6-16-2024

Naturalistic Driving Action Recognition

Andy Xiao

Antonio Fontan

SANTA CLARA UNIVERSITY

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Date: June 16, 2024

I HEREBY RECOMMEND THAT THE THESIS PREPARED UNDER MY SUPERVISION BY

Andy Xiao
Antonio Fontan

ENTITLED

Naturalistic Driving Action Recognition

BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

BACHELOR OF SCIENCE IN COMPUTER SCIENCE AND ENGINEERING

David C. Anastasiu

Digitally signed by David C. Anastasiu
Date: 2024.06.17 17:13:23 -04'00'

Thesis Advisor

Department Chair

Naturalistic Driving Action Recognition

by

Andy Xiao
Antonio Fontan

Submitted in partial fulfillment of the requirements
for the degree of
Bachelor of Science in Computer Science and Engineering
School of Engineering
Santa Clara University

Santa Clara, California
June 16, 2024

Naturalistic Driving Action Recognition

Andy Xiao
Antonio Fontan

Department of Computer Science and Engineering
Santa Clara University
June 16, 2024

ABSTRACT

ACKNOWLEDGMENTS

I would like to thank my advisor Dr. Anastasiu, for providing the support and resources for every step along the way needed for this project.

I would also like to thank Dr. Figuera, the chair of the Computer Science and Engineering Department at SCU, for her unwavering support in completing this project.

Table of Contents

Chapter 1

Introduction

1.1 Problem Statement

Driving is one of the most tasks people do day to day to get to places more efficiently, but with many people using the road every day, distracted driving can be a big problem. Grothlaw firm states that in the recent years, at least 9 out of 3287 deaths related to auto accidents per day are caused by distracted driving [1]. Distracted driving refers to things that drivers do that are not related to driving [2] This can include texting, eating, talking to passengers, tuning the radios, etc... that can get the driver's attention away from the road [2]. Many drivers do not realize the impact and importance of safety when they are multitasking when driving. Therefore, we will be participating in the Naturalistic Driving Action Recognition city track in the AI City Challenge, where we designed a recognition system utilizing Deep learning and computer vision that is used to identify distracted driving and to make drivers aware the impacts of distracted driving.

Chapter 2

Development Timeline

2.1 Tentative Timeline Plan

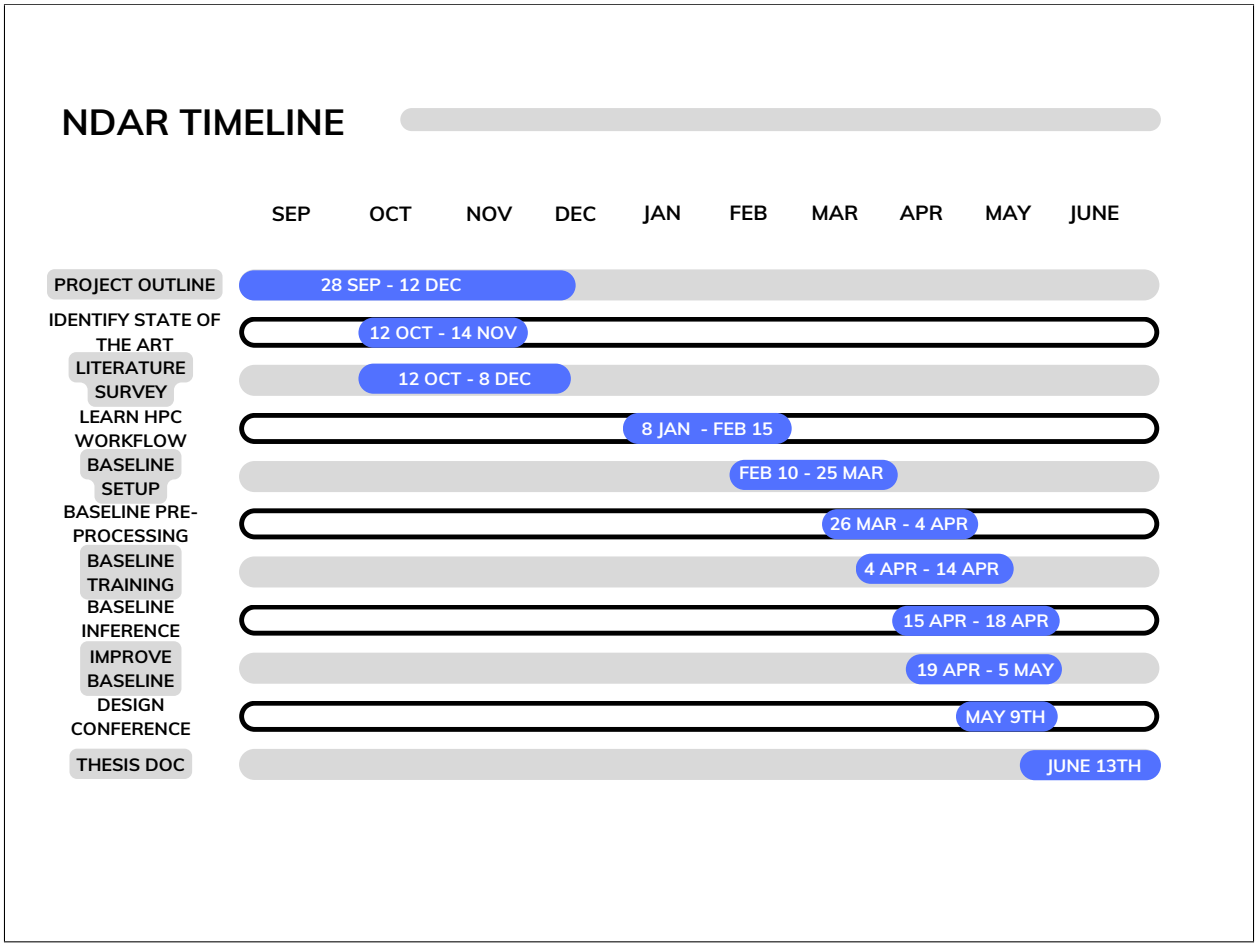

Figure 2.1: Project Timeline

In Figure 2.1 we listed our original tentative timeline of start and completion of each step of our project. We created the timeline based of the due dates and our anticipated finish time. The y-axis is the tasks and the x-axis is the month. As

shown in Figure 2.1 most of our time was spent on the line outline. We also have faced issues during our development that cause us to go behind our anticipated timeline. We discuss our struggle to get our workflow going later in the report.

