Master Thesis

Improved Driver Distraction Detection Using Self-Supervised Learning

by

Suraj Bhardwaj
University of Siegen

Examiner 1: Prof. Dr. Michael Möller
Head of Computer Vision Group
University of Siegen

Examiner 2: Dr. Jovita Lukasik
Post-Doctoral Researcher at Computer Vision Group
University of Siegen

Additional Supervisor: David Lerch M.Sc.
Perceptual User Interface Group
Fraunhofer IOSB, Karlsruhe

A Thesis Submitted to the Faculty of Electrical Engineering and Computer Science
In Partial Fulfillment of the Requirements for the Degree of Master of Science (M.Sc.)
International Graduate Studies in Mechatronics

Universität Siegen
North Rhine-Westphalia, Germany
Date of Submission: 15 May 2024

Declaration of Authorship

I hereby confirm that this thesis and the work presented in it is entirely my own. Where I have consulted the work of others this is always clearly stated. All statements taken literally from other writings or referred to by analogy are marked and the source is always given. This paper has not yet been submitted to another examination office, either in the same or similar form.

Place, Date: **Siegen, 15 May, 2024** Student's Signature: Suraj Bhardwaj

Acknowledgments

I am deeply grateful to my advisors, Prof. Dr. Michael Möller, Dr. Jovita Lukasik, and David Lerch M.Sc., for their invaluable guidance, enduring patience, and profound expertise throughout my research journey. Their insightful feedback and thoughtful recommendations were instrumental in shaping this thesis.

I also want to thank everyone at the Perceptual User Interface Department at Fraunhofer IOSB, Karlsruhe, for their support and for providing the tools and environment I needed for my work. My peers and colleagues at Fraunhofer IOSB also deserve profound acknowledgment for their friendship, engaging discussions during our regular student workshops, and the shared insights that significantly enriched my graduate experience.

I would also like to recognize the pioneering authors and researchers in my field whose foundational work has inspired and guided my research. Their significant contributions have illuminated my path and set a high standard for my work.

Finally, heartfelt gratitude goes to my family—my parents, Shri Sohan Singh and Smt. Jai Dei, and my niece Shivanshi. Their endless support and understanding through the demands of my academic pursuits have been a constant source of strength and motivation. Their love and encouragement have been the pillars of my strength and inspiration.

Contents

List of Figures

List of Tables

Acronyms

ADAS Advanced Driver Assistance Systems. 13

CLARA Clustering Large Applications. 27

CNN Convolutional Neural Networks. 16–18, 31, 34, 35

DAA Drive and Act. 11, 14–16, 19–23, 34, 38–40, 52, 77, 80, 83

DBSCAN Density-Based Spatial Clustering of Applications with Noise. 27

DINO Self-Distillation with no labels. 29, 33–36

DINOv2 Self-Distillation with no labels v2. 14, 29, 35

HDBSCAN Hierarchical Density-Based Spatial Clustering of Applications with Noise. 27

iBOT Image bert pre-training with online tokenizer. 35, 36

IR Infra-Red. 11, 14, 59

KIR Kinect Infra Red. 11

MLP Multi-layer perceptron. 17, 18, 32, 33

RGB Red Green Blue. 14

RNN Recurrent neural network. 17

SCUT Multi-class imbalanced data classification using SMOTE and cluster-based undersampling. 27

SF State Farm Dataset. 16–19

SMOTE Synthetic Minority Oversampling Technique. 25–27

SSL Self-Supervised learning. 11, 14, 36, 37, 55, 80, 81

SVM Support Vector Machines. 16, 26

SwAV Swapping Assignments between Views. 35, 36

ViT Vision Transformer. 27, 28, 31, 34, 35

Abstract

This thesis investigates the binary classification task of classifying driver distraction using Drive and Act (DAA) (Martin et al., 2019) image datasets and vision transformer encoders that are pre-trained using Supervised and Self-Supervised learning (SSL) learning approaches respectively. The main focus is dealing with data imbalance, comparing the performance of vision transformer encoders pre-trained using supervised and SSL approaches, and evaluating their ability to generalize across different views and modalities. In order to utilize the DAA video dataset, this thesis creates image datasets derived from the DAA dataset's videos. This study presents the 'Clustered Feature Weighting' dataloading technique as a solution to the data imbalance issue in the image datasets derived from DAA (Martin et al., 2019) video dataset. This approach uses the HDBSCAN (Campello et al., 2013) algorithm for unsupervised clustering of features extracted by a pre-trained vision transformer model. It also incorporates a weighted random sampler (PyTorch Contributors, 2023c) to balance the training batches based on generated weights using unsupervised clustering. The results show that 'Clustered Feature Weighting' successfully achieves a balanced class distribution in batches during data loading. Additionally, it shows signs of improvements in model performance on the Kinect Color DAA dataset (Martin et al., 2019). Specifically, it increases the train balanced accuracy by 1.03%, the validation balanced accuracy by 0.5%, and the test balanced accuracy by 0.08% compared to the traditional imbalanced dataloading method. Nevertheless, additional research is necessary to elucidate this technique's potential advantages and disadvantages for model training and generalization.

Furthermore, this thesis evaluates cross-view generalization across two different image views (right top and front top) and cross-modality generalization across two modalities (RGB and infrared) from DAA dataset on the driver distraction detection task. The SSL-based encoder consistently performed well in all permutations, especially when using infrared or grayscale imagery. Despite the problems in cross-view generalization, the findings affirm that SSL based encoders have the potential to enhance the adaptability and robustness of driver distraction detection systems. The comparison between the SSL-based vision transformer (Dosovitskiy et al., 2020) encoder, specifically the DINOv2 (Oquab et al., 2023) based vit_b_14 encoder (Facebook AI Research, 2023), and the supervised learning-based vit_b_16 (PyTorch Contributors, 2024) encoder, demonstrated that the SSL-based model displayed remarkable feature extraction abilities, leading to significantly improved performance, especially in tasks that require generalization across grayscale or Infra-Red (IR) image modality. On the Kinect Infra Red (KIR) DAA cross-modality test dataset, the SSL-based model showed a higher performance level on cross-modality generalization than the supervised learning-based model, with an improvement of up to 7.17% . While the initial train balanced accuracies are satisfactory, underfitting highlights the urgent need for better hyperparameter optimization to fully use the model's potential. This thesis contributes significantly to automotive safety, demonstrating the feasibility and advantages of employing SSL-based encoders for improving driver distraction detection.

Keywords: Self-Supervised Learning, Driver Distraction Detection, Cross-Modality Generalisation, Cross-View Generalisation, Drive and Act Image Dataset, Dataset Imbalance, Vision Transformer.

Chapter 1

Introduction

This chapter briefly introduces driver distraction detection. Why is it essential for the computer vision community, and how does it relate to real-world applications and problems? Furthermore, it includes the central research questions guiding this thesis, along with a brief outline of the structure of this thesis.

