Driver Activity Detection

Shuvam Aich
Department of Computer Science
University of Stuttgart
Stuttgart, Germany
st187673@stud.uni-stuttgart.de

Mugdha Asgekar
Department of Computer Science
University of Stuttgart
Stuttgart, Germany
--@stud.uni-stuttgart.de

Prateek Chaturvedi
Department of Computer Science
University of Stuttgart
Stuttgart, Germany
--@stud.uni-stuttgart.de

Mahdi Jafarkhani
Department of Computer Science
University of Stuttgart
Stuttgart, Germany
st186851@stud.uni-stuttgart.de

Abstract

Driver activity detection in videos presents unique challenges due to the temporal dependencies inherent in video data. This report addresses these challenges by employing a sliding window approach to analyze sequences of frames, which helps to preserve context and improve detection performance. This method uses a fixed-size window that moves across the video, capturing localized segments. Each window is then processed using a pre-trained I3D model to predict the activity. Model performance is evaluated using metrics such as the midpoint hit criteria and Intersection over Union (IoU). The I3D model, a deep learning architecture that extends 2D convolutional neural networks into 3D, is used to capture spatiotemporal features, achieving an average accuracy of 82.26%. The study also explores advanced model architectures like transformers, temporal CNNs, and recurrent neural networks (RNNs) to further enhance accuracy. Our code and experiment results are available on the GitHub.

1 Introduction

Activity detection in videos is a challenging problem because videos contain temporal dependencies where activities evolve over time. Unlike static images, where an object classification model can predict a single label per frame, videos require models to understand the progression of actions. A direct classification approach, where each frame is independently classified, often fails to capture contextual information and temporal dependencies, leading to inaccurate predictions. To solve this problem, we employed a sliding window approach, which makes it possible to analyze a sequence of frames together rather than in isolation. The key idea behind the sliding window approach is to use a fixed-size window that moves over the video, capturing a localized segment at each step. This enables better context preservation and helps improve detection performance. A video typically consists of thousands of frames, and activities often span across multiple consecutive frames rather than occurring in a single instance. Using a window of frames allows the model to capture transitions effectively. The sliding window approach retains context from nearby frames, helping the model understand activity progression. Grouping frames into a window allows the model to process fewer but more meaningful segments, which is more efficient.

Machine Perception and Learning Course Project Report
Winter Semester 2024-2025, University of Stuttgart, Germany

2 Dataset Overview

The Drive&Act dataset [MRH+19] addresses the need for domain-specific action recognition benchmarks, featuring twelve hours of drivers engaged in secondary tasks during both manual and automated driving.**??** depicts an overview of the dataset. The dataset includes 9.6 million frames, which is more than four times greater than any other previously published dataset for driver action recognition. The activities are labeled with 83 classes. The dataset is multi-modal, incorporating color, depth, infrared, and 3D body pose data. Six synchronized camera views cover the vehicle cabin to address the issue of limited body visibility. The dataset includes hierarchical activity labels on three levels of abstraction, as well as context annotations.

2.1 Data Collection

Data was collected in a static driving simulator to ensure safety, with vehicle surroundings simulated and projected on multiple screens around a converted Audi A3. The simulation software used was SILAB. Each driver completed twelve different tasks per session, with instructions given on a mounted tablet. Tasks included manual driving, switching to autonomous mode, and responding to unexpected take-over requests. The specific execution of tasks was left to the driver to encourage diverse behavior. Fifteen participants (4 female, 11 male) with varied body types, driving styles, and familiarity with automation systems were recorded twice, resulting in 30 driving sessions averaging 24 minutes each.

2.2 Recorded Data Streams

The dataset includes video data from multiple views and modalities, 3D body and head pose, and features capturing interactions with the car interior. Five near-infrared cameras and a Microsoft Kinect for XBox One were used to capture color, infrared, and depth data. The sensor interfaces were calibrated and synchronized using ROS (Robot Operating System) with global timestamps. OpenPose was used to determine the 3D upper body skeleton with 13 joints, triangulated from 2D poses from three frontal views (right-top, front-top, left-top). Missing joints were filled using interpolation. OpenFace was also used to obtain the 3D head pose of the driver from all views except the back camera, selecting the most frontal view for each frame. Car-interior features are provided based on 3D primitives, depicting interactions with storage spaces and car controls. Video frames were manually labeled by annotators on three levels of abstraction, resulting in 83 action classes. The annotation scheme includes high-level scenarios, fine-grained activities, and low-level atomic action units.