Chapter 3

Project Risks

3.1 Risks

Risks we faced during the course of the project include compatibility with each software we used including Python, Pytorch, and Slowfast. With our HPC machines may be running different versions than the state of the art, there can be a lot of modifications we have to make to satisfy the capability. However, when we make the capability changes to the state of the art, there can be risks of completely messing up the state of the art and we have nothing to show. In one of the state of the art tests we did, we have experienced capability errors, but once we fixed the code, more errors arose in the test phase. Therefore, risks we have faced include capability errors or big modifications due to the capability error.

Chapter 4

Project Requirements

4.1 Functional Requirements

Evaluation

Given the 2023 A2 Data Set from the AI City Challenge, the system should be able to classify the timestamps of each activity class in the 10 videos within the dataset. After we preprocessed, trained, and inferenced the videos, we rewatched each of the videos and marked any errors we see. We then adjusted the weights based of the issues. Once we fixed the issue and we're confident the results are good, we exported the classification results including timestamps and distraction classes in the specified .txt format so it is compatible with the evaluation system. We then compared the new result with previous results to see the changes we made. We then sent the results to the challenge, in our case our advisor who has access to the challenge evaluation system. Our advisor then gave us the percentage accuracy rate.

4.2 Non-Functional Requirements

Performance

Shortening our training times allows us to iterate and improve our models more rapidly. With us shortening training times, it can also allow each training video to focus on a one action at a time reducing the chances of multiple actions being detected at the same time. This strategy also can also help mitigate time in case if there are issues with the training by not retraining large videos when it is midway through. This process can therefore save time based of less retraining and focus on one action at a time.

Accuracy

Accurate detection helps in effectively addressing distracted driving, thereby contributing to overall road safety and potentially saving lives. With more accurate detections, we eliminated more false negative results and ensure that any type of distracted driving is caught. With less accurate results, it is easier for people to get away and continue to drive distractively. Therefore, accurate results are an important factor as it addresses distracted driving more effectively.

Chapter 5

Literature Survey

Ahangari et al. [3] described that the distracted driving model can use the Bayesian Network and determine whether the driver is distracted or not based on the prediction it receives. With the Bayesian Network, distracted driving is determined based on two different steps with building a network structure as the first step and determining the probability based on the network structure as the second step. With the Bayesian Network method, the data has to be divided into both training and testing using k-fold validation formulas. In the "A Machine Learning Distracted Driving Prediction Method", determining if a driver is distracted, the system was able to detect distracted driving based on prediction and data collected is divided into training and testing using the k-fold validation method.

Li et al. [4] used a different approach. Instead of the Bayesian Network, they used three different approaches to combat distracted driving using AI/ML: MMDetection, Stacked Hourglass Network Detection, and ResNet50 . With MMDetection, the system will use a target detection model and will screen if the driver has anything that isn't matching with the image of not distracted driving . The second method: Stacked Hourglass Network Detection. With this method, it can use pointers to detect the facial and movement and create an image of what the user may be doing and it can reduce gradient instability etc... The third method: ResNet50 is the method that uses 7 layers with the first layer being the intake and perform the activation of the model, the second to fifth layer will be the one that repeats the convolution and then with the last layer be the one that calculate the output.

Streiffer et al. [5]'s approach uses both IoT and deep learning to solve the problem with distracted driving. To solve the problem, they used DarNet to determine distracted driving. With DarNet, they collect both IMU and image data and the server uses the bayes net classification algorithm to determine whether the driver is distracted or not. The image data is collected and intially processed from input to CNN and the IMU data is collected via RNN and both of the data are sent to the server and is processed via the bayes net classification which can better determine if the driver is distracted or not.

Similar to Streiffer et al., Shang, et al. [6]" has a similar approach in terms of classification for distracted driving distraction. With their approach of using CNN classification, the difference they have compared to Streiffer et al's

article is that instead of using the bayes net classification, they can identify and determine the facial points to get an visualization of the driver's face. This approach works by first getting the images followed by reducing the size of the image and increasing the face sizes for the purpose of facial detection, and then determine if the face is detected through the frames. This can collect the data based on the driver's eyes and mouth and can be used to train the dataset. After the dataset is created, they use CNN to classify the trained data and determine the based on results of the classification if the driver is distracted or not.

Wang et al. [7]'s approach also uses the CNN classification too. In their approach they first identify the inertial and image and collect and pre-process the data. During the pre-processing of data, they split the data into 80 percent for training and 20 percent for testing. After the training and testing data is determined, CNN comes in and extract the data. After CNN extracts the data, the extracted data is moved to the capsule. The capsule is powerful as it can effectively model both the data's hierarchical and structural relationship and it can include more information through vectors compared to neuros. There can be multiple capsules within one capsule if the vector value is larger than initially predicted. After the data passes through the capsules, the results generated will determine if the driver is distracted or not

Jeon et al. [8]'s approach to distracted driving uses the DDDA algorithm. How they work to troubleshoot distracted driving, the DDDA will detect the driver's behavior and alert them if they are driving distractedly or not. The DDDA algoritm works by taking in the input video of the driver, the DDDA algorithm will detect the driver's face via the face and landmark detection algorithms. Through the face detection they will determine the 4 following aspects: Blink estimation, Gaze estimation, Lower Head Detection, and Head pose estimation. They will calculate the blink of the drivers eyes using the areal and aspect ratio equations. For the gaze detection of the driver, they'll use the three lines and determine the direction of gaze based on the line. With the lower head detection, they will use the chin distance ratio formula to determine if the driver's head is below the safe driving threshold. The Head pose algorithm determines the angle of the face and is used similar formulas to as the gazing detection and are determined based on how far the driver's head is turned. After these data are collected, DDA will use the results to determine if the driver is distracted or not.

