests opportunities for further refinement. The model's suboptimal results in the *LaneChange*, *WrongLane*, and *Cutting* categories underline the need to enhance its proficiency for less represented categories in the dataset.

Colar. The Colar model demonstrated robust overall performance, registering the highest F1 and recall scores of all models at 0.682. Its strong proficiency in handling the 'Background' category, denoted by the highest precision score of 0.832, indicates a particular strength in identifying true positives. Additionally, the model showed superior sensitivity in the *OverTaking* category with a recall

score of 0.682, surpassing the other two models' performance. The Colar model notably outperformed other models in the *WrongLane* and *Cutting* categories. However, its relatively lower ROC_AUC and precision compared to the OadTR_v3 model suggest potential areas for further refinement. Its comparatively superior performance on underrepresented categories suggests that categories with fewer training/testing examples especially benefit from comparison with representative exemplars.

OadTR_v4. The OadTR_v4 model, a synthesis of OadTR and Colar concepts, exhibits remarkable precision, scoring 0.802 for the 'Background' category. The model however underperforms in overall ROC_AUC and recall. Interestingly, OadTR_v4 outshines all other models in the less frequently encountered 'WrongLane' category, with an F1 score of 0.076 - almost twice the score of its predecessor, OadTR_v3. This suggests that the integration of Colar and OadTR methodologies is particularly effective for rare labels. On the other side, the model failed to classify any instances of 'Cutting', as indicated by zeroes across all metrics for this category. This underscores the need for further refinements in the merging of OadTR and Colar concepts. Despite these shortcomings, the unique strengths displayed by OadTR_v4 imply a potential for the integrated model, particularly when adjustments are made for better assimilation of the constituent models' features.

The comparable performance across the models, particularly in less represented categories, may be more reflective of the characteristics of the dataset rather than the inherent capabilities of the models. This observation suggests that enhancements in the dataset, such as increasing the representation of underrepresented categories, could potentially augment the performance of these models.

5.2 Discussion on Salient Cue Generation

While no quantitative observations have been made, several factors appear to qualitatively influence the efficacy of the generated explanations.

Actor Size. By definition, larger, more prominent agents have a greater impact on the extracted image features and thus are more likely to cause substantial variation in classification results. This effect is exemplified in Figure 3c, where the explanations for *Cutting* are dominated by the orange agent, which is more prominently positioned than the red or blue agent. Additional examples in Figure 10 in Appendix E confirm this effect for the other implemented models.

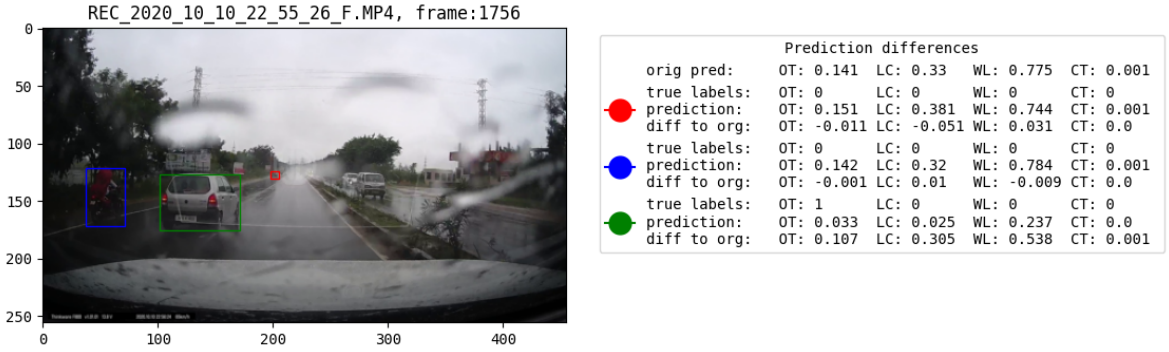
Number of Agents. The effectiveness of the explanations seems to be inversely correlated with the number of agents. As the scene becomes more crowded, the explanation is less likely to identify the correct agent with a high level of certainty.

Contrasting Aggressors and Victims. A key challenge of this approach is the lack of a reliable way to differentiate between an aggressor and a victim of aggression. In terms of the generated explanations, there is minimal conceptual difference if the aggressor or the victim of aggression is omitted from the scene. Specifically, an overtaking maneuver can only be labeled as aggressive if there is an agent being overtaken aggressively. Removing one of the two from the frame should yield similar results.

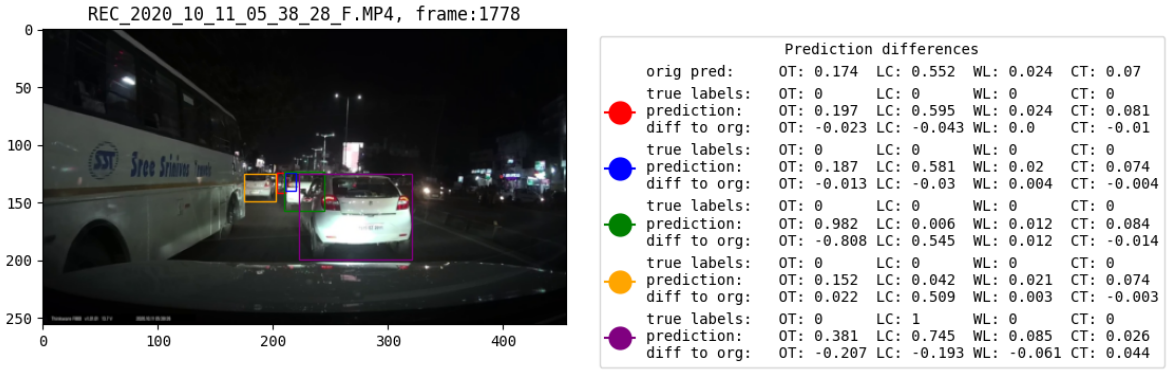
Background Light. It was initially hypothesized that masking agents in black would be less effective in night scenes. However, the examples generated contradict this assumption. One possible explanation could be that the model predominantly focuses on the lower half of the frames, often disregarding the upper half which tends to be less critical for its objective. Moreover, even during nighttime, the frames rarely, if ever, contain entirely black pixels, especially within the road sections.

Utilizing Visual Explanations. To gain a better understanding, we will discuss some examples generated with the *OadTR_v4* model, which can be found in Figure 3.