2.3 Data Splits

The dataset is divided into three splits based on the driver's identity, with data from ten subjects for training, two for validation, and three for testing to evaluate generalization to new drivers. Action segments are divided into 3-second chunks for the benchmark, with evaluation scripts provided for comparable results. The dataset's vocabulary was created based on a literature review of secondary tasks during driving, expert surveys, and analysis of accident reports and naturalistic car studies. The vocabulary covers eight areas: eating and drinking, clothing and accessories, working, entertainment, entering/exiting and car adjustment, body movement, object manipulation, and using vehicle-internal devices. The vocabulary comprises 83 activity labels across three levels of granularity, forming a complexity- and duration-based hierarchy. It also employs three levels of abstraction:

- **Scenarios/Tasks:** The first level consists of twelve tasks that subjects had to complete, which are either typical during manual driving or expected to become common with increasing automation.
- **Fine-grained Activities:** The second level breaks down scenarios into 34 concise categories that retain a clear semantic meaning.
- **Atomic Action Units:** The third level portrays basic driver interactions with the environment, detached from long-term semantic meaning and comprising action, object, and location triplets.

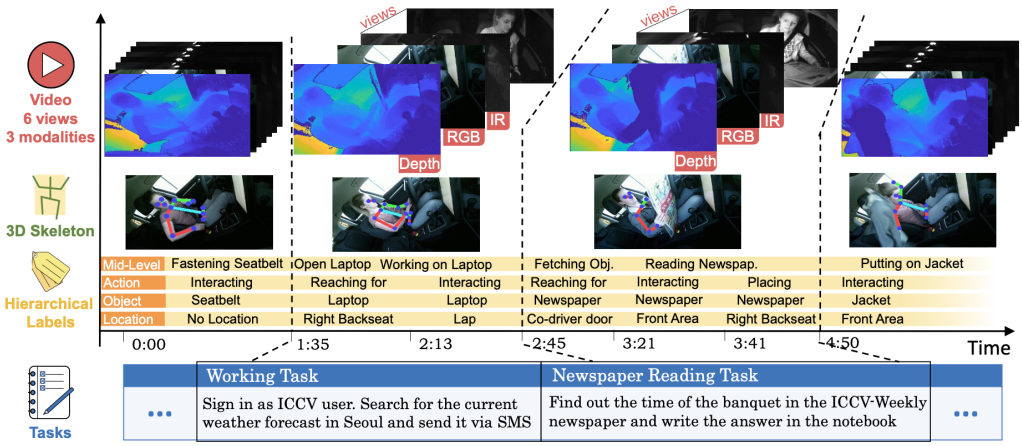

Figure 1: Overview of the Drive&Act dataset for driver behavior recognition. The dataset includes 3D skeletons in addition to frame-wise hierarchical labels of 9.6 Million frames captured by 6 different views and 3 modalities (RGB, IR and depth). [MRH+19]

3 Sliding Window

Activity detection in videos is a challenging problem due to the continuous and dynamic nature of video data. Unlike static images, where an object classification model can predict a single label per frame, videos contain temporal dependencies where activities evolve over time.

A direct classification approach—where each frame is independently classified—often fails to capture contextual information and temporal dependencies, leading to inaccurate predictions. To address this, we employ a sliding window approach, which allows us to analyze a sequence of frames together rather than in isolation.

The key idea behind this approach is that instead of treating the entire video as a single entity or classifying individual frames separately, we use a fixed-size window that moves over the video, capturing a localized segment at each step. This enables better context preservation and helps improve detection performance.