Qi and Li [9].'s approach used two types of deep learning networks. In their approach they used one and two stage networks for deep learning detection networks. With the one stage network such as YOLO using regression and faster speed and two stage networks able to generate classification and regression. With both one and two stage networks working together, stage one will extract the data and stage two can can do the classification and regression. With both stage one and stage two networks working together, it can have a improved algorithm of detecting distracted driving based on it's training data and testing data.

Yadawadkar et al. [10]'s idea was to have the SHRP2 NDS dataset where it contains training data of what distracted driving looks like. In the vehicle there is a system called the DAS which includes cameras, radar, GPS, vehicle network

information etc... The approach they have to detect distracted driving is to training set and classification algorithms in these following behaviors: drowsy, distracted, and attentive. Once one of these behaviors have been detected, data of the time when the behaviors take place is extracted and will be used to create epochs. Once the epochs are created and the PCA removed data redundancies, the epochs will be classified using features like CNN for videos and LSTM for time-series. After the epochs are classified, the Bayes algorithm will be used to determine the accuracy for both the CNN and LSTM classification algorithms. In their experiment, it turns out that CNN has an higher accuracy at 70 percent compared to LSTM which is 54 percent accurate

In Ezzouhri et al. [11]'s method, they determine if the driver is distracted or not using three data sources: visual, vehicle control, and physiological. In terms of distracted driving there are 10 datasets. With the three different methods: thresholding, classical machine learning, and deep learning to detect distracted driving, there is a segmentation process when distracted driving has been detected. With 2 types of segmentation processes and both of the segmentation processes work together to determine the best images for classification. After the segmentation process, the images are classified using the VGG-19 and Inception-v3 models and it can determine the accuracy of the prediction based on the classified results.

Le et al. [12] in the Viettel Group introduced a temporal triplet algorithm that segments dataset videos into clips based on frame count and frame distance. The algorithm enhances temporal information by merging the current frame with the next two equidistant frames, creating a new frame with heightened temporal density. This approach augments spatial information in each frame with details about future moments, increasing the temporal information density of the video clip. For their feature extraction backbone they selected X3D, or Expand 3D, which represents a CNN-based approach for action recognition, recognized for its efficacy in processing spatiotemporal features through a 3D CNN architecture and its computational efficiency, boasting a relatively small parameter count compared to other 3D CNN networks. Additionally, Improved Multiscale Vision Transformers (MViTv2) is a transformer-based model that they have adopted for its robust capabilities in both natural language processing and computer vision tasks. MViTv2, an improved iteration of MViT, showcases competitive performance in video tasks by learning a hierarchy from dense to coarse features. They use a two-level feature fusion approach, combining information from three camera views: dashboard, rear, and right-side window. The first step involves multi-view fusion, combining features from different perspectives to obtain a comprehensive multi-view feature. In the second step, multi-model fusion combines the multi-view features from the X3D and MViTv2 networks, resulting in an overall fused feature for improved context understanding in action detection. Finally, they all feed into the ActionFormer model which performs action localization by employing a Transformer-based encoder generating a temporal feature pyramid from clip features and a lightweight convolutional decoder for action classification and temporal boundary regression without relying on predefined anchor windows or action proposals.

Li et al. [13] they decided that the method of generating localization results based on snippet-level class probabil-

ities faced issues, such as limited temporal resolution and assigning the same class probability score to all positions within a snippet. To overcome these challenges, a proposed action probability calibration method was introduced, generating frame-level scores from snippet-level results, addressing reliability concerns, and utilizing prior knowledge from multiple camera views. This calibration enabled the use of overlapping snippets of varying sizes for parsing videos without information loss, and the calibrated action probabilities guided a class-customized filter mechanism for extending and filtering local action recognition results into long-term action regions. The complete process began with segmenting the input video into overlapping snippets of different sizes. These snippets underwent analysis by action recognition models to forecast action probabilities at the snippet level. Subsequently, these probabilities were transformed to the frame level and fine-tuned through an action probability calibration method. The refined probabilities were then inputted into the action localization component to produce discernible action regions.

Ma et al. [14] they introduce a M2DAR system which consists of two stages, Dar and Election. In the DAR stage, a sliding window technique is used to classify video clips into action categories, ensuring effective processing of long video sequences. This method captures both spatial and temporal information, highlighting the spatiotemporal characteristics of actions. In the Election stage, preliminary results from the action recognition module are refined to make a final prediction. This step is crucial for enhancing DAR performance by consolidating information from various camera views and selecting the most reliable action candidates. They use MViTv2 as the backbone model of the DAR system.

Zhou et. al [15] their approach was to divide the input video into brief clips and the assignment of class labels by distinct camera-view classifiers occurred first. Following this, an empirical selective multi-view multi-fold ensemble method is employed to obtain the action sequence at the clip level. Subsequently, a non-trivial post-processing technique is executed to derive segment-level localization results. They have determined certain camera angles that allow for better detection for certain actions, "By intuition, dashboard camera is expert in capturing the face-related activities (e.g. yawning, hand on head), rear view camera does well in hand-related activities (e.g. eating, texting), while right side camera has a good view for body-related actions(e.g. picking, taking)". In the post-processing they perform smoothing, linking and removing. Essentially, they correct mis-classified clips in smoothing, link adjacent segments into one clip, and remove isolated segments to denoise.

Vaswani et al. [16]'s model uses an "encoder-decoder structure" in which encoder has 6 layers and 2 sub layers on each layer. In the encoder-decoder model, there is a scaled dot-product and a multi-head attention layer. The scaled dot prodectuoin layer is calculated using the Attention$(Q, K, V) = softmax(QK^T/sqrtdk)V$ equation. The multi-head attention is used as it can perform the softmax in parallel and can give a dv dimensional output. The outputs than are concatenated to each other and they can be projected again. In the encoder-decoder model, multihead attention can be used from 3 algorithms. With the encoder-decoder structure, it can be used in machine translations, model variations, etc... and it can be useful as it can translate what the image detects to the results.