1.1 Introduction to Driver Distraction Detection

Integrating artificial intelligence and advanced computing in the automotive industry has led to significant road safety and autonomous driving developments. Although there have been significant technological improvements, most vehicles still rely on manual operation by the driver. However, the primary cause for traffic accidents continues to be driver distraction, which includes participating in activities such as using mobile phones, texting, or conversing with passengers (Li et al., 2022a; Wang et al., 2022). World Health Organization (2023) investigated worldwide road traffic fatalities, advancements in safety legislation, and initiatives to decrease the number of deaths. The data indicates a slight decline in annual fatalities caused by road traffic accidents, reaching 1.19 million [1]. However, it highlights the urgent requirement for significant measures to achieve the United Nations's target of reducing road traffic deaths and injuries by 50% by 2030 (World Health Organization, 2023; Walugembe et al., 2020). According to the National Highway Traffic Safety Administration (2023), distracted driving resulted in 3,308 deaths in 2022 in the USA. The phenomenon of driver distraction displays notable variations in its development throughout its examination by researchers. Numerous seminal publications and review papers, such as (Regan et al., 2011; Young et al., 2007; Wang et al., 2022; Moslemi et al., 2021), have thoroughly examined this phenomenon.

According to National Highway Traffic Safety Administration (2023),

> "distracted driving is any activity that diverts attention from driving, including talking or texting on your phone, eating and drinking, talking to people in your vehicle, fiddling with the stereo, entertainment or navigation system — anything that takes your attention away from the task of safe driving." (National Highway Traffic Safety Administration, 2023)

This broad definition captures a variety of distraction types, including visual, manual, cognitive, and auditory distractions, as described in previous research works such as Moslemi et al. (2021); Regan et al. (2011); Li et al. (2020a). For example, visual distractions can occur when drivers glance at passengers in the back

[1] Data for Action on Road Safety: The 2023 Global Status Report on Road Safety

seat, watch a multimedia screen, or look at objects placed on the passenger seat, thus diverting their focus from the road. In contrast, cognitive distractions [2] arise when a driver is mentally absorbed in personal concerns or daydreams, even while visually monitoring the road (Moslemi et al., 2021). As vehicles evolve towards higher levels of autonomy and reduce the driver's responsibilities, the likelihood of drivers engaging in distracting activities grows. This trend is expected to persist until vehicles reach full autonomy and no longer need any driver input (Li et al., 2022a). This situation highlights the paradox where greater vehicle autonomy and minimal driver involvement do not necessarily equate to enhanced safety and reliability.

Moreover, the use of entertainment technology in vehicles has increased from its inception to the present. According to the recent Allianz Distraction Study, such technologies significantly distract drivers, as evidenced in figure 1.1 (SE, 2023). This figure illustrates the rise in distracted driving activities from 2016 to 2022, paralleling technological advancements in the automotive sector. At the same time, to mitigate these distractions, innovations in driver monitoring systems and their integration in the Advanced Driver Assistance Systems (ADAS) play a crucial role by alerting distracted drivers, potentially reducing accident rates (Moslemi et al., 2021).

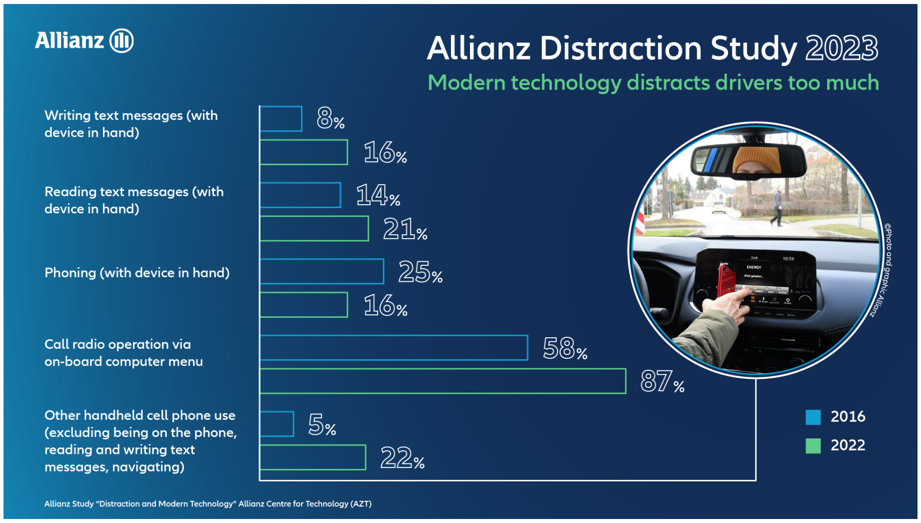


Figure 1.1: Increase in instances of inattentive driving from 2016 to 2022. This figure illustrates the upward trend in various activities, such as reading text messages on a mobile phone while driving, from 2016 to 2022. These activities have played a role in the rise of distracted driving behavior. Source: (SE, 2023)

1.2 Motivation

Driver distraction stands as a significant cause of road accidents, necessitating rigorous research to foster a safer environment for road users (Li et al., 2022a; Wang et al., 2022; Regan et al., 2011; Young et al., 2007; Moslemi et al., 2021). Such research aligns with the United Nations' 2030 goal to decrease road traffic fatalities and injuries substantially. Traditionally, in computer vision, driver distraction detection has leveraged supervised learning algorithms, which depend on labeled data. However, acquiring this data is often expensive and labor-intensive. Furthermore, supervised learning may not sufficiently generalize across diverse driving conditions, underscoring the need for unsupervised or self-supervised learning methods (Li

[2] Cognitive Distractions While Driving: What You Need to Know

et al., 2022a). Recent technological advances have equipped vehicles with various sensors and cameras, significantly enhancing the capability to monitor a broad spectrum of driver actions Wang et al. (2022); Martin et al. (2019). The DAA Dataset (Martin et al., 2019) presents a precious resource for pushing the boundaries of current research in driver activity recognition and distraction detection. This dataset includes multiple data modalities, multi-view images, hierarchical labeling, and a detailed differentiation between various driver actions (Martin et al., 2019). These attributes combined establish it as an exceptional platform for extensive research and analysis.

The primary objective of this research is to conduct a comprehensive exploration of the DAA Dataset (Martin et al., 2019) aimed at enhancing our understanding and detection of driver distractions. This study engages in meticulous experimentation and analysis to assess the efficacy of using Red Green Blue (RGB) (color) and IR (Gray Scale) imaging modalities from the DAA video dataset (Martin et al., 2019). A focal point of this research is extracting and creating image datasets and addressing dataset imbalances.

In advancing the field of driver distraction detection, this thesis utilizes advanced pre-trained vision transformer encoders. It explores their application across different image modalities and views, testing their adaptability and generalizability. The comparative analysis includes an encoder trained using a supervised learning technique and the one developed through SSL framework DINOv2 (Oquab et al., 2023). The sole purpose of this analysis is to highlight the performance of encoders trained using different techniques like supervised and SSL on driver distraction detection and to evaluate the impact of various data modalities and image views on the accuracy and generalization of distraction detection.

Ultimately, this thesis aims to enhance driving safety, aligning with global safety objectives significantly.