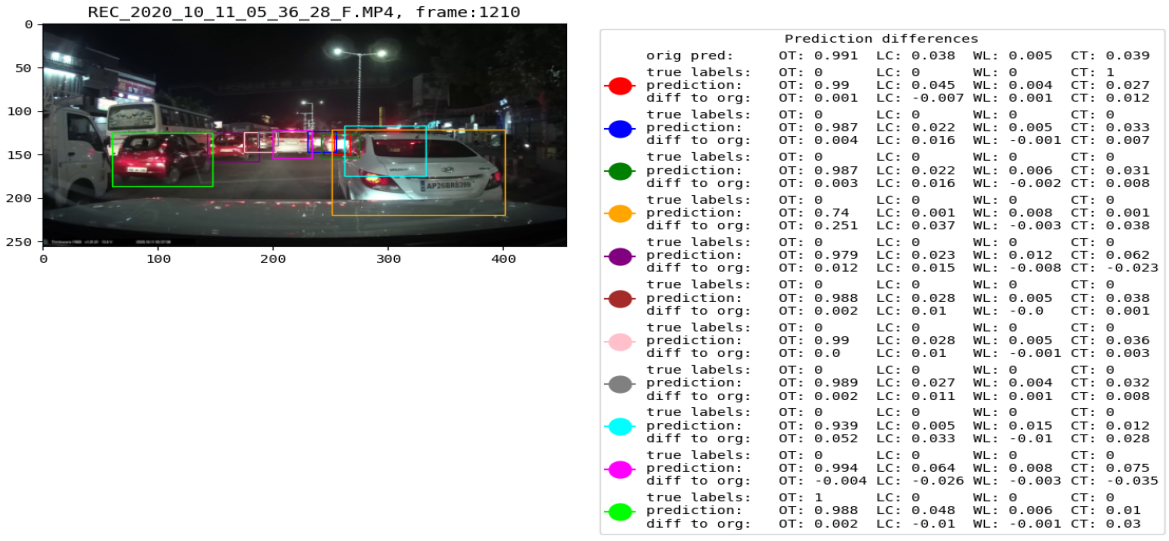
In the first example depicted in Figure 3a, the model incorrectly interprets the action *OverTaking* as *WrongLane* or *LaneChange*. This confusion might arise from the fact that *OverTaking* often involves elements of both these actions. Additionally, in the original scene the green agent overtakes the blue one on the wrong side, making this an especially challenging case. With regards to *OverTaking* explanations,



(a) OverTaking



(b) LaneChanging



(c) Cutting

Fig. 3: Explanations generated using OadTR_v4

it is noteworthy that the v4 model provides the most persuasive explanations. The discrepancy from the original prediction is greatest when excluding the green agent, which carries the positive label. Despite the confusion regarding the nature of aggression depicted, the explanation correctly identify the aggressor as the green agent. While this information is not directly relevant to prediction accuracy, it is significant for understanding which agent might pose a threat to the *Ego Vehicle*'s safety.

The second example, presented in Figure 3b, differs from Figure 3a in two primary aspects. Firstly, the scene is more crowded, containing two additional agents compared to Figure 3a. This leads to the second, more complex difference: partial occlusion among these agents. For instance, in the last frame, the green agent, partially occluded by the purple agent, in turn obscures the red and blue agents. This implies a significant challenge for the masking approach, which inherently relies on occlusion. The model exhibits high values for the *LaneChanging* label. However, the explanation shows a steep drop, when masking the green agent. Indicating, that this agent could be performing the aggressive action. The aggressive behavior of the purple agent is not adequately represented in the explanations, highlighting a limitation of the masking method.

Lastly, Figure 3c presents arguably the most challenging example discussed here. Not only does it involve 11 agents -a significantly higher number than previous examples— but it also contains two distinct actions. The red agent, obscured in the final frame, is *Cutting* another agent, while the bright green agent is *OverTaking*. Unfortunately, *OadTR_v4* struggles to correctly identify *Cutting* in the scene, thus highlighting the difficulties arising from the dependence on the original prediction. Considering *OverTaking*, the model attributes the responsibility wrongly. Even though the light green agent is overtaking, the largest difference is observed when the overtaken agent is occluded.

The explanations for *Cutting* fail to correctly attribute the action to the agent. *OadTR_v4* reports the largest difference when masking the orange agent. Despite the red agent performing the aggressive action.

Additional examples generated with the other models are included in the Figures 8, 9 and 10 in Appendix E. As is emergent from reviewing these figures, the explanations for the different models are fairly similar to each other. Hence, they face similar problems like a degradation in goodness of explanation as the number of agents increases, a difficulty to correctly identify the aggressor and a dependence on the initial classification scores.

6 Concluding Remarks

In the quest for road safety, understanding aggressive driving behavior is essential. This research offers a comprehensive exploration into such behaviors by leveraging attention-based models, feature extraction, and the METEOR driving dataset. Two significant innovations were introduced: the integration of the OadTR and Colar models and the generation of salient cues. Despite the hybrid model not outperforming individual models in global metrics, there was a marked improvement for lesser-represented labels, underscoring the potential of the combined approach. Model interpretability was emphasized, producing transparent and efficient explanations, particularly crucial for real-time applications like autonomous vehicles. The pivotal role of the METEOR dataset in this study emphasizes the need for comprehensive data sources to gain a deeper understanding of driving behaviors.

Moving forward, the journey to understand aggressive driving behavior through attention-based models continues. Future studies would benefit from a specialized backbone for targeted feature extraction, moving beyond pre-trained models to capture aggressive driving nuances. Improved model interpretability, especially in legal contexts, remains paramount. Utilizing diverse datasets can pave the way for more general models. The growing capabilities of these models also bring to the forefront the necessity of real-time assessment tools and the need to address ethical and privacy concerns. Collaborations with stakeholders, including policymakers and automotive manufacturers, are vital for translating research insights into practical solutions for road safety.