Yan et al. [17] used the CNN approach to detect distracted driving. In part of their experience they used a dataset that can recognize 6 distracted driving behaviors. With the CNN approach, they used the Gaussian mixture model which can identify different regions of distracted driving, the extracted images are then set to the primary and secondary regions where it will be processed by the CNN model to determine the action. In their model they used both the training and testing examples with training taking in examples and testing extracting the images. As a result the test accuracy for this model is considered as 95 percent or above accuracy.

Yang et al. [18] used the temporal action localization to determine distracted driving. With the A2Net, it can detect distracted driving in all instances from the starting time and ending time and it can indicate a category. The features are extracted by the I3D model and it is passed on to the Hierarchical Feature model. The Hierarchical feature model can generate features which can be used in the CNN layers etc... They then used the Anchor Based Localization Module which can than perform regression and can compare past predictions. As a result the A2Net has a higher performance than other modalities for video extraction.

In Yang et al. [19]'s approach, they used the TAD model to identify distracted driving. In the TAD model, each of the action in the video can be given an action class. With the TAD method, it can be a one stage or multi stage approach. In the one stage detection it can be focused with detecting the boundaries whereas on the multi stage detection, it can focus on different aspects of the action to get better action results. In their table it shows that with the TAD model, the mAP speed is higher than other detection models. The TAD model extracts the images and than goes through the augmentation, backbone, neck, and then the head to generate the result. In conclusion TAD has better performance than other detection methods for temporal modeling.

In Zhang et al. [20]'s approach they use the TAL to localize the actions in the videos. TAL extracts features from the video and creates the pyramids and each pyramid is considered as an action. TAL has single and two stage approaches. The two state approach can classify the proposals deeper and determine the action whereas the single approach only localizes without the action proposals. In terms of training they used the Adam training for warm up and it can generate good performance. With all pyramids each pyramid will determine an action and there are 2 convolutions for this approach. As a result the mAP of the TAL approach surpasses other approaches in the transformer models.

Zhao et al. [21] used the TAL approach to detect actions from video. In the TAL approach, the video can be divided into "temporal action proposal and action classification". In their method they used the network intraC and interC design for image convulsion. TAL extracts the video and they use "three 2-layer ProbNets" which will determine the prediction of the action. In their experiment they have 2:1:1 ratio in "training, validation, and testing." As a result the high of their mAP result is 0.95 which is considered one of the top performance on the AR@50 metric. As a result the intraC and interC can understand the learning process and improve the temporal action performance.

Baheti et al. [22] discusses the use of a dataset with ten classes, including activities like safe driving, texting,

and adjusting the radio. The dataset, collected from 31 participants in seven countries using four different cars, comprises 17,308 images divided into training and test sets. The study employs a Deep Convolutional Neural Network (CNN), a type of Artificial Neural Network inspired by the animal visual cortex, specifically modifying the VGG-16 architecture for distracted driver detection. VGG-16, known for its simplicity and depth, utilizes 3×3 filters in thirteen convolutional layers, ReLU activation function, and 2×2 max pooling. The architecture has been influential in image classification and localization tasks, contributing to the rapid progress in CNNs since 2012. The paper aims to adapt and explore VGG-16 for improved distracted driver detection by leveraging its proven effectiveness in various computer vision applications.

Buch et al. [23] introduces a novel recurrent model architecture designed for generating temporal action proposals, depicted in Figure 2. Unlike previous approaches, this model exhibits two distinctive features: firstly, it processes the input video precisely once at various time-scales without overlapping sliding windows, ensuring efficient runtime during inference; secondly, it evaluates numerous action proposals across densely sampled time-scales and locations, resulting in proposals with high temporal overlap with groundtruth action intervals. During inference, the model takes a long, untrimmed video sequence as input and sequentially processes each frame without utilizing overlapping sliding windows. The Visual Encoder module utilizes a 3D Convolutional (C3D) network to compute a feature representation capturing the visual content of the video. The Sequence Encoder module accumulates evidence across time, leveraging a Gated Recurrent Unit (GRU)-based architecture to efficiently operate over the input video in a single pass. The output module produces confidence scores for multiple proposals at each time step, considering proposals of various time scales in a single forward pass and applying standard post-processing techniques for proposal selection. Training examples are generated densely by extracting temporal segments in a sliding window fashion, facilitating the recurrent sequence encoder's full unrolling over long video sequences during testing.

Feichtenhofer et al. [24] introduces X3D, an innovative spatiotemporal architecture derived from a compact spatial network. The expansion of X3D occurs progressively, considering multiple potential axes such as space, time, width, and depth, carefully balancing computational efficiency and accuracy. An unexpected discovery during this expansion process is the effectiveness of networks featuring a slender channel dimension and elevated spatiotemporal resolution in video recognition tasks. X3D demonstrates commendable efficiency, and we anticipate that its performance will inspire further exploration and applications in the realm of video recognition.

Escorcia et al. [25] introduces the DAPs network transforms a sequence of visual observations spanning T frames into discriminative states, allowing for the deduction of temporal characteristics siK i=1 and durations of K action proposals within the stream. Each proposal si is accompanied by a confidence score ci. The network incorporates distinct modules: a Visual encoder, which condenses a small video volume into a concise low-dimensional feature vector using the top layer of a pre-trained 3D convolutional network (C3D network); a Sequence encoder, utilizing a long-short term memory (LSTM) network to encode visual codes into a discriminative sequence of hidden states,

effectively capturing sequential information ; a Localization module, predicting the positions of K proposals within the stream based on a linear combination of the last state in the sequence encoder, enabling the model to output segments of varying lengths in a single pass without the need for scanning overlapping segments; and a Prediction module, forecasting the confidence ci that a proposal si encompasses an action within its temporal extent. In practice, ci is determined by a sigmoid function applied to a linear combination of the last state of the sequence encoder.