1.3 Research Questions

This thesis embarks on an exploration guided by these essential research questions:

- **Practical Challenges:** How can the issue of data imbalance in the DAA dataset (Martin et al., 2019) be addressed? Is it possible to employ unsupervised learning techniques for this purpose, and if so, how can they be effectively implemented?
- **Effectiveness of SSL Models:** What are the benefits and drawbacks of using vision transformer encoders pretrained using SSL based approaches, such as Self-Distillation with no labels v2 (DINOv2) (Oquab et al., 2023) over supervised learning based encoder, for detecting driver distraction?
- **Generalization Capabilities:** How do different image views, such as the right top view and the front top view in the DAA dataset (Martin et al., 2019), impact the detection of driver detection using computer vision? Additionally, does a vision transformer (Dosovitskiy et al., 2020) encoder pre-trained using a self-supervised learning approach maintain consistent performance on driver distraction detection tasks or generalize well across different views at hand?
- **Data Modality Impact:** What are the impacts of varying data modalities, such as RGB and IR images, on detecting driver distractions? Furthermore, does the vision transformer encoder pre-trained using the self-supervised learning approach DINOv2 (Oquab et al., 2023) demonstrate effective generalization across these modalities?

1.4 Brief Outline of the Thesis

The structure of the rest of the thesis is as follows:

- **Chapter 2: Related Work** - A thorough literature review examines existing methods and technologies relevant to driver distraction detection. This chapter highlights the limitations of conventional supervised learning approaches and highlights the necessity for innovative unsupervised or self-supervised learning models. It also highlights the practical challenges with DAA dataset and identifies the research gaps this thesis aims to address.

- **Chapter 3: Background** - This chapter details the essential methodologies employed in this thesis. It explains the working principle of the HDBSCAN (Campello et al., 2013) clustering algorithm and vision transformer Dosovitskiy et al. (2020) and introduces relevant mathematical functions and terminology necessary for subsequent discussions. Furthermore, it explains the self-supervised learning-based framework DINO (Caron et al., 2021), as well as the DINOv2 framework (Oquab et al., 2023).
- **Chapter 4: Proposed Methods** - This chapter provides a comprehensive explanation of the innovative dataloader implementation, as well as a methodology to compare it with the traditional dataloader. This chapter also provides descriptions of the methodologies used in supervised learning-based encoder experiments and self-supervised learning-based encoder experiments. Furthermore, this chapter discusses the theoretical foundations of the selected methods and explains their relevance in addressing the research goals outlined in the section 1.3.
- **Chapter 5: Experiments and Results** - This section delves into the experimental setup, highlighting the datasets utilized and the hyperparameter choices made. It also presents the implementation and results of the previously outlined methodologies across all experiments, offering a comprehensive analysis of the findings. Moreover, it specifically addresses the research questions posed in section 1.3.
- **Chapter 6: Conclusions and Future Work** - Concluding the thesis, this section integrates the findings, and identifies potential areas for future research to foster continued advancements in the field.

Chapter 2

Related Work

Continuing from the previous chapter, where we discussed this thesis's goals and their relevance to real-world applications, this chapter provides a detailed literature review of existing distraction detection methods in computer vision. It highlights the limitations of traditional supervised learning methods and underscores the need for innovative unsupervised or self-supervised learning models. Additionally, this chapter addresses research gaps and discusses practical issues associated with the DAA dataset.

2.1 Supervised Learning Based Methods

Supervised learning is a prevalent approach in detecting driver distractions, leveraging annotated datasets to train models to recognize distraction patterns. In the research work on driver distraction detection, Zhang (2016) applied and compared the performance of Support Vector Machines (SVM) (Cortes & Vapnik, 1995) and Convolutional Neural Networks (CNN) (O'shea & Nash, 2015) algorithms using the State Farm Dataset (SF) (Montoya et al., 2016) dataset which contains 10 classes for driver distraction detection. For SVM (Cortes & Vapnik, 1995), Zhang (2016) utilized a Support Vector Classifier (SVC) model employing a one-vs-one scheme to handle multi-class classification. The CNN (O'shea & Nash, 2015) approach involved three models: a simple CNN based on the MNIST (Deng, 2012) digit classification task, a transfer learning model leveraging the pre-trained VGG-16 (Simonyan & Zisserman, 2014), and a modified VGG model (VGG-GAP) (Zhou et al., 2016) with global average pooling layers. The CNN models outperformed the SVM, with the simple CNN (O'shea & Nash, 2015) achieving an accuracy of 63.3%, VGG-16 (Simonyan & Zisserman, 2014) reaching 90.2%, and VGG-GAP (Zhou et al., 2016) achieving 91.3%. The ensemble of VGG-16 and VGG-GAP demonstrated superior performance with an accuracy of 92.6% (Zhang, 2016).

Okon & Meng (2017) used the AlexNet (Krizhevsky et al., 2012) architecture as a baseline model (Model A) for the driver distraction detection task on the SF dataset (Montoya et al., 2016). AlexNet (Krizhevsky et al., 2012) was chosen because of its demonstrated capacity to classify a variety of items, including phones and driver's hands, which are useful for detecting distracted driving behaviors. Model A achieved a classification test accuracy of 96.8% on the SF dataset (Montoya et al., 2016). To increase the model's performance even more, Okon & Meng (2017) introduced an upgraded version (Model B) that incorporated triplet loss. This approach considerably improved classification accuracy, with Model B scoring 98.7% on the SF dataset(Okon & Meng, 2017).

Majdi et al. (2018) proposed 'Drive-Net' a supervised learning method that combines a CNN (O'shea & Nash, 2015) with a random decision forest (Ho, 1995) for driver distraction detection. Drive-Net's (Majdi et al., 2018) CNN architecture is derived from a modified U-Net (Ronneberger et al., 2015) model, where the up-convolution layers and the last two down-sampling layers are replaced by a 1x1 convolution layer. The random decision forest (Bosch et al., 2007) in Drive-Net comprises multiple decision trees trained with randomized data subsets and optimized by maximizing information gain at each node. Drive-Net was evaluated against a Recurrent neural network (RNN) (Liang & Hu, 2015) and a Multi-layer perceptron (MLP) (Haykin, 2009) using the SF dataset (Montoya et al., 2016). Utilizing k=5 in k-fold cross-validation (Hastie et al., 2009), Drive-Net achieved a 95% classification accuracy, outperforming RNN and MLP classifiers, which scored 91.7% and 82%, respectively(Majdi et al., 2018).

Janet et al. (2020) used CNN to detect driver distractions using the SF dataset. Janet et al. (2020) tested three models: a vanilla CNN, a vanilla CNN with data augmentation, and a CNN with transfer learning. The vanilla CNN, which included three convolutional layers and three dense layers, had the highest accuracy of 97.66%. The data-augmented vanilla CNN achieved an accuracy of 97.05%, while the transfer learning model, which used VGG (Simonyan & Zisserman, 2014) and MobileNet (Howard et al., 2017) to shorten training time, achieved 71.72% (Janet et al., 2020).