A Additional Dataset Statistics

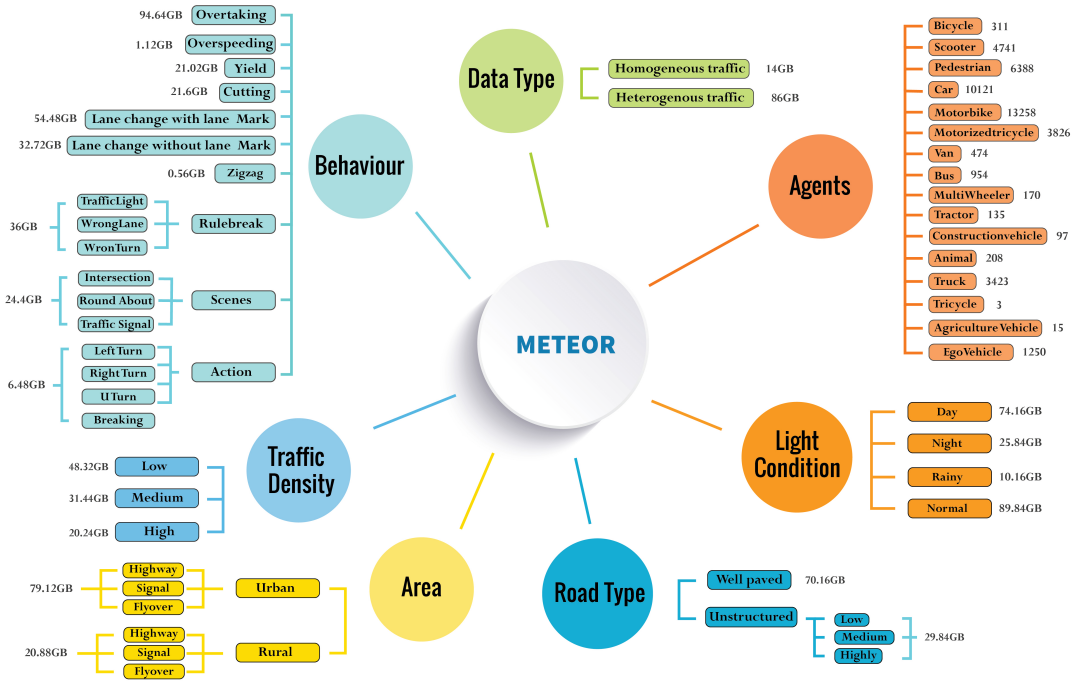


Fig. 4: Overview of Labels in METEOR Dataset [10]

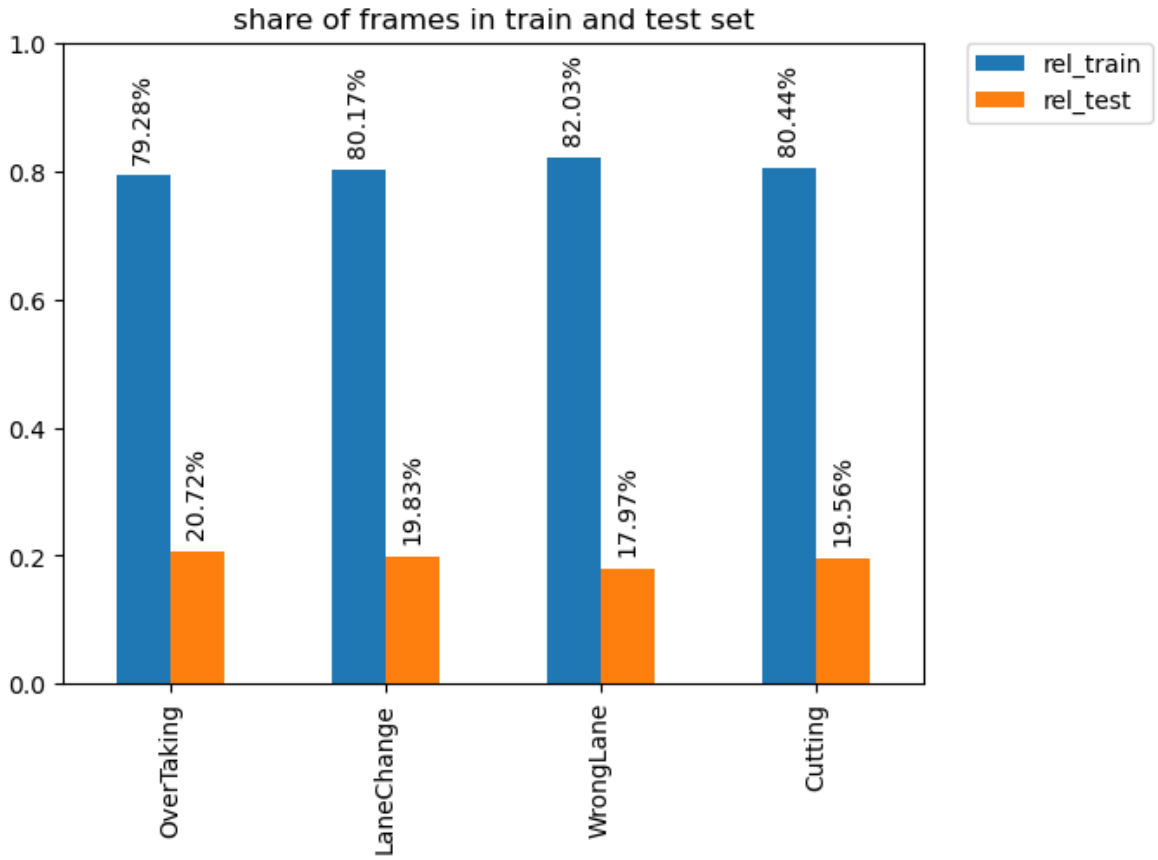


Fig. 5: Share of frames used for training and testing

Table 2: Number of Frames where a certain actor is annotated as showing a specific aggressive behaviour.

Categories and Actors that are part of the final configuration are marked with a *

	OverTaking*	OverSpeeding	LaneChanging(m)*	LaneChanging*	TrafficLight	WrongLane*	WrongTurn	Cutting*
EgoVehicle	227225	9	165866	37371	32	23733	187	4595
MotorizedTricycle	66556	209	9974	3800	7	4985	2	1233
Car*	304322	87	88686	28558	208	8515	24	5534
Van	6501	0	929	820	0	268	0	0
MotorBike*	319702	285	14271	6917	727	40863	115	4994
Pedestrian	45	0	0	0	0	377	4	5180
Bicycle	852	0	18	0	0	2762	1	142
MultiWheeler	0	0	0	0	0	332	0	0
Truck	63728	89	18889	4376	0	4532	0	3868
Scooter*	130465	7	7237	2897	220	15440	281	2289
Tractor	57	0	0	0	0	2598	0	0
Bus	33756	10	9263	3296	0	4852	0	817
ConstructionVehicle	802	0	245	0	0	0	0	21
Animal	0	0	0	0	0	0	0	390
AgricultureVehicle	0	0	0	0	0	0	0	0
Pedestrain	0	0	0	0	0	0	0	0
Tricycle	0	0	0	0	0	0	0	0