3.1 Motivation for Using Sliding Window

A video typically consists of thousands of frames, and activities often span across multiple consecutive frames rather than occurring in a single instance. The sliding window approach is designed to address the following challenges:

- **Temporal Overlap in Activities**: Many activities start and end gradually rather than having sharp boundaries. Using a window of frames allows the model to capture transitions effectively.
- **Context Awareness**: Classifying each frame in isolation lacks temporal context. The sliding window approach retains context from nearby frames, helping the model understand activity progression.
- **Efficiency and Performance Trade-off**: Instead of analyzing every individual frame separately (which would be computationally expensive), grouping frames into a window allows the model to process fewer but more meaningful segments.

3.2 Implementation

3.2.1 Window Size and Stride

- A **fixed-size window** (e.g. 8, 16, or 32 frames) is defined, which slides across the video.
- The **stride** determines how much the window moves forward after each step.

- If the stride is **small** (e.g., 1 frame), the overlap between windows increases, leading to **better temporal coverage but higher computational cost**.
- If the stride is **large** (e.g., 8 frames), fewer windows are processed, reducing computation but potentially **missing fine-grained transitions**.

3.2.2 Processing Each Window

- Each window of frames is sent as input to a pre-trained I3D model.
- The model infers on the window and predicts the most likely activity to occur in that segment.
- The predicted label is assigned to each frame of the window, along with the confidence scores (Figure 2).
- The activity with the highest confidence score is selected for each frame.
- As the window progresses, new predictions are generated, leading to a continuous activity timeline.

(a) Opening door outside

(b) Fastening Seat Belt

(c) Working on Laptop

(d) Using Multimedia Display

Figure 2: Grid for multiple predictions using Sliding Window.

4 Metrics

Evaluating activity detection models requires quantitative metrics that measure how well predictions align with ground truth activities. One of the objectives involved was to develop metrics for the sliding window approach to understand how accurate are the predictions. We employ two key metrics as mentioned in the next sections.

4.1 Midpoint Hit Criteria

To determine whether a prediction is correct, we used the **midpoint hit criteria**. For each ground truth activity segment, we computed its midpoint and checked if a predicted segment of the same activity class covered this midpoint. The evaluation steps are as follows:

- Compute the **midpoint** of each ground truth segment as:

$$\text{midpoint} = \frac{\text{frame\_start} + \text{frame\_end}}{2} \tag{1}$$

- A prediction is counted as a **true positive (TP)** if it covers the midpoint and belongs to the correct activity class.
- If no prediction matches the midpoint, it is considered a **false negative (FN)**.
- Any extra predicted segments that do not match a ground truth midpoint count as **false positives (FP)**.
- Precision and recall for each activity class are calculated as:

$$\text{Precision} = \frac{TP}{TP + FP} \times 100 \tag{2}$$

$$\text{Recall} = \frac{TP}{TP + FN} \times 100 \tag{3}$$

- Overall precision and recall are computed by aggregating across all activity classes.

4.2 Intersection over Union (IoU)

To measure how well the predicted activity windows overlap with the ground truth, we used **IoU**:

- **Intersection**: The overlapping frames between the ground truth and predicted segment.
- **Union**: The total number of frames covered by both segments.
- The IoU score for each ground truth segment is calculated as:

$$\text{IoU} = \frac{\text{Intersection}}{\text{Union}} \tag{4}$$

- The **mean IoU** is obtained by averaging the IoU scores across all ground truth segments.

4.3 Final Evaluation Results

The model outputs the following results:

- **Precision per class (%)**: Precision values for each activity class.
- **Recall per class (%)**: Recall values for each activity class.
- **Overall Precision**: Aggregated precision across all activities.
- **Overall Recall**: Aggregated recall across all activities.
- **Mean IoU**: The average IoU score across all activity segments.

These metrics provide a comprehensive evaluation of the accuracy and localization performance of the model.

5 Neural Detection Model

Manuel and Roitberg [MRH$^+$19] evaluated various models for fine-grained driver activity recognition using the DriveAndAct dataset. They tested different approaches, including CNN-based methods (like C3D, P3D ResNet, and I3D) and body pose-based methods. They found out that the Inflated 3D ConvNet (I3D), an extension of the Inception-v1 network with 3D convolutions, achieved the highest accuracy (69.57% on Validation, and 63.64% on test) among all tested models for recognizing fine-grained driver activities. I3D also outperformed body pose-based methods, which were less effective in classifying actions despite incorporating spatial and temporal streams. In atomic action unit classification, I3D performed best in recognizing actions (56.07%) and objects (56.15%), though body pose-based approaches were better at identifying locations (56.5%).Although for cross-view action recognition, I3D models struggled with domain shifts, performing significantly worse when tested on unseen views.