Dhakate & Dash (2020) introduced an ensemble of CNN for driver distraction detection on the SF dataset. The proposed approach entails training several CNN models, including VGG-16, VGG-19 (Simonyan & Zisserman, 2014), InceptionV3 (Szegedy et al., 2016), ResNet-50 (He et al., 2016a), and Xception (Chollet, 2017), by eliminating their final layers to extract feature vectors. These vectors are then combined using a stacking ensemble technique to train a meta-classifier CNN, achieving a classification accuracy of 97%. The ensemble stacking technique blends the outputs of different base-level models, enhancing overall prediction accuracy. The best-performing ensemble model in (Dhakate & Dash, 2020) , a combination of ResNet-50 (He et al., 2016a), Xception (Chollet, 2017), InceptionV3 (Szegedy et al., 2016), and VGG-19 (Simonyan & Zisserman, 2014), achieved 97% accuracy, significantly outperforming a simpler ensemble model of Xception and InceptionV3, which achieved 73% accuracy on the SF dataset (Dhakate & Dash, 2020).

Huang et al. (2020) developed a hybrid CNN framework (HCF) aimed at detecting distracted driving behaviors on the SF dataset. This framework integrated three pretrained models—ResNet50, Inception V3, and Xception—to extract comprehensive behavior features through cooperative transfer learning. The extracted features were integrated into a detailed feature set, which is then classified using fully connected layers. Huang et al. (2020) employed an improved dropout algorithm to prevent overfitting and class activation mapping (CAM) to highlight key features. HCF framework achieved a classification accuracy of 96.74% on the SF dataset (Huang et al., 2020).

Qin et al. (2021) proposed D-HCNN, a CNN with decreasing filter fize, for real-time distracted driving detection. Qin et al. (2021) focused on building a highly accurate, fast, and low parameter count based model. As a result, D-HCNN uses only 0.76M parameters, and incorporates Histogram of Oriented Gradients (HOG) (Dalal & Triggs, 2005) images, L2 regularization, dropout, and batch normalization. D-HCNN begins with large convolution filters to capture broad features and progressively reduces filter sizes for detailed feature extraction. The D-HCNN architecture includes four convolutional layers with decreasing filter sizes (12x12, 9x9, 6x6, 3x3), followed by ReLU (Agarap, 2018), max-pooling (Fukushima, 1988), batch normalization (Ioffe & Szegedy, 2015), and dropout layers (Srivastava et al., 2014), concluding with global average pooling (Lin et al., 2013) and softmax classification. Qin et al. (2021) converts RGB images to grayscale in the proposed D-HCNN model to mitigate lighting effects and reduce computation. It also employs zero-mean normalization and random cropping for data augmentation. Evaluated on the AUC Distracted Driver (AUCD2) (Abouelnaga et al., 2017a) and SF datasets, D-HCNN achieved accuracies of 95.59% and 99.87%, respectively.

Li et al. (2022b) introduced "OLCMNet" a lightweight octave-like CNN (Chen et al., 2019), for detecting driver distraction. Li et al. (2022b) developed an octave-like convolution mixed block (OLCM) to efficiently process feature maps by separating them into low and high-frequency branches, followed by global information fusion through squeeze-and-excitation (SE) (Hu et al., 2018) modules. The architecture consists of a head, feature extraction, and final stage, with the OLCM block reducing spatial redundancy and improving computational efficiency. Li et al. (2022b) also created Lilong Distracted Driving Behavior (LDDB) dataset, containing 267,378 annotated images from on-road experiments. The proposed OLCMNet achieved 95.98% accuracy when evaluated on the (LDDB) dataset and 89.53% accuracy on the SF dataset. Li et al. (2022b) highlighted that squeeze-and-excitation module (Hu et al., 2018) in the final stage enhanced information exchange between layers, resulting in higher classification accuracy.

The aforementioned research demonstrates the effectiveness of various supervised learning techniques and CNN structures, including simple CNNs, transfer learning models, hybrid frameworks, and ensemble methods, in accurately detecting driver distractions on conventional datasets such as SF and AUCD2 (Abouelnaga et al., 2017a). However, there are notable drawbacks linked to using supervised learning methods for the driver distraction detection task.

Drawbacks of Supervised Learning Based Methods: The aforementioned research typically employs supervised CNN models, which demand large amounts of labeled data for training. This labeling process is resource-intensive, making it difficult to implement such models in real-world applications. Also these methods rely on manually selected features combined with classifiers and suffer from non-universality and poor adaptability to diverse driving scenarios (Li et al., 2022a; Zhang et al., 2023). The reliance on supervised frameworks, such as CNNs, limits their ability to generalize across different driving scenarios due to the need for extensive reliable labeled data. Furthermore, while CNNs are effective at learning local image features, they struggle with capturing the global context necessary for accurately detecting driver distractions in complex real-world environments (Li et al., 2022a; Zhang et al., 2023). This narrow focus hinders their overall perceptual ability, which is essential for comprehending spatial relationships and high-level semantic information in driving scenes (Zhang et al., 2023). Additionally, the intricate nature of real-world driving situations complicates the accurate labeling of data, thereby escalating the difficulty and cost of dataset creation . Overall, these factors contribute to the limited generalization performance and weak iterative ability of current supervised CNN-based models (Zhang et al., 2023).

2.2 Unsupervised and Self-Supervised Learning Based Methods

This section presents solutions to the constraints of methods that rely on supervised learning and highlights the advantages of unsupervised and self-supervised learning techniques for detecting driver distraction. Li et al. (2022a) introduced a novel unsupervised deep learning technique called "Unsupervised deep learning framework (UDL)" to address the constraints of current supervised methods in driver distraction detection. UDL harnesses vast quantities of unannotated data, hence increasing its applicability for industrial purposes (Li et al., 2022a). In order to enhance generalization, the Simsiam (Chen & He, 2021) unsupervised model was updated. Li et al. (2022a) incorporated a MLP design influenced by RepMLP (Ding et al., 2021). This architecture combines both local and global feature extraction methods. This method improved the model's capacity to adjust to different driving situations. Li et al. (2022a) also incorporated residuals into the projection head as a means of mitigating feature deterioration in multilayer architectures. This enhances the process of extracting deep features and improves the precision of detecting distractions (Li et al., 2022a). A novel loss function was developed by integrating comparative learning (Chen et al., 2020a) with the stop-gradient (Chen & He, 2021) technique. This loss function aims to improve the model's ability to learn robust features, hence boosting its generalization performance (Li et al., 2022a). The UDL technique underwent testing using the SF dataset, and it achieved a notable accuracy rate of 97.38% during linear evaluation. This performance surpassed that of other unsupervised models like SimSiam with ResNet50 backbone (86.29%), and SimCLR (Chen & He, 2021) with ResNet50 (94.32%) during linear evaluation (Li et al., 2022a). The UDL method offers a substantial improvement in identifying driver distraction by utilizing unsupervised learning. The approach optimizes the utilization of unlabeled data, enhances the process of extracting features, and exhibits exceptional performance and flexibility, effectively overcoming the constraints of supervised models (Li et al., 2022a).

Self-supervised models outperform supervised models in capturing high-level semantic information by focusing feature attention on important discrimination regions more efficiently (Zhang et al., 2023). Zhang et al. (2023) developed SL-DDBD, a self-supervised learning method for detecting driver distraction behavior. The method uses a masked image modeling framework to reduce labeling costs. Zhang et al. (2023) reconfigured the Swin Transformer (Liu et al., 2021) encoder, and utilized data augmentation strategies and optimal random masking strategies in the SL-DDBD method. The method achieved 99.60% accuracy on the SF dataset (Montoya et al., 2016), with pre-training and transfer learning for downstream driver distraction task.