B Model Visualizations

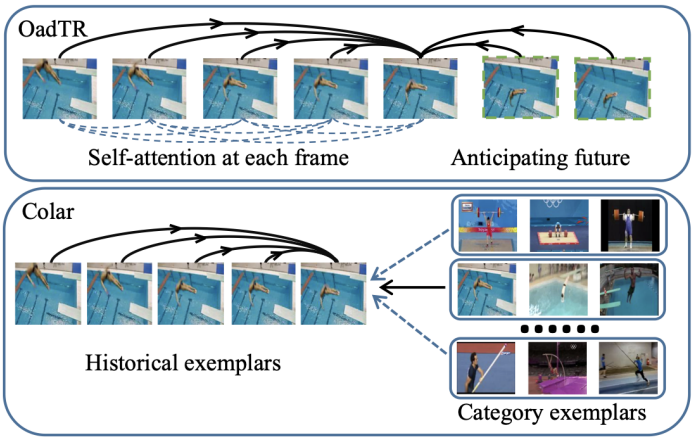


Fig. 6: Illustration of the conceptual distinction between OadTR [57] and Colar [61]. Adapted from the original representation in [61].

C Model Descriptions

Table 3: List of Colar hyperparameters and their default values

Argument	**Default Value**
exp_name	'ColarMETEOR'
data_root	'/workspace/pvc-meteor/features/features_i3d.pkl'
dataset_file	'/workspace/pvc-meteor/features/METEOR_info.json'
kmean	'/workspace/pvc-meteor/features/colar/exemplar.pickle'
checkpoint	'./checkpoint/THUMOS-TSN-Kinetics.pth'
seed	20
lr	3e-4
weight_decay	1e-4
cuda_id	0
lr_drop	1
input_size	2048
enc_layers	64
numclass	5
batch_size	512
overlap	1
num_workers	8
start_epoch	1
epochs	20
output_dir	'checkpoint'
clip_max_norm	1.0
feature_type	'METEOR'
command	'kinetics'

Table 4: List of OadTR hyperparameters and their default values

Argument	**Default Value**
lr	1e-4
batch_size	512
weight_decay	1e-4
epochs	60
resize_feature	False
lr_drop	1
lr_drop_size	0.5
clip_max_norm	1.0
dataparallel	None
removelog	None
use_flow	False
version	'v3'
query_num	8
decoder_layers	4
decoder_embedding_dim	1024
decoder_embedding_dim_out	1024
decoder_attn_dropout_rate	0.4
decoder_num_heads	4
classification_pred_loss_coef	0.5
enc_layers	64
lr_backbone	1e-4
feature	'3D_Resnet'
dim_feature	2048
patch_dim	1
embedding_dim	1024
num_heads	8
num_layers	3
attn_dropout_rate	0.4
positional_encoding_type	'learned'
hidden_dim	512
dropout_rate	0.4
numclass	22
classification_x_loss_coef	0.3
classification_h_loss_coef	1
similar_loss_coef	0.1
margin	1.0
weighted_loss	'True'
weight_values	0
dataset_file	'/workspace/pvc-meteor/features/METEOR_info.json'
frozen_weights	None
thumos_data_path	'/home/dancer/mycode/Temporal.Online.Detection/ Online.TRN.Pytorch/preprocess/'
thumos_anno_path	'data/thumos_{}_anno.pickle'
remove_difficult	None
device	'cuda'
binary_label	False
output_dir	'models'
seed	20
resume	"
start_epoch	1
eval	None
num_workers	8
use_frequent	'True'
use_infrequent	'False'
pickle_file_name	'METEOR.pickle'
world_size	1
dist_url	'tcp://127.0.0.1:12342'