5.1 I3D Model

The Two-Stream Inflated 3D ConvNet (I3D) [CZ18] is a deep learning model designed for video action recognition. It extends 2D convolutional neural networks (CNNs) into 3D by "inflating" their filters and pooling kernels to operate in both spatial and temporal dimensions. This allows the model to effectively capture motion patterns in videos (Figure 3).

5.1.1 Key Features of I3D

- **3D Convolutions**: Unlike traditional 2D CNNs that process images independently, I3D uses 3D convolutional layers to extract spatiotemporal features from videos.
- **Inflation from 2D Networks**: The model is based on the Inception-v1 architecture, with its 2D filters expanded into 3D. This allows it to reuse ImageNet pre-trained weights, improving efficiency and performance.
- **Two-Stream Configuration**: I3D processes both RGB frames (spatial information) and Optical flow (motion information). These two streams are trained separately and their outputs are fused at the end.

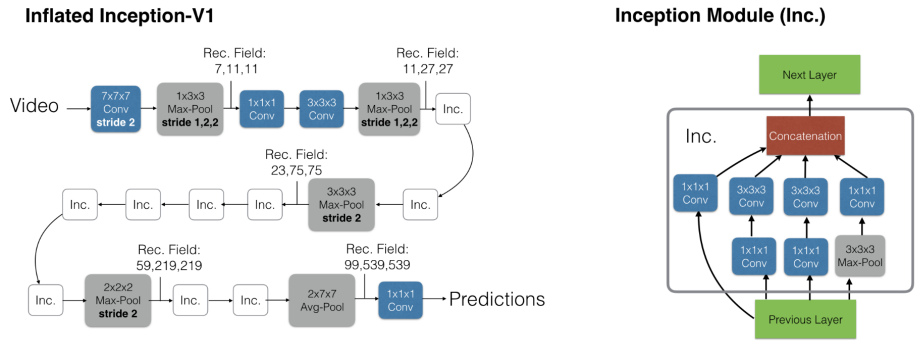

Figure 3: The Inflated Inception-V1 architecture (left) and its detailed inception submodule (right). The strides of convolution and pooling operators are 1 where not specified, and batch normalization layers, ReLu's and the softmax at the end are not shown. The theoretical sizes of receptive field sizes for a few layers in the network are provided in the format "time,x,y" – the units are frames and pixels. The predictions are obtained convolutionally in time and averaged.[CZ18]

5.2 I3D Model as baseline model

We utilized the I3D model for recognition in our project. We evaluated a pre-trained I3D model and performed inference on all videos in the Kinetic Color category. Figure 4 illustrates the model's accuracy on the DriveAndAct dataset. The dataset includes 15 viewpoints, each comprising two runs, except for viewpoint 9, which lacked annotation for the second run. The average accuracy is 82.26% for our baseline model. We noticed the main reason for misclassfying each frame to its action is either by:

- **Latency in Annotation**: Since human-labeled annotations (ground truths) lack a strict consensus on the exact start time of an action, some degree of misclassification is inevitable. In this context, misclassifying an action by at most 10 frames is considered acceptable or, in other words, irreducible.
- **Fast Switching Actions**: The model exhibited rapid transitions between actions, sometimes within less than 3 frames. This behavior was of particular interest to us, and we hypothesized that applying a neural network or transformer architecture on top of the I3D model could help reduce this misclassification.