Despite the numerous advantages of unsupervised learning and self-supervised learning over supervised learning, there have been limited research that have applied these methods for identifying driver distraction. Therefore, it is imperative to transition towards these learning paradigms in order to develop more resilient, dependable, and effective systems for detecting driver distraction.

2.3 Different Data Modalities

The success of machine learning models is significantly determined by the quality and characteristics of the data. Similarly, the reliability of driver distraction detection systems relies on the integration of several data modalities and their relationship, as well as their collective impact on overall effectiveness (Shajari et al., 2023). Several academics have examined driver distraction detection using various perspectives of data modalities. Physiological data offer direct insights into the driver's condition and correlate strongly with distraction levels (Reimer et al., 2011; Son & Park, 2021; Almahasneh et al., 2014; Lin et al., 2011). Visual data, crucial for the success of supervised algorithms, includes tracking eye movement and body posture. The precision of these models depends on the accuracy of data collection methods like electroencephalography (EEG), which require adjustments to reduce interference from external physiological activities (Lin et al., 2005; Lakshmi et al., 2014). Shajari et al. (2023) provided an overview of various types of data used in driver distraction detection research. These include physiological data (such as brain activity, breathing rate, skin conductivity and heart rate) and visual data (such as eye movement, body movement, and head movement). Shajari et al. (2023) emphasized the importance of combining different data modalities to improve the accuracy and reliability of driver distraction detection models.

2.4 Role of Drive and Act Dataset:

The availability and quality of datasets are critical in exploring solutions to driver distraction. Moslemi et al. (2021) summarized the research on driver distraction based on key datasets such as the SF (Montoya et al., 2016), and the American University in Cairo (AUC) (Abouelnaga et al., 2017a) datasets. These datasets vary in characteristics, presenting challenges like differing lighting conditions, camera angles, and the level of detail in recorded actions, which can affect the models' general applicability and effectiveness (Moslemi et al., 2021).

The DAA dataset, as detailed in (Martin et al., 2019), is an innovative resource carefully assembled to advance research in driver behavior detection during both manual and automated driving scenarios. This extensive dataset comprises over 9.6 million frames, encapsulating 12 hours of video footage. It systematically captures a broad spectrum of distracting behaviors by integrating diverse types of data such as color, depth, infrared, and 3D body pose information, as illustrated in Figure 2.1. Data collection employed six different camera angles, utilizing five Near-Infrared (NIR) cameras and one Kinect v2 camera, the latter used for capturing color images in three channels (RGB), as well as infrared and depth data. The NIR cameras in the dataset operate at a resolution of 1280 x 1024 pixels and a frame rate of 30 Hz, while the Microsoft Kinect camera records color videos at 950 x 540 pixels with a 15 Hz frame rate and captures infrared and depth data at 512 x 424 pixels at 30 Hz (Martin et al., 2019). The dataset was acquired using a stationary driving simulator, which was selected to ensure participant safety and effectively simulate various driving scenarios. This controlled environment mitigates risks associated with real-world driving and allows for consistent data collection across diverse conditions. Furthermore, the dataset benefits from including a heterogeneous group of participants varying in body size, driving experience, and familiarity with car automation technologies (Martin et al., 2019). This diversity enriches the dataset, capturing various driving behaviors and styles. The DAA dataset is randomly segmented into three groups based on the identity of the driver. This splitting approach ensures that data from the same individual is not present across multiple splits, preventing potential overfitting or data leakage Martin et al. (2019). Specifically, Martin et al. (2019) state:

> “for each split, we use the data of ten different identities for training, two for validation, and two for testing.” (Martin et al., 2019)

By splitting the dataset in this manner, the authors aim to create a challenging benchmark that evaluates the generalization capabilities of models trained on the DAA dataset. The three distinct splits allow for proper training, validation, and testing procedures, enabling robust evaluation of driver behavior recognition and driver distraction detection algorithms on unseen data from new individuals not present in the training set.

The DAA dataset uses a three-level hierarchy to detail driver interactions as depicted in the figures 2.3 and 2.1, providing a comprehensive framework for analyzing driver behavior under both manual and automated conditions. At the highest level of the hierarchy are the tasks, which outline broad scenarios that participants encountered during the data collection. These tasks, such as entering the car and switching to autonomous driving, set the context for the actions and are vital for simulating realistic driving environments. They also include potentially distracting situations anticipated with increased automation, like using a laptop. The mid-level in this hierarchy consists of mid-level actions. These are fine-grained activities that further break down the tasks into more specific behaviors but still maintain a clear semantic meaning. For example, while a task might involve using a laptop, the mid-level actions detail the individual activities involved, such as typing or browsing. At the most detailed level are atomic action units, which describe basic interactions within the driving environment without long-term semantic implications. These units are defined by a combination of action, object, and location—such as ‘reaching for a jacket in the left backseat’. This level provides the fundamental building blocks of driver behavior, capturing the minute detail of every interaction (Martin et al., 2019).

This hierarchical labeling not only enhances the granularity of behavioral analysis but also supports the development of sophisticated models that can predict and interpret diverse driving behaviors in real-time. By examining these interactions across different levels of abstraction, researchers can gain deeper insights into how drivers respond to various driving conditions and tasks, contributing to safer automotive technologies.

2.4.1 Advantages of the Drive and Act Dataset:

The DAA dataset provides different modalities, i.e., Color, Infra-Red, Depth and 3D skeleton data, which can be combined with each other to develop complex and reliable driver distraction detection systems. Given the imbalanced nature of the DAA video dataset, Martin et al. (2019) segmented the DAA dataset into three distinct partitions: Split 0, Split 1, and Split 2. They advocated for the training, evaluating, and testing deep learning models across these three splits. They recommended that the resultant metrics, such as balanced accuracy (Brodersen et al., 2010), be averaged to yield a statistically robust performance measure. This approach is a proposed remedy to counter the dataset’s imbalance. This method of dataset division can be effectively integrated with additional strategies to further mitigate the imbalance in the DAA dataset (Martin

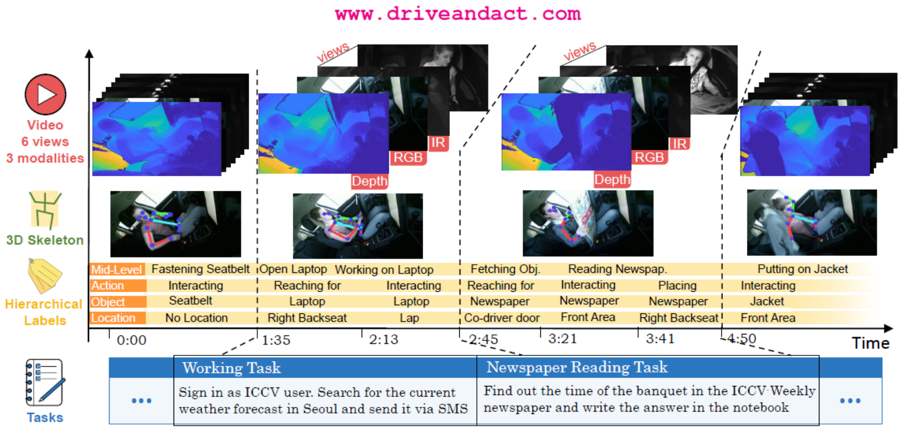


Figure 2.1: Drive and act dataset with its salient features. On the y-axis different modalities in the drive and act dataset can be seen along with the hierarchical labeling scheme guided by the 12 predefined tasks that each participant is subjected to perform in order to do the desired data collection. On the x-axis the time stamps along with the tasks that each participant in performing with respect to the time is shown. This figure also depicts the hierarchy in labeling where mid-level activities are fine grained activities and action, object and location combined together forms a complete driver action. Source: (Martin et al., 2019).

et al., 2019). This approach facilitates robust model evaluation across different data segments, ensuring statistical reliability in our findings.