Chapter 6

Design Methods

In our method, we used the baseline "Action Probability Calibration for Efficient Naturalistic Driving Action Localization" by Li, Wu, et al. [13]. In our design, we have three phases: 1) Preprocessing 2) Training 3) Inference [13].

6.1 Preprocessing

In the preprocessing phase, we took the video datasets with each video being 7-8 minutes long and created smaller segments of each video based on the corresponding classification [13]. During this process, we also resized the dataset videos from 1024x1024 resolution to 512x512 to shorten the training time [13]. After reducing the resolution and splitting the videos, we created a new folder with the preprocessed data, which was used for training. We also created CSV files for each orientation, including each preprocessed video within that orientation and its predicted value [13].

6.2 Training

In the training phase, we used the Temporal Action Localization method as our baseline by Li, Wu, et al. [13]. In their model, there are 16 different classifications to identify distracted driving behaviors. The actions are: 0) Normal Forward Driving, 1) Drinking, 2) Phone Call Right, 3) Phone Call Left, 4) Eating, 5) Text Right, 6) Text Left, 7) Reaching Behind, 8) Adjusting Controls, 9) Pick Up From Floor Driver, 10) Pick Up From Floor Passenger, 11) Talk to Passenger at the Right, 12) Talk to Passenger at Backseat, 13) Yawning, 14) Hand on Head, 15) Singing or Dancing with Music.

Part of their training method, they used the X3D model to predict the probabilities of the trained data. With the X3D model, the action predicted is based of the average score of all videos containing the classification. To improve the classification results, they used the formula:

$$\frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{i^2 f}{2\sigma^2}}$$

to calculate the weights.

To recognize distracted driving datasets, we used SlowFast Networks by Facebook Research, which have strong performance rates for video recognition [26]. With SLowfast, their are two frame rates that work simutanously to produce a predicted output. The slower pathways rates are calculated by view cost $\times$ the number of views (T $\times$ t) and the rates $\times$ the temporal resolution (a $\times$ T) for faster pathways.

For our training, we used convolutional neural networks utilizing two Nvidia GPUs. We used 50 epochs per frame with a learning rate of 5e-4 and a batch size of 48 [13]. In each epoch, the learning rate is updated using the Adam optimizer based on the current epoch and data size. After updating the learning rate, we reduced the mean and calculated the loss and nanloss. Finally, we filter out any probabilities with lower confidence values to better optimize our result. After filtering out low-confidence probabilities, the remaining results were merged and we calculate the validation using the equation:

$$\text{Probability} = \frac{\sum \text{weights} \times \text{Probability of the frame}}{\sum \text{weights}}$$

[13]. From the validation, we filtered out redundant results and took the maximum result using Max(Probability) $\times$ ratio.

6.3 Inference

The last stage of the classification process is inference [13]. In this stage, we used our testing data and the most recent trained epoch results to validate our trained results. During the inference process, we resized the videos in the testing dataset and set the padding. We determined the padding size of x1 = (old width - new width)/2 and y1 = (old height - new height)/2, and we set the resized videos to be the padding of x1 and y1. Once the videos were resized, we calculate the normalization and permutations of each video. The GPU then performs gradient descent using the forward passing algorithm based of the results from normalization and permutations. This process was repeated for all six frames we have: expand rear, expand right, expand dashboard, original rear, original right, and original dashboard. After performing gradient descent, we obtain the probable results for each frame based of the 16 classifications. After analyzing our baseline results and examining each angle's probability score at each timestamp when a distraction was detected, we found errors in the final two videos. Specifically, we observed that activity 9 (picking up something from the floor) would be ignored if performed for short periods. To address this, we increased the weight for activity 9 in the dashboard view. Additionally, we noticed that activity 4 (eating) in the right side view was not being considered, despite having a better angle. Therefore, we increased the weight for activity 4 in the right side view.

6.4 Architecture Diagram

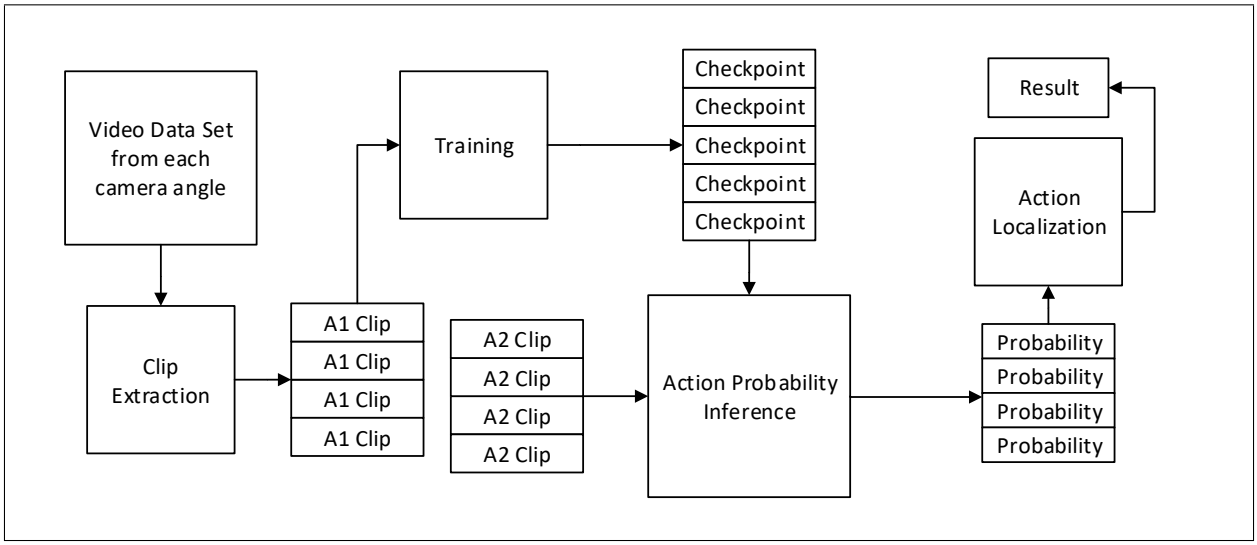

Figure 6.1: Architecture Diagram

In Figure 6.1, we describe the process of how our distracted driving system works. We first input the video datasets from every angle. After the videos are inputted, we preprocess the videos by extracting and splitting each video dataset into smaller videos. Once the videos are extracted, we go through the training phase. During the training phase, we write a checkpoint file for each epoch based on the angle. Once the checkpoint files are generated, we perform inference using our testing dataset and generate a probability. Based on the probability, we generate an action, which then becomes the final result.