D Additional Experiment Results

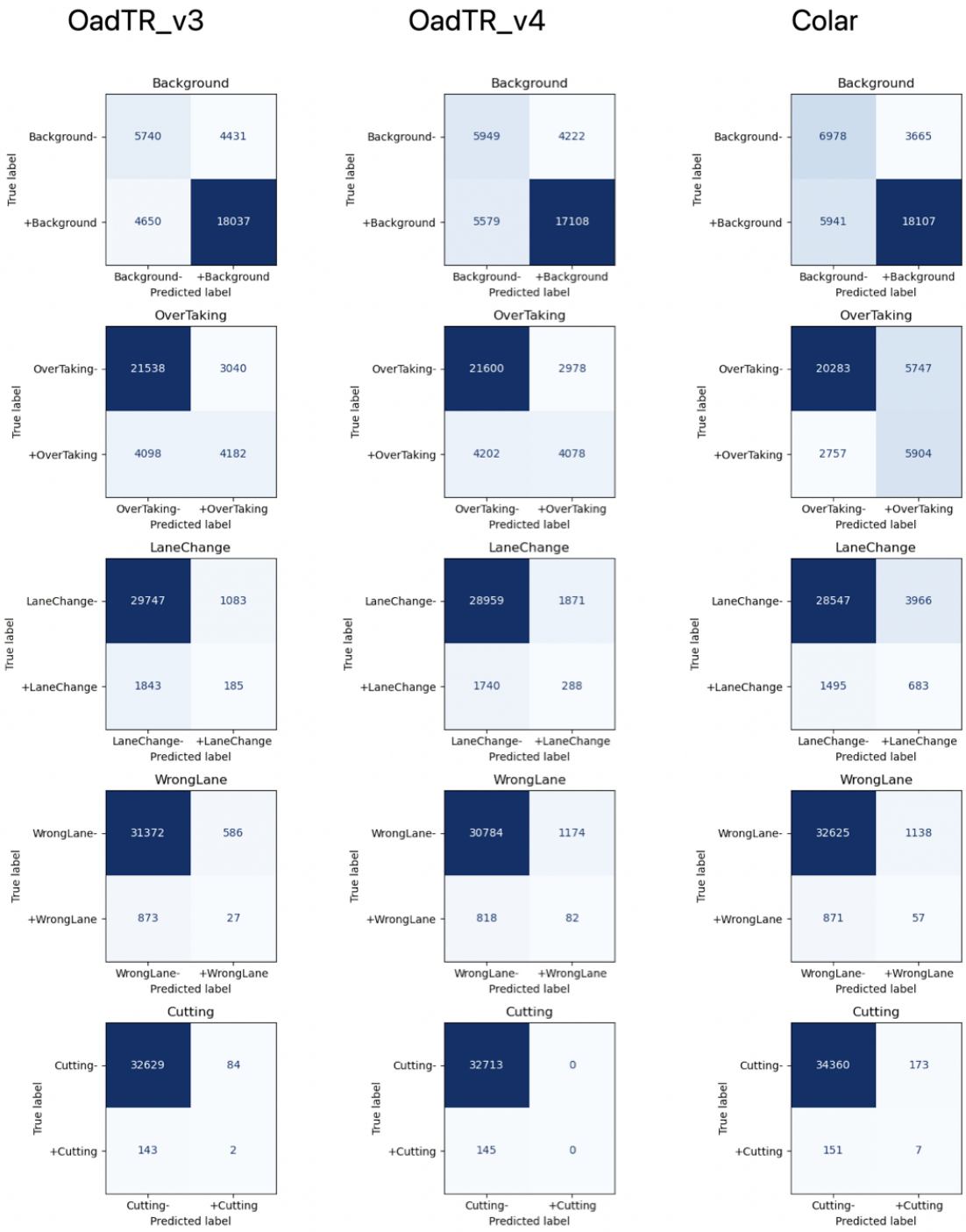


Fig. 7: Multilabel Confusion Matrices for Different Model Architectures

Table 5: Measures for different model architectures, specific for each label

	category	F1	recall	precision	mAP
OadTR_v3	Background	0.799	0.795	0.803	0.858
	OverTaking	0.540	0.505	0.579	0.542
	LaneChange	0.112	0.091	0.146	0.107
	WrongLane	0.036	0.030	0.044	0.046
	Cutting	0.017	0.014	0.023	0.012
OadTR_v4	Background	0.777	0.754	0.802	0.849
	OverTaking	0.532	0.493	0.578	0.546
	LaneChange	0.138	0.142	0.133	0.1
	WrongLane	0.076	0.091	0.065	0.041
	Cutting	0.000	0.000	0.000	0.077
Colar	Background	0.790	0.753	0.832	0.875
	OverTaking	0.581	0.682	0.507	0.574
	LaneChange	0.200	0.314	0.147	0.125
	WrongLane	0.054	0.061	0.048	0.045
	Cutting	0.041	0.044	0.039	0.029

Table 6: Classification performance of models with different numbers of decoder layers.

Metric	**3**	**4**	**5**	**6**
ROC_AUC	0.690	**0.697**	0.686	0.691
F1	0.676	**0.679**	0.644	0.666
Precision	0.676	**0.678**	0.673	0.675
Recall	0.699	**0.707**	0.664	0.691
mAP	0.303	**0.307**	0.296	0.301

Table 7: Impact of number of encoder layers on classification performance.

Metric	**2**	**3**	**4**	**5**
ROC_AUC	0.695	0.693	0.698	**0.700**
F1	**0.679**	0.671	0.675	0.651
Precision	0.676	0.678	0.679	**0.685**
Recall	**0.709**	0.693	0.700	0.671
mAP	**0.316**	0.305	0.305	0.302

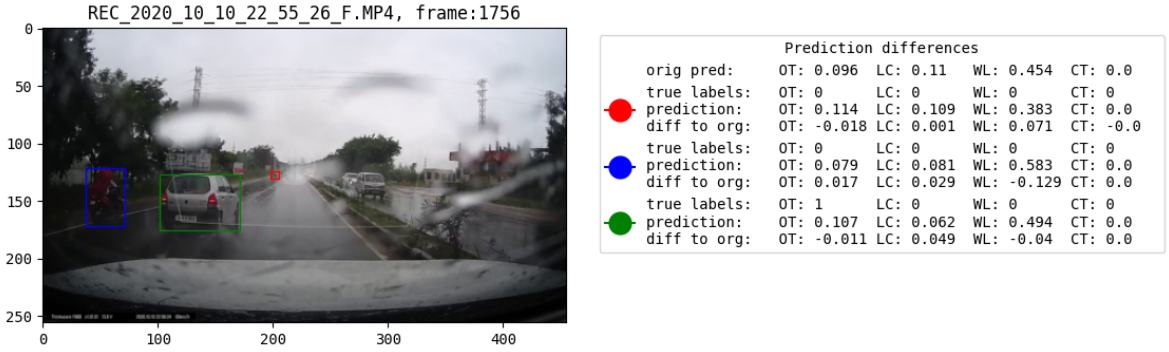
Table 8: Impact of dropout probability on classification performance.

Metric	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**
ROC_AUC	**0.702**	0.698	0.694	0.690	0.701
F1	0.662	0.666	0.667	0.676	**0.682**
Precision	**0.684**	0.681	0.676	0.676	0.681
Recall	0.681	0.687	0.689	0.699	**0.711**
mAP	**0.312**	0.309	0.311	0.303	0.309

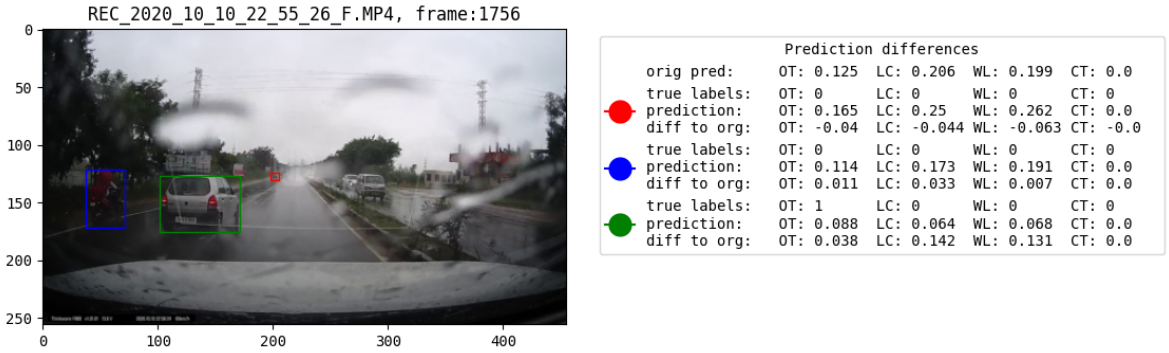
Table 9: Impact of hidden dimension size on classification performance.