2.4.2 Challenges of the Drive and Act Dataset

The distribution of all 83 driver actions provided by the DAA video dataset is depicted in figure 2.3. Each activity is captured for a duration of 3 seconds in one video sample (Martin et al., 2019). In the figure 2.3, the y-axis depicts the frequency of each activity, while the x-axis represents the 83 activities. It is evident from the figure 2.3 that an important obstacle in the DAA video dataset is the unequal distribution of classes among all 83 activities including the 34 fine-grained activities (Martin et al., 2019). This disparity is a substantial obstacle for creating models that can effectively identify rare but potentially vital driving behaviors. It is crucial to address this issue in order to develop strong models that can accurately detect a variety of actions, including some that are less frequent but may have greater potential for harm.

2.4.3 Comparing Drive and Act to Other Datasets

The DAA dataset (Martin et al., 2019) is distinguished from current datasets such as NTU (Shahroudy et al., 2016), HEH (Ohn-Bar & Trivedi, 2014), AUC (Abouelnaga et al., 2017b) and Kinetics (Carreira & Zisserman, 2017) by its large size, wide range of activities recorded, and inclusion of both manual and automated driving situations for Action Recognition. For example, the AUC dataset (Abouelnaga et al., 2017b) contains only 17,000 images, and the NTU dataset (Shahroudy et al., 2016) contains 4 million images, both of which are significantly smaller than the over 9.6 million images provided by the DAA dataset. The Kinetics (Carreira & Zisserman, 2017) contains more than 76 million images and is exception in terms of size, however it only offers one camera view compared to 6 camera views in DAA dataset. This comprehensive approach distinguishes it as a significant resource for the research community and highlights its potential to promote breakthroughs in driver behavior identification systems. However, there is an urgent require-

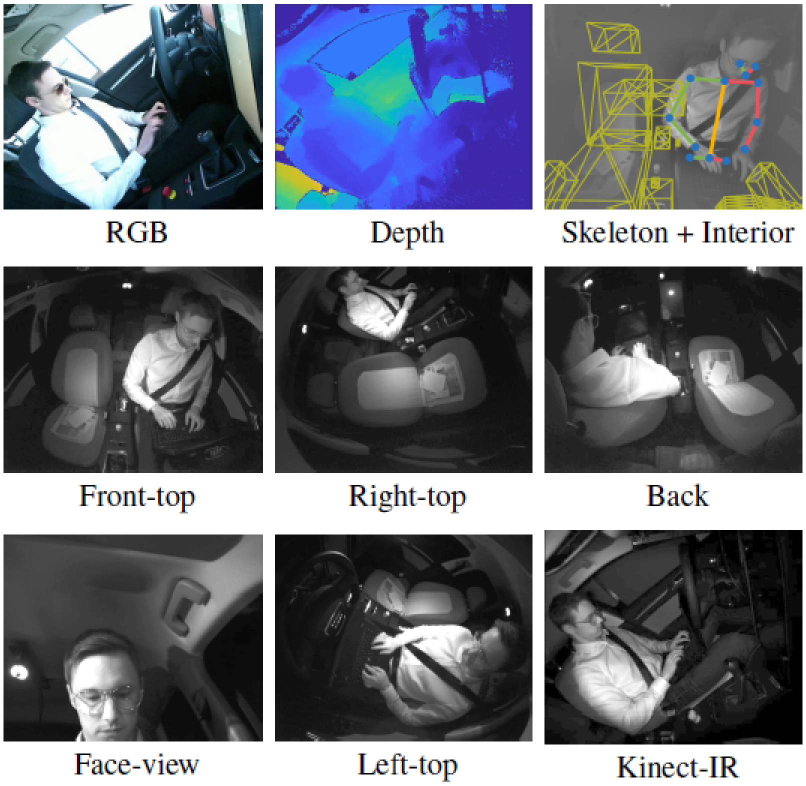


Figure 2.2: Illustrative images depicting the action of working on a laptop from various views and using different modalities. Source: (Martin et al., 2019).

ment for creative solutions to address the issues of class imbalance and to efficiently utilize the dataset's multi-modal and multi-view data. This dataset provides unparalleled opportunity for researchers in a broad range of domains, particularly in tackling the issue of imbalanced datasets. It enables the DAA dataset to be bench-marked against the most recent models in a variety of domains, including multi-class and binary classification, view and modality generalization, and the creation of hybrid approaches that use multi-modal technology. These cutting-edge technologies are essential for driver distraction detection, behavior tracking, and the creation of intelligent perceptual user interfaces. By leveraging cutting-edge technology across multiple modalities, this dataset sets the path for major developments in understanding and increasing driver safety and interaction.

In this thesis, we focus on two data modalities from the DAA dataset: Color and Infrared. An illustration of these modalities, as seen in the action of working on a laptop, is shown in Figure 2.2. The DAA dataset, derived from video captured with Kinect and NIR cameras, serves as the source of video data for this research. The transformation of this video data into image data is accomplished by extracting frames, a process guided by the dataset's annotation files across all splits. This thesis explores the challenges posed by the significant class imbalance found within these image datasets, a phenomenon that is often extrinsic He & Garcia (2009). For example, categorizing actions from the DAA dataset into distracted versus non-distracted

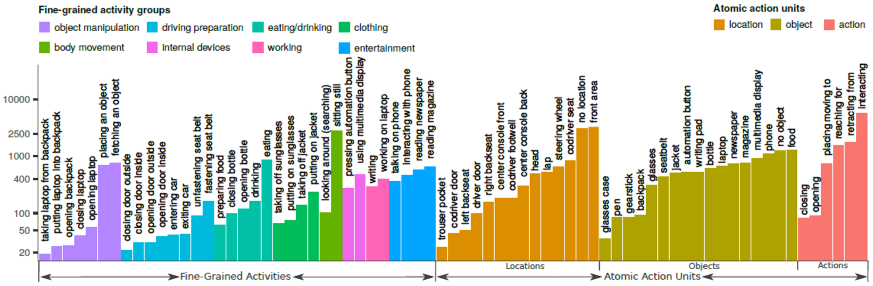


Figure 2.3: Distribution of Activities in the Drive and Act Video Dataset. The x-axis represents fine-grained activities, followed by atomic action units, while the y-axis shows the number of video samples, ranging from 0 to 10,000. Source: (Martin et al., 2019).

classes introduces an inherent imbalance due to varying sample sizes across the 34 fine-grained classes present in the mid-level hierarchical labeling of the DAA dataset, and can be seen in the figure 2.3. Chapter 4 provides a detailed analysis of the image datasets and tackle the difficulties caused by imbalanced datasets by proposing and explaining the 'Clustered Feature Weighting' algorithm.