Chapter 7

Constraints and Standards

7.1 Constraints

Throughout the phases of our projects, we have faced constraints during our initial setup. When setting up, we faced constraints when it came with different Pytorch versions. When we first installed following the author's original instructions, during our training phase, we have received module errors relating to $torch._six$ not found. At first we didn't know what the issue is as we just cloned and tested the baseline in the author's original work. We however, then did researched the error including looking at forums to see if other people faced the $torch._six$ error. We then spoke to our advisor as well as contacted the HPC Machine director at our school to check our environments to see if we're using the right environment. After intense digging, we then figured out that $torch._six$ module is no longer supported on newer versions of Pytorch including version 2.3.0 which we used. We then looked into the code where the $torch._six$ error is happening and we removed all the $torch._six$ imports and rerun the training. By removing the $torch._six$, we were able to circumvent the module error and get training working afterwards. Many of the modules didn't include specified versions which is what caused many issues and headaches.

7.2 Standards

Open-Source Collaboration: We adopted open-source models and libraries to promote collaboration and extendibility. Using frameworks like SlowFast [27] and adhering to open-source principles allows other researchers and developers to build on our progress, contributing to a larger body of work and advancing the field collectively.

ISO/IEC 27001 (Information Security Management) [28]: Although not explicitly certified, we followed the principles of this standard to ensure data security and privacy. This was particularly important when handling sensitive video data of drivers that should only be used within the competition.

Model Evaluation Metrics: We followed standard evaluation metrics in our results provided by the AI City Challenge [29], such as accuracy, precision, recall, and F1-score. These metrics helped us objectively evaluate the performance of our models.

Chapter 8

Evaluation

8.1 Evaluation Methodology

For our evaluation methodology, after every inference, we verify the results and the video datasets where the results refer to. In our results, we have the video number, classification number, and the start and the end intervals. Based of the results, we re-watch the videos and compare it with the results generated. We note down all the errors and we would adjust the weights, epochs, and learning rates, etc... based of the wrong results. We then retrained or re-inference the data and see if the new results are worse or better. We keep on experimenting changes until we can determine the results are accurate. After making these changes and improve the results, we will send our data and get an overall report of our results. If the overall results isn't good, we kept retrain and/or re-inference the data under different weights, epochs, and learning rates, etc...

8.2 Results

Attempt	Metric	Value
Initial	Success	0.7041
Final	Success	0.7173

Table 8.1: Comparison of Initial and Final Results

We were able to improve upon our baseline result where Table 8.1 shows the comparison between our initial and final results. Activity classes 4 and 9 were more accurately identified in videos 9 and 10, which is why we see improvement in the result. However, the results are still not accurate as we have seen other clips that have been misidentified and has to be updated. Therefore, we did improve on our baseline from 0.7041 to 0.7173 but we believe that there are still false results that have not been accurately identified

Chapter 9

Societal Issues

With distracted driving being on the rise around the world, distracted driving can pose huge impacts on society in terms of both life and the economy. According to Understanding Driving Distractions: A Multimodal Analysis on Distraction Characterization and the NHTSA, in addition to the number of fatalities, more than 400,000 casualties happen per year, and distracted driving is the main cause [30, 31]. With distracted more than 154 billion U.S dollars have been loss due to workforce loss. By implementing machine learning to address distracted driving at an early stage, we can lower the number of distracted driving-related casualties and financial losses due to these casualties.

9.1 Ethical

With distracted driving, numerous ethical issues have been on the rise. According to Steven Mintz, Ethics, drivers engaging in distracted driving and not paying attention to the road are endangering the community around them [32]. When people are driving distractedly and not caring for their surroundings, it means they are being self-centered and not caring for their community members. With distracted drivers sharing the road with other people, they pose a life and safety risk to the community, and we believe it is unjust as everybody deserves equal rights in both life and safety. Therefore, we want to educate drivers using machine learning so they can understand the impacts of distracted driving early on in order to protect the community.

9.2 Environmental Impact

Distracted driving can create negative impacts on the environment. With more distracted drivers, there can be higher accident rates, leading to more traffic congestion due to obstructions on the road [33]. According to the NHTSA, distracted driving contributes to 16.9/60.3 of congested traffic and 27.5/62.0 of obstructions on roadways [33]. These incidents can affect the environment by increasing CO2 emissions, causing air pollution from exhausts, and leading to more congested roads and accidents [34]. With more CO2 being released into the environment, air quality can become unhealthy and harmful to humans, plants, and animals. Therefore, by preventing distracted driving, we can lower the

number of CO2 emissions and improve air quality.

9.3 Usability

In terms of usability, when the driver is identified as distracted, the system we designed can alert and remind drivers to focus on the road [13]. Given that many people aren't aware of the dangers of distracted driving, the distracted driving detection tool can help detect and identify distracted driving and inform drivers that they shouldn't be driving distractedly [32, 13]. Using this product would help improve experiences on the road by educating distracted drivers early before any severe incidents occur [13, 30]. In conclusion, users will use the distracted driving detection system to understand the impacts and safety of distracted driving.