Metric	**128**	**256**	**512**	**1024**
ROC_AUC	0.680	0.682	0.690	**0.691**
F1	0.657	0.658	**0.676**	0.674
Precision	0.668	0.670	**0.676**	0.674
Recall	0.679	0.677	**0.699**	**0.699**
mAP	**0.306**	0.305	0.303	0.305

E Visual Explanations

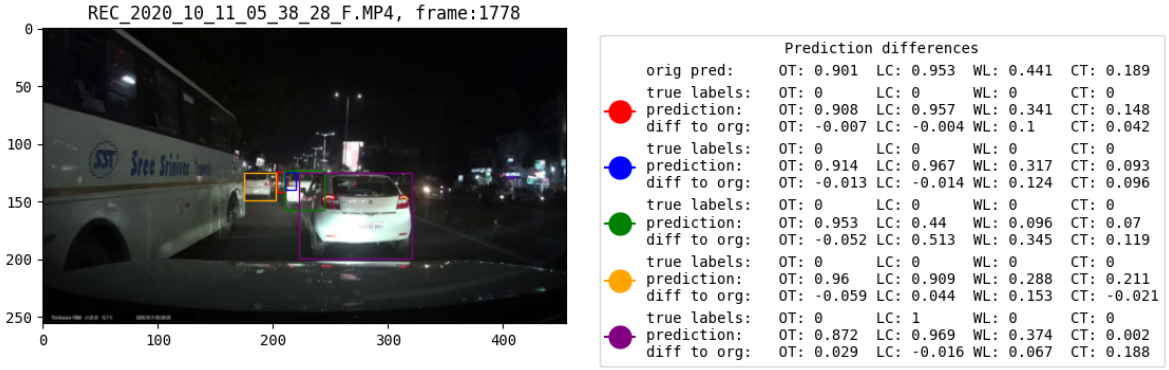


(a) Colar

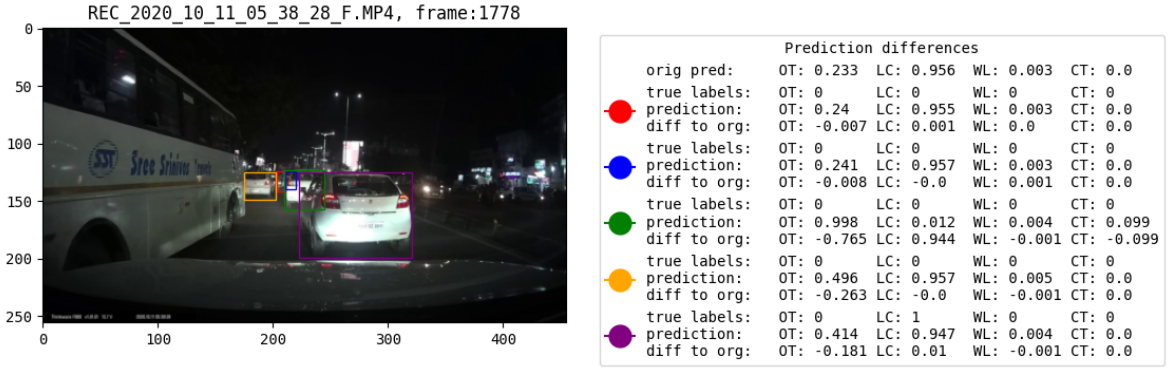


(b) OadTR_v3

Fig. 8: Explanations for action category *OverTaking*

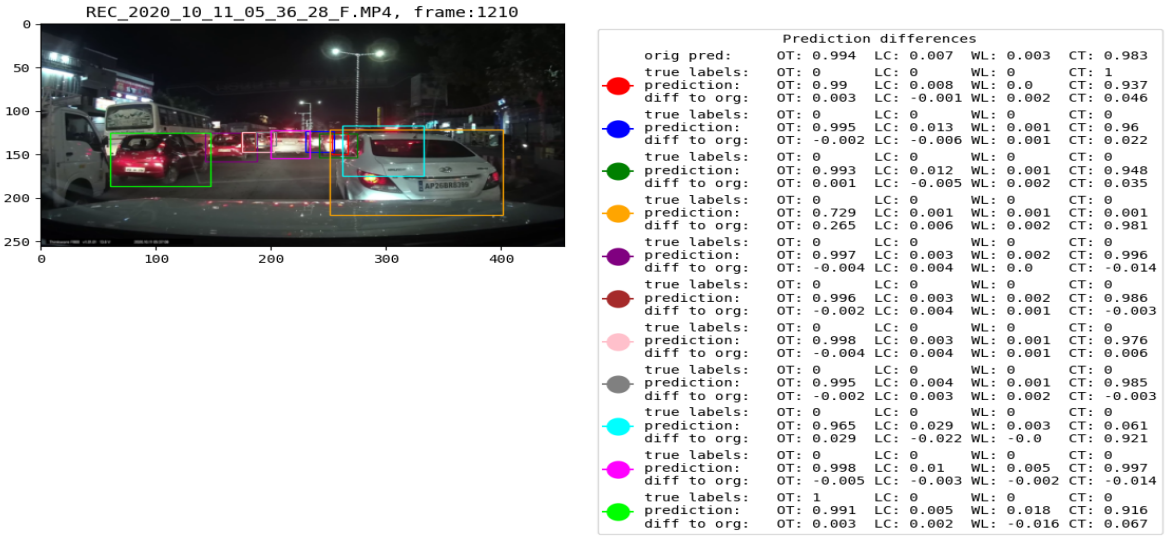


(a) Colar

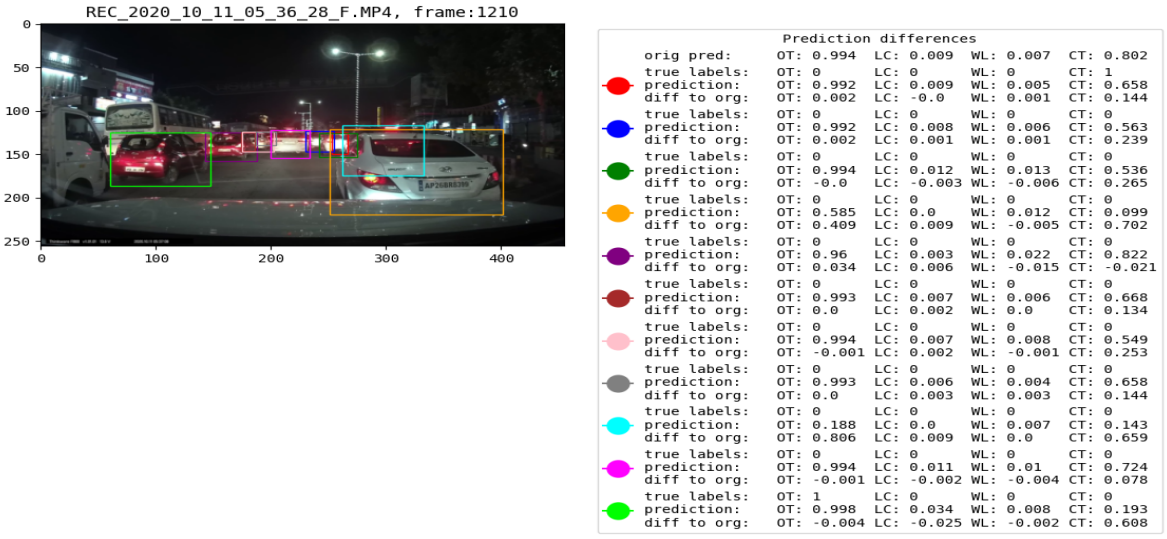


(b) OadTR_v3

Fig. 9: Explanations for action category *LaneChanging*



(a) Colar



(b) OadTR_v3

Fig. 10: Explanations for Explanations for action category *Cutting*

References

1. Scikit-learn: Machine learning in Python, https://scikit-learn.org/stable/modules/classes.html#module-sklearn.metrics, accessed: 2023-06-11
2. Aechtner, J., Cabrera, L., Katwal, D., Onghena, P., Valenzuela, D.P., Wilbik, A.: Comparing user perception of explanations developed with xai methods. In: 2022 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE). pp. 1–7. IEEE (2022)
3. Aljaafreh, A., Alshabatat, N., Al-Din, M.S.N.: Driving style recognition using fuzzy logic. In: 2012 IEEE International Conference on Vehicular Electronics and Safety (ICVES 2012). pp. 460–463. IEEE (2012)
4. Arrieta, A.B., Díaz-Rodríguez, N., Del Ser, J., Bennetot, A., Tabik, S., Barbado, A., García, S., Gil-López, S., Molina, D., Benjamins, R., et al.: Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. Information fusion **58**, 82–115 (2020)
5. Bach, S., Binder, A., Montavon, G., Klauschen, F., Müller, K.R., Samek, W.: On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. PloS one **10**(7), e0130140 (2015)
6. Baptista-Ríos, M., López-Sastre, R.J., Caba Heilbron, F., Van Gemert, J.C., Acevedo-Rodríguez, F.J., Maldonado-Bascón, S.: Rethinking online action detection in untrimmed videos: A novel online evaluation protocol. IEEE Access **8**, 5139–5146 (2020). https://doi.org/10.1109/ACCESS.2019.2961789
7. Carreira, J., Noland, E., Hillier, C., Zisserman, A.: A short note on the kinetics-700 human action dataset. CoRR **abs/1907.06987** (2019), http://arxiv.org/abs/1907.06987
8. Castelvecchi, D.: Can we open the black box of ai? Nature News **538**(7623), 20 (2016)
9. Chandra, R., Bhattacharya, U., Mittal, T., Bera, A., Manocha, D.: Cmetric: A driving behavior measure using centrality functions. In: 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 2035–2042. IEEE (2020)
10. Chandra, R., Mahajan, M., Kala, R., Palugulla, R., Naidu, C., Jain, A., Manocha, D.: Meteor: A massive dense & heterogeneous behavior dataset for autonomous driving. arXiv preprint arXiv:2109.07648 (2021)