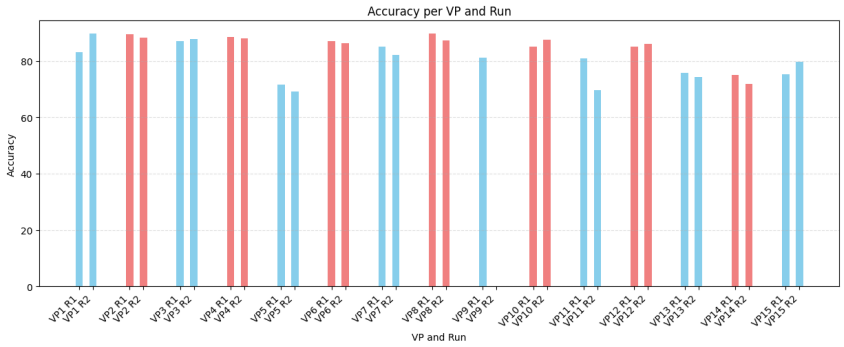

Figure 4: Accuracy for recognition task with I3D model on all viewpoints and runs on Kinetic Color category, DriveAndAct dataset

5.3 Extending I3D Model

5.3.1 Preprocessing

The input data utilized in this study was originally trained using the I3D model, which generated class probabilities as output. These probabilities were stored in .pt files for subsequent processing. To prepare the data for our detection model, we computed the argmax of the arrays containing the class probabilities, thereby converting them into discrete class labels. These labels were then used as input for training the detection model.

Due to computational constraints, we processed each video in the dataset individually. The training dataset consisted of 19,071 frames, with each frame represented by an array of 34 class probabilities. Frame-wise annotations were stored in corresponding CSV files, where missing annotations were denoted by a placeholder value of 40.

The preprocessing phase involved converting the class probabilities into discrete labels using the numpy argmax function. Additionally, batch processing techniques were employed to optimize memory usage, ensuring efficient handling of large datasets during training and inference. This approach facilitated the effective training of the detection model while mitigating computational limitations.

5.3.2 Model Enhancements and Challenges

To enhance accuracy, several advanced model architectures were explored, including transformers, temporal convolutional neural networks (CNNs), and recurrent neural networks (RNNs). A major challenge encountered was aligning the input and target dimensions for loss computation. Additionally, batch size mismatches and tensor shape inconsistencies required careful debugging and adjustments to the data pipeline. Some approaches achieved high training accuracies; however, test accuracy remained significantly lower.

To stabilize training and mitigate overfitting, techniques such as early stopping and batch normalization were implemented. Early stopping prevented unnecessary overtraining by halting training once validation performance plateaued; however, accuracy did not improve. Increasing the training data and performing hyperparameter tuning also had minimal impact. In some cases, modifications to the model architecture even reduced the accuracy of the training.

The transformer-based architecture uses an embedding layer, positional encoding, a stack of Transformer encoder layers, and a final linear layer for classification. If the model needs to consider the entire sequence (e.g., correlations across distant frames), the transformer should excel hence this approach. Training was faster because the sequence is processed as a whole rather than step by step and the self-attention mechanism captured long-range relationships more effectively. However, our training accuracy was very low indicating that our model was not able to learn the features properly.

The Long Short-Term Memory (LSTM) model was configured with one layer. This vanilla LSTM does not include attention mechanisms, which allow the model to selectively focus on specific time steps rather than relying only on the last hidden state. Stacked LSTMs, which use multiple

layers, can model more complex temporal hierarchies but come at a higher computational cost. This vanilla LSTM does not include attention mechanisms, which allow the model to selectively focus on specific time steps rather than relying only on the last hidden state. While the LSTM demonstrated promise in capturing temporal relationships within the data, tuning its hyperparameters (e.g., number of layers, hidden units, and learning rate) was critical for optimization. Despite these efforts, overfitting remained a challenge, particularly on smaller subsets of data, necessitating further exploration of regularization techniques. One model configuration achieved a high training accuracy but a very low test accuracy. The best-performing approach, with additional data manipulations, showed negligible improvement. This study does not claim the approach to be entirely correct, as it involved experimental exploration of both the data, the inputs and the models.

6 Conclusion

References

[CZ18] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset, 2018.

[MRH+19] Manuel Martin, Alina Roitberg, Monica Haurilet, Matthias Horne, Simon Reiß, Michael Voit, and Rainer Stiefelhagen. Driveact: A multi-modal dataset for fine-grained driver behavior recognition in autonomous vehicles. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2019.