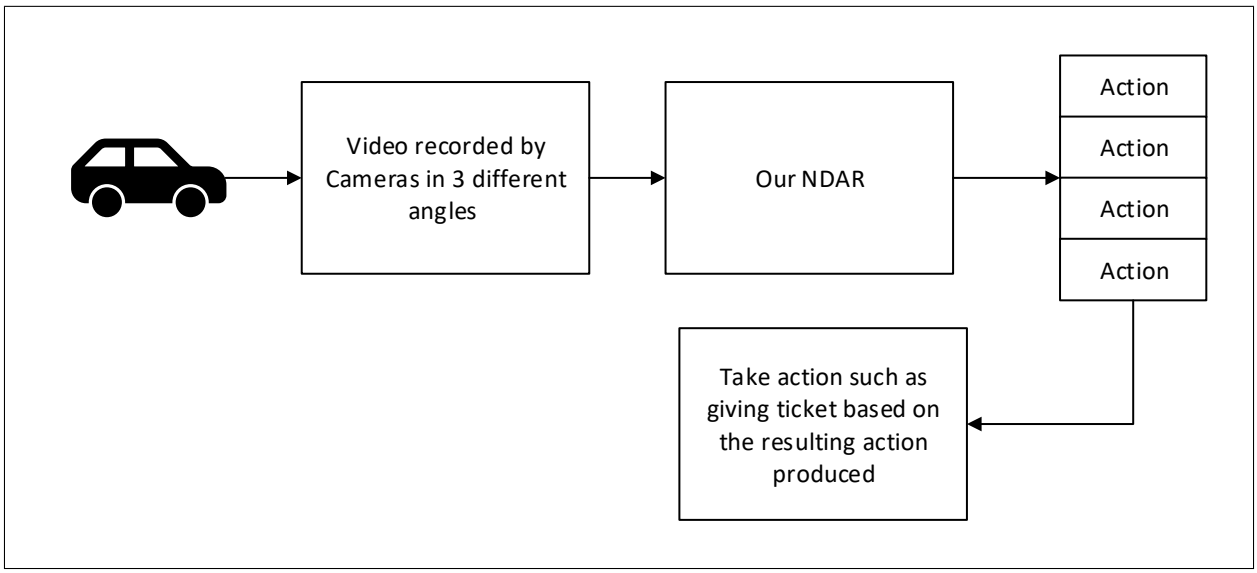

Figure 9.1: Use Case Diagram

9.4 Lifelong Learning and Compassion

We decided to undertake this project because we believe that distracted driving remains a problem even with existing laws, and it needs to be addressed [32, 13]. We believe that distracted driving is dangerous, and we want to educate drivers about its impacts at an early stage to protect them and the community. Our naturalistic distracted driving detection aims to educate drivers about different types of distracted driving early on so they can learn and become safer drivers. By implementing this detection system, we can educate drivers at an early stage and teach them the importance of driving safely and not being distracted.

9.5 Engineering Character

In terms of engineering, we designed the system to be user-friendly, with built-in cameras in the car that can identify distracted driving from different views [13]. We value the privacy of the user; instead of transmitting footage to others, the driver will be the only one aware of the footage. Our focus is on educating drivers rather than creating trouble with the law and insurance. Each time the system detects distracted driving, it alerts and reminds drivers to focus on driving. In general, while maintaining privacy, we aim to educate drivers by reminding them of the importance of safety and not getting distracted while on the road.

Chapter 10

Conclusion and Future Work

10.1 Future Work

While our initial idea for the Naturalistic Distracted Driving Detection project was to detect if a driver is being distracted in the views of the camera, it may not 100 percent stop all drivers from driving distracted. During the Senior Design Conference at Santa Clara University, one of our judges suggested the idea that we haven't thought about was how can we address the situation if the driver knows how to evade being detected of distracted driving. This has brought us the attention and we believe it is a reasonable feedback and if we are able to improve our project, we would also train the datasets not only to detect the actual distracted driving, but also detect any body language that can resemble as distracted driving. We would also include other classifications in our datasets that can pose suspicion on distracted driving including if hands are not on the wheel or stick shift etc... We would also include datasets that could identify distracted driving with more angles like lower Dashboard, and Left Side etc... in addition to the three angles we have (Dashboard, Rearview, and Right Side). With these new additions being added, we believe we can improve the current version of our project by having more datasets and classifications to detect distracted driving.

10.2 What We Have Learned

Through doing the Naturalistic Distracted Driving Detection project, we have learned and better understand machine and deep learning. Through training the datasets we can better understand how convolution neural networks and slowfast works and how it can identify distracted driving from the datasets[13][26]. We were also able to understand how the learning rate and epoch sizes can be an important factor in determining the results as well as the error margins etc[13]... Based of each training and inference result, we were able to experiment the weights and understand the results after gong through the convolutional neural network model. As a result, we learned and understand more about convolutional neural network, artificial intelligence, and machine learning.

10.3 Why it is Important

We did this project as we believe that distracted driving has been a serious problem in our community[31][35]. According to the National Highway Traffic Safety Administration (NHTSA), there have been at least 3000 fatalities relating to distracted driving just in 2022 alone[31][35]. With us implementing this project, we believe we can use machine learning to detect distracted driving and educate drivers on the importance of avoiding distracted driving, as many drivers are not aware that their act of distracted driving is putting people's lives in danger. With the 16 classifications we have, we are able to identify multiple types of distracted driving, and we believe that by detecting distracted driving and educating drivers on the importance of driving safely, we can see the number of fatalities caused by distracted driving go down and lower auto accident rates[31][35].

Bibliography

[1] G. L. F. Groth Law Firm, "100 distracted driving facts amp; statistics," May 2022.

[2] H. D. Le, M. Q. Vu, M. T. Tran, and N. Van Phuc, "Triplet temporal-based video recognition with multiview for temporal action localization," in *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 5428–5434, 2023.

[3] S. Ahangari, M. Jeihani, and A. Dehzangi, "A machine learning distracted driving prediction model," in *Proceedings of the 3rd International Conference on Vision, Image and Signal Processing*, ICVISP 2019, (New York, NY, USA), Association for Computing Machinery, 2020.

[4] Y. Li, X. Ling, and J. Yang, "Analysis of distracted driving detection based on deep learning human posture," in *Proceedings of the 2023 9th International Conference on Computing and Artificial Intelligence*, ICCAI '23, (New York, NY, USA), p. 46–50, Association for Computing Machinery, 2023.