2.4.4 Key Challenges Posed by Imbalanced Datasets

Researchers such as (He & Garcia, 2009; Johnson & Khoshgoftaar, 2019a), summarised the key challenges posed by imbalanced datasets as follows:

- **Model Bias:** Models trained on imbalanced data typically exhibit a bias towards the majority class, leading to insufficient representation and prediction of minority classes (Rawat & Mishra, 2022; He & Garcia, 2009).
- **Poor Generalization:** Models may generalize poorly on new, unseen data, especially for under-represented classes (He & Garcia, 2009; Johnson & Khoshgoftaar, 2019a).
- **Evaluation Challenges:** Conventional accuracy measurements can be false as they may primarily indicate the frequency of the dominant category rather than the actual prediction capacity of the model in various situations (He & Garcia, 2009; Johnson & Khoshgoftaar, 2019a).

2.5 Existing Solutions to Imbalanced Datasets

In machine learning, deep learning, and computer vision, learning from datasets that are imbalanced has become a major problem that makes it hard to get high-performance algorithmic results. An imbalanced dataset is characterized by an unequal distribution of classes, a topic that has been extensively explored in the existing literature (Fernández et al., 2018), (He & Garcia, 2009). This imbalance can develop either organically, as a result of the inherent differences in the frequency of data occurrence, as shown in medical diagnostics, or extrinsically, due to external factors such as the methods used to collect data (Johnson & Khoshgoftaar, 2019b), (He & Garcia, 2009)). According to Krawczyk (2016b), it is possible to achieve appropriate outcomes regardless of class imbalance, as long as both classes for example in binary classification are sufficiently represented and come from separate distributions (Johnson & Khoshgoftaar, 2019b).

Within the field of image classification, extensive study has been conducted on the issue of imbalance in both binary and multi-class frameworks, resulting in the development of numerous strategies for reducing imbalance problems (He & Garcia, 2009; Chawla et al., 2002; Han et al., 2005; He et al., 2008). However, the usefulness of these strategies varies depending on the application instance, with each approach having its own set of advantages and drawbacks. Traditionally, research has concentrated mostly on machine learning models (Akbani et al., 2004; Vilariño et al., 2005; Kang & Cho, 2006; Tang & Zhang, 2006). Nonetheless, current advances in the field of computer vision, natural language processing (Chen et al., 2021) and deep learning need a move toward investigating how deep learning models can tackle data imbalance problems or how efficient they are when exposed to imbalanced data (Johnson & Khoshgoftaar, 2019b). According to Japkowicz & Stephen (2002b), the severity of the imbalance problem depends on the degree of class imbalance, the complexity of data representation, the volume of training data, and the classification technique used (Kulkarni et al., 2021). The Imbalance Ratio (ImR) is a metric that measures the degree of class imbalance. It is calculated as the ratio of the number of samples in the majority class to the number of samples in the minority class (Fernández et al., 2018). With respect to binary image classification, the terms 'majority class' or 'negative class' refer to the class with the most samples, whereas 'minority class' or 'positive class' implies the class with the fewest examples (Kulkarni et al., 2021). The same terminology can be transferred to the domain of multi-class image classification problems.

In order to mitigate the consequences of imbalance, a number of measures are outlined in (Johnson & Khoshgoftaar, 2019a). The following are the main solutions:

- **Resampling Techniques:** The dataset can be balanced by oversampling minority classes or undersampling majority classes (He & Garcia, 2009; Johnson & Khoshgoftaar, 2019a).
- **Evaluation Metrics:** Adoption of metrics such as the balanced accuracy score, precision, recall, F-1 score, and the area under the Receiver Operating Curve (AUC-ROC) provide a more accurate measure of model performance in the context of imbalanced data (Johnson & Khoshgoftaar, 2019a; Wang et al., 2016).
- **Class Weight Adjustments:** During model training, adjusting class weights compensates for imbalances by assigning greater importance to minority classes within the loss function (Johnson & Khoshgoftaar, 2019a).

2.5.1 Learning from Imbalanced datasets

The seminal work "Learning from Imbalanced Data" by He & Garcia (2009) not only offers a thorough examination of the challenges posed by datasets characterized by under-representation and substantial class imbalances, but also provides practical recommendations for addressing these issues. This study elucidates the difficulties conventional machine learning techniques, which are typically designed for balanced class distributions, encounter in the face of pronounced class disparities. It underscores the inadequacy of traditional metrics, such as overall accuracy (Grandini et al., 2020) or error rate, in accurately evaluating the performance of algorithms in imbalanced learning scenarios. He & Garcia (2009) advocate for the adoption of more sophisticated evaluation methods, such as receiver operating characteristics (ROC) curves (Fawcett, 2006), precision-recall curves (Giglioni et al., 2021), and cost curves (Giglioni et al., 2021), which provide a more detailed insight into performance dynamics. Additionally, the paper highlights the challenges posed by relative imbalances—common in real-world settings—and the significant learning obstacles introduced by the scarcity of representative data for rare instances. Importantly, the authors acknowledge the complexity of datasets and relative imbalances as key contributors to the degradation of classification performance, further exacerbated by factors such as within-class imbalances and small disjuncts (He & Garcia, 2009). The authors have presented a comprehensive categorization of solutions available for addressing the issue of learning from imbalanced datasets, as depicted in figure 2.4. They have also provided detailed explanations for each category and method. Additionally, they have offered criticisms of commonly used solutions, such as sampling methods designed to balance datasets. These critiques address the limitations of undersampling and oversampling, such as the possibility of losing important information and the potential for overfitting. He & Garcia (2009) express significant criticism towards the Synthetic Minority Oversampling Technique (SMOTE) (Chawla et al., 2002) method, highlighting its vulnerability to overgeneralization and variance issues. Specifically, the SMOTE algorithm's tendency to generate synthetic data without careful consideration, which might unintentionally blur class boundaries or in other words obscure the distinction between different classes as originally highlighted by (Wang & Japkowicz, 2004). The blurring of class boundaries poses challenges for models in differentiating between classes due to the potential resemblance or encroachment of synthetic data points onto the data space of the majority class, or the inaccurate representation of the minority class. This could potentially diminish the efficacy of the model, as it may encounter difficulties in appropriately categorizing novel instances that lie in close proximity to these indistinct boundaries (Wang & Japkowicz, 2004).

Additionally, Krawczyk (2016a) categorize the strategies for addressing imbalanced datasets into three principal techniques. Initially, they examine data-level approaches, which involve adjustments to the sampling process either to balance the class distribution or to eliminate problematic samples. The second category encompasses algorithm-level strategies, which modify the core learning algorithms to better manage data with skewed distributions, thereby reducing the bias towards majority class instances (Krawczyk, 2016a; Fernández et al., 2018). Subsequently,Krawczyk (2016a) provide hybrid methods that combine the benefits of both data-level and algorithm-level approaches (Chakrabarty & Biswas, 2020).

Fernández et al. (2018) explored the challenges associated with learning from imbalanced datasets thoroughly in *Learning from Imbalanced Data Sets* (Fernández et al., 2018). Similarly, Johnson & Khoshgoftaar (2019a) provide a detailed review of methods to tackle class imbalance in machine learning, with an emphasis on deep learning techniques. Johnson & Khoshgoftaar (2019a) categorizes the various techniques, defines appropriate assessment criteria for imbalanced datasets, and underscores the significance of both supervised and unsupervised learning approaches, including those that incorporate transfer learning. Johnson & Khoshgoftaar (2019a) notes that traditional metrics like accuracy can misleadingly reflect performance in imbalanced contexts due to their susceptibility to the prevalence of the majority class (Fernández et al., 2018; Johnson & Khoshgoftaar, 2019a). Instead, Balanced Accuracy (BA) (Brodersen et al., 2010) is recommended as it accounts for both the True Positive Rate (TPR) and the True Negative Rate (TNR), offering a comprehensive measure of model efficacy across both majority and minority classes and mitigating the inherent bias of simpler metrics (Wang et al., 2016). Johnson & Khoshgoftaar (2019a) identifies a gap in the exploration of deep learning strategies for managing class imbalance, with most current methods, particularly those involving sampling or algorithmic modifications, falling within the domain of supervised learning due to their reliance on labeled data for adjusting class distributions or refining learning algorithms based on class-specific insights (Johnson & Khoshgoftaar, 2019a). The discourse also touches on innovative deep learning techniques such as dynamic sampling, two-phase learning, and enhancements involving novel loss functions and cost-sensitive learning (Johnson & Khoshgoftaar, 2019a). Additionally, the integration of transfer learning with deep learning strategies to enhance model performance on imbalanced datasets is discussed. Despite the prevalence of supervised methods, the field of unsupervised learning in relation to class imbalance remains under-explored and represents a promising area for future research.

2.5.2 Transfer Learning and Clustering-Based Techniques

Transfer learning is an advantageous approach in deep learning that can address the problem of class imbalance. By initially training on diverse and wide datasets and then fine-tuning the model on the unbalanced dataset, it is possible to strengthen the model's capacity to generalize its knowledge to new scenarios and enhance its overall performance (Johnson & Khoshgoftaar, 2019a). This approach is especially beneficial for classes that have a restricted number of examples, as conventional learning methods may lead to below-average model performance (Johnson & Khoshgoftaar, 2019a). The review paper discusses various methods,

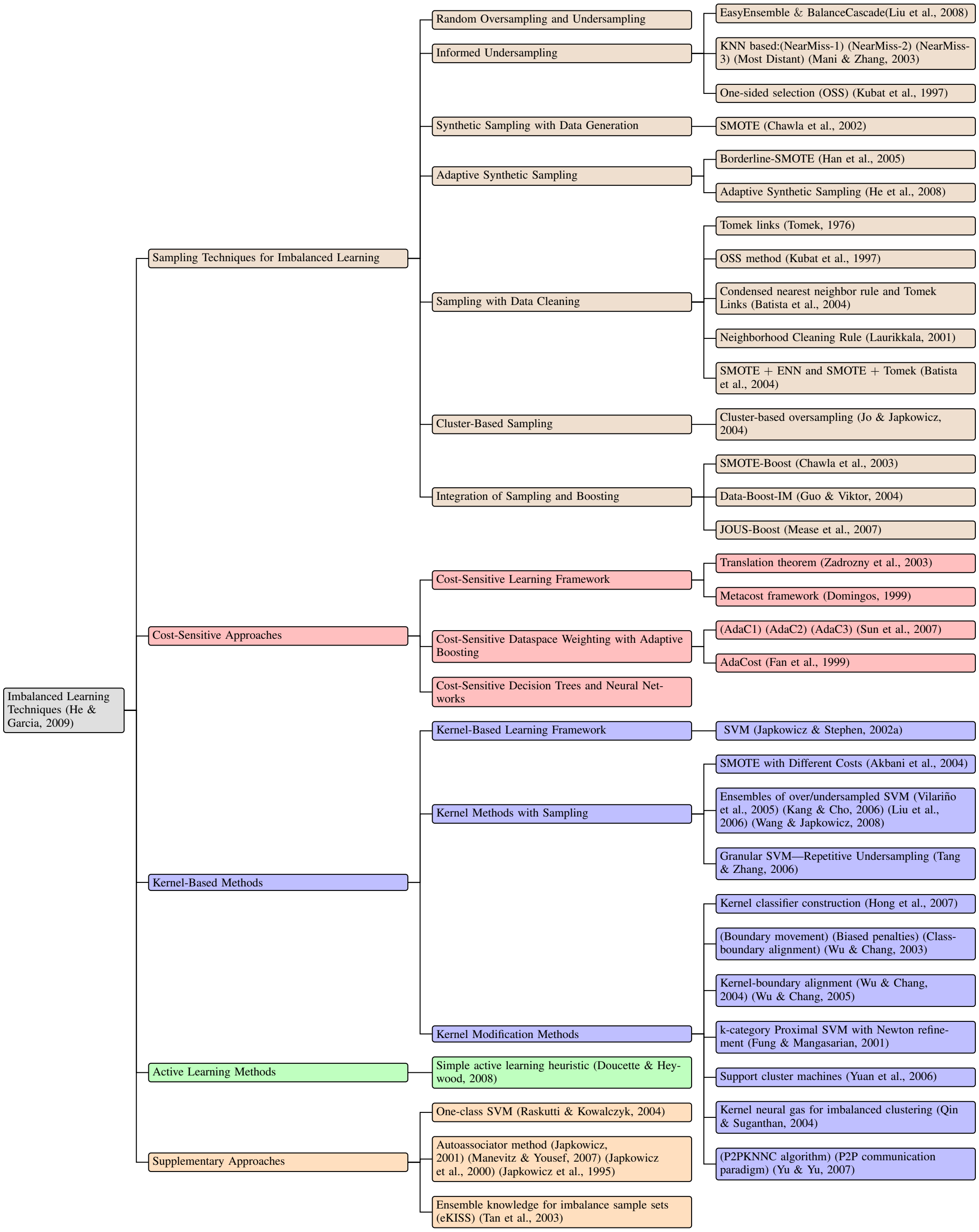

Imbalanced Learning Techniques (He & Garcia, 2009)
Sampling Techniques for Imbalanced Learning
Random Oversampling and Undersampling
EasyEnsemble & BalanceCascade(Liu et al., 2008)
Informed Undersampling
KNN based:(NearMiss-1) (NearMiss-2) (NearMiss-3) (Most Distant) (Mani & Zhang, 2003)
One-sided selection (OSS) (Kubat et al., 1997)
Synthetic Sampling with Data Generation
SMOTE (Chawla et al., 2002)
Adaptive Synthetic Sampling
Borderline-SMOTE (Han et al., 2005)
Adaptive Synthetic Sampling (He