11. Chefer, H., Gur, S., Wolf, L.: Transformer interpretability beyond attention visualization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 782–791 (2021)
12. Clark, A.: Whatever next? predictive brains, situated agents, and the future of cognitive science. Behavioral and brain sciences **36**(3), 181–204 (2013)
13. De Geest, R., Gavves, E., Ghodrati, A., Li, Z., Snoek, C., Tuytelaars, T.: Online action detection. In: Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part V 14. pp. 269–284. Springer (2016)
14. Deffenbacher, J.L., Oetting, E.R., Lynch, R.S.: Development of a driving anger scale. Psychological reports **74**(1), 83–91 (1994)
15. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255. Ieee (2009)
16. Donahue, J., Anne Hendricks, L., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., Darrell, T.: Long-term recurrent convolutional networks for visual recognition and description. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2015)
17. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929 (2020)
18. for Europe, U.N.E.C.: Aggressive driving behaviour: Background paper (2018), https://unece.org/aggressive-driving-behaviour-background-paper, accessed on [January 12, 2023]
19. Fong, R., Patrick, M., Vedaldi, A.: Understanding deep networks via extremal perturbations and smooth masks. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 2950–2958 (2019)
20. Gao, J., Yang, Z., Nevatia, R.: Red: Reinforced encoder-decoder networks for action anticipation. arXiv preprint arXiv:1707.04818 (2017)
21. Ghosh, R.: Deep learning for videos: A 2018 guide to action recognition. URL: http://blog. qure. ai/notes/deep-learning-for-videos-actionrecognition-review (visited on 13/05/2020) (2018)
22. Girdhar, R., Carreira, J., Doersch, C., Zisserman, A.: Video action transformer network. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 244–253 (2019)
23. Goebel, R., Chander, A., Holzinger, K., Lecue, F., Akata, Z., Stumpf, S., Kieseberg, P., Holzinger, A.: Explainable ai: the new 42? In: Machine Learning and Knowledge Extraction: Second IFIP TC 5, TC 8/WG 8.4, 8.9, TC 12/WG 12.9 International Cross-Domain Conference, CD-MAKE 2018, Hamburg, Germany, August 27–30, 2018, Proceedings 2. pp. 295–303. Springer (2018)
24. Gu, J., Yang, Y., Tresp, V.: Understanding individual decisions of cnns via contrastive backpropagation. In: Computer Vision–ACCV 2018: 14th Asian Conference on Computer Vision, Perth, Australia, December 2–6, 2018, Revised Selected Papers, Part III 14. pp. 119–134. Springer (2019)
25. Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F., Pedreschi, D.: A survey of methods for explaining black box models. ACM computing surveys (CSUR) **51**(5), 1–42 (2018)

26. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
27. Iwana, B.K., Kuroki, R., Uchida, S.: Explaining convolutional neural networks using softmax gradient layer-wise relevance propagation. In: 2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW). pp. 4176–4185. IEEE (2019)
28. Kay, W., Carreira, J., Simonyan, K., Zhang, B., Hillier, C., Vijayanarasimhan, S., Viola, F., Green, T., Back, T., Natsev, P., et al.: The kinetics human action video dataset. arXiv preprint arXiv:1705.06950 (2017)
29. Khan, S., Naseer, M., Hayat, M., Zamir, S.W., Khan, F.S., Shah, M.: Transformers in vision: A survey. ACM computing surveys (CSUR) **54**(10s), 1–41 (2022)
30. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems. pp. 1097–1105 (2012)
31. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. nature **521**(7553), 436–444 (2015)
32. LeCun, Y., Boser, B., Denker, J., Henderson, D., Howard, R., Hubbard, W., Jackel, L.: Handwritten digit recognition with a back-propagation network. Advances in neural information processing systems **2** (1989)
33. Leeming, J., Mackay, G., Pole, K., Fitzgerald, P.: Road accidents: prevent of punish? (1969)
34. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: Hierarchical vision transformer using shifted windows. CoRR **abs/2103.14030** (2021), https://arxiv.org/abs/2103.14030
35. Lowe, D.G.: Object recognition from local scale-invariant features. In: Proceedings of the seventh IEEE international conference on computer vision. vol. 2, pp. 1150–1157. Ieee (1999)
36. Lundberg, S.M., Lee, S.I.: A unified approach to interpreting model predictions. Advances in neural information processing systems **30** (2017)
37. Mavrogiannis, A., Chandra, R., Manocha, D.: B-gap: Behavior-guided action prediction and navigation for autonomous driving. arXiv preprint arXiv:2011.03748 (2020)
38. Montavon, G., Lapuschkin, S., Binder, A., Samek, W., Müller, K.R.: Explaining nonlinear classification decisions with deep taylor decomposition. Pattern recognition **65**, 211–222 (2017)
39. PyTorch: Pytorch vision models (2022), https://pytorch.org/docs/stable/generated/torch.nn.BCEWithLogitsLoss.html
40. Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Grad-cam: Visual explanations from deep networks via gradient-based localization. In: Proceedings of the IEEE international conference on computer vision. pp. 618–626 (2017)
41. Shrikumar, A., Greenside, P., Shcherbina, A., Kundaje, A.: Not just a black box: Learning important features through propagating activation differences. arXiv preprint arXiv:1605.01713 (2016)
42. Simonyan, K., Zisserman, A.: Two-stream convolutional networks for action recognition in videos. Advances in neural information processing systems **27** (2014)
43. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
44. Smilkov, D., Thorat, N., Kim, B., Viégas, F., Wattenberg, M.: Smoothgrad: removing noise by adding noise. arXiv preprint arXiv:1706.03825 (2017)
45. Soomro, K., Zamir, A.R., Shah, M.: Ucf101: A dataset of 101 human actions classes from videos in the wild. arXiv preprint arXiv:1212.0402 (2012)
46. Sundararajan, M., Taly, A., Yan, Q.: Axiomatic attribution for deep networks. In: International conference on machine learning. pp. 3319–3328. PMLR (2017)
47. Talebloo, F., Mohammed, E.A., Far, B.: Deep learning approach for aggressive driving behaviour detection (2021)
48. Tasca, L.: A review of the literature on aggressive driving research (2000)
49. Tran, D., Bourdev, L., Fergus, R., Torresani, L., Paluri, M.: Learning spatiotemporal features with 3d convolutional networks. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV) (December 2015)
50. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems **30** (2017)
51. Voulodimos, A., Doulamis, N., Doulamis, A., Protopapadakis, E., et al.: Deep learning for computer vision: A brief review. Computational intelligence and neuroscience **2018** (2018)
52. Wang, H., Kläser, A., Schmid, C., Liu, C.L.: Dense trajectories and motion boundary descriptors for action recognition. International journal of computer vision **103**, 60–79 (2013)
53. Wang, H., Schmid, C.: Action recognition with improved trajectories. In: Proceedings of the IEEE international conference on computer vision. pp. 3551–3558 (2013)
54. Wang, L., Li, Z., Li, M.: Impact of personality traits on driving behavior: A systematic review. Transportation Research Part F: Traffic Psychology and Behaviour **55**, 342–358 (2018)
55. Wang, L., Xiong, Y., Wang, Z., Qiao, Y., Lin, D., Tang, X., Van Gool, L.: Temporal segment networks: Towards good practices for deep action recognition. In: European conference on computer vision. pp. 20–36. Springer (2016)

56. Wang, W., Xi, J., Chong, A., Li, L.: Driving style classification using a semisupervised support vector machine. IEEE Transactions on Human-Machine Systems **47**(5), 650–660 (2017)
57. Wang, X., Zhang, S., Qing, Z., Shao, Y., Zuo, Z., Gao, C., Sang, N.: Oadtr: Online action detection with transformers (2021). https://doi.org/10.48550/ARXIV.2106.11149, https://arxiv.org/abs/2106.11149
58. Xiao, J., Ma, D., Yamane, S.: Optimizing 3d convolution kernels on stereo matching for resource efficient computations. Sensors **21**(20), 6808 (2021)
59. Xu, M., Gao, M., Chen, Y.T., Davis, L.S., Crandall, D.J.: Temporal recurrent networks for online action detection. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (October 2019)
60. Xu, W., Wang, J., Fu, T., Gong, H., Sobhani, A.: Aggressive driving behavior prediction considering driver's intention based on multivariate-temporal feature data. Accident Analysis & Prevention **164**, 106477 (2022)
61. Yang, L., Han, J., Zhang, D.: Colar: Effective and efficient online action detection by consulting exemplars. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3160–3169 (2022)
62. Zhang, Y., Li, X., Liu, C., Shuai, B., Zhu, Y., Brattoli, B., Chen, H., Marsic, I., Tighe, J.: Vidtr: Video transformer without convolutions. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 13577–13587 (2021)
63. Zhang, Y., Li, B., Fang, H., Meng, Q.: Current advances on deep learning-based human action recognition from videos: a survey. In: 2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA). pp. 304–311. IEEE (2021)