[5] C. Streiffer, R. Raghavendra, T. Benson, and M. Srivatsa, "Darnet: A deep learning solution for distracted driving detection," in *Proceedings of the 18th ACM/IFIP/USENIX Middleware Conference: Industrial Track*, Middleware '17, (New York, NY, USA), p. 22–28, Association for Computing Machinery, 2017.

[6] H. Shang, Y. Qian, R. Shi, Y. Li, and X. Liang, "A novel method for unsafe driving states detection," in *Proceedings of the 2020 12th International Conference on Machine Learning and Computing*, ICMLC '20, (New York, NY, USA), p. 373–378, Association for Computing Machinery, 2020.

[7] Y. Wang, C. Wang, H. Luo, and F. Zhao, "Drivcapsnet: A driving style detection algorithm based on capsule network," in *Proceedings of the 2023 4th International Conference on Computing, Networks and Internet of Things*, CNIOT '23, (New York, NY, USA), p. 157–163, Association for Computing Machinery, 2023.

[8] S. Jeon, S. Lee, E. Lee, and J. Shin, "Driver monitoring system based on distracted driving decision algorithm," in *2022 13th International Conference on Information and Communication Technology Convergence (ICTC)*, pp. 2280–2283, Oct 2022.

[9] X. Qi and T. Li, "Driver distracted behavior detection based on deep learning," in *Proceedings of the 2022 6th International Conference on Electronic Information Technology and Computer Engineering*, EITCE '22, (New York, NY, USA), p. 786–790, Association for Computing Machinery, 2023.

[10] S. Yadawadkar, B. Mayer, S. Lokegaonkar, M. R. Islam, N. Ramakrishnan, M. Song, and M. Mollenhauer, "Identifying distracted and drowsy drivers using naturalistic driving data," in *2018 IEEE International Conference on Big Data (Big Data)*, pp. 2019–2026, 2018.

[11] A. Ezzouhri, Z. Charouh, M. Ghogho, and Z. Guennoun, "Robust deep learning-based driver distraction detection and classification," *IEEE Access*, vol. 9, pp. 168080–168092, 2021.

[12] H. D. Le, M. Q. Vu, M. T. Tran, and N. Van Phuc, "Triplet temporal-based video recognition with multiview for temporal action localization," in *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 5428–5434, 2023.

[13] R. Li, C. Wu, L. Li, Z. Shen, T. Xu, X.-J. Wu, X. Li, J. Lu, and J. Kittler, "Action probability calibration for efficient naturalistic driving action localization," in *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 5270–5277, 2023.

[14] Y. Ma, L. Yuan, A. Abdelraouf, K. Han, R. Gupta, Z. Li, and Z. Wang, "M2dar: Multi-view multi-scale driver action recognition with vision transformer," in *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 5287–5294, 2023.

[15] W. Zhou, Y. Qian, Z. Jie, and L. Ma, "Multi view action recognition for distracted driver behavior localization," in *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 5375–5380, 2023.

[16] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2023.

[17] S. Yan, Y. Teng, J. S. Smith, and B. Zhang, "Driver behavior recognition based on deep convolutional neural networks," in *2016 12th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD)*, pp. 636–641, 2016.

[18] L. Yang, H. Peng, D. Zhang, J. Fu, and J. Han, "Revisiting anchor mechanisms for temporal action localization," *IEEE Transactions on Image Processing*, vol. 29, p. 8535–8548, 2020.

[19] M. Yang, G. Chen, Y.-D. Zheng, T. Lu, and L. Wang, "Basictad: an astounding rgb-only baseline for temporal action detection," 2023.

[20] C. Zhang, J. Wu, and Y. Li, "Actionformer: Localizing moments of actions with transformers," 2022.

[21] P. Zhao, L. Xie, C. Ju, Y. Zhang, Y. Wang, and Q. Tian, "Bottom-up temporal action localization with mutual regularization," 2021.

[22] B. Baheti, S. Gajre, and S. Talbar, "Detection of distracted driver using convolutional neural network," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 1145–11456, 2018.

[23] S. Buch, V. Escorcia, C. Shen, B. Ghanem, and J. C. Niebles, "Sst: Single-stream temporal action proposals," in *SST: Single-Stream Temporal Action Proposals*, 07 2017.

[24] C. Feichtenhofer, "X3d: Expanding architectures for efficient video recognition," *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 200–210, 2020.

[25] V. Escorcia, F. C. Heilbron, J. C. Niebles, and B. Ghanem, "Daps: Deep action proposals for action understanding," in *European Conference on Computer Vision*, 2016.

[26] C. Feichtenhofer, H. Fan, J. Malik, and K. He, "Slowfast networks for video recognition," 2019.

[27] facebookresearch *PySlowFast: video understanding codebase from FAIR for reproducing state-of-the-art video models*, Jan 2023.

[28] I. O. for Standardization *ISO/IEC 27001 - Information security management*, Oct 2013.

[29] A. C. Challenge *AI City Challenge Evaluation Metrics*, Jan 2023.

[30] M. Papakostas, K. Riani, A. B. Gasiorowski, Y. Sun, M. Abouelenien, R. Mihalcea, and M. Burzo, "Understanding driving distractions: A multimodal analysis on distraction characterization," in *Proceedings of the 26th International Conference on Intelligent User Interfaces*, IUI '21, (New York, NY, USA), p. 377–386, Association for Computing Machinery, 2021.

[31] United States Department of Transportation, "Distracted driving," 2024.

[32] S. Mintz, "The ethics of distracted driving," Jan 2017.

[33] N. NHTSA *Distracted Driving and Driver, Roadway, and Environmental Factors*, p. 18–24, Sep 2010.

[34] M. C. METROPOLITAN COUNCIL *The Negative Effects of Traffic Congestion on the Twin Cities and the State of Minnesota*, Jan 2020.

[35] J. Bieber and C. Bieber, "Distracted driving statistics & facts in 2024," Apr 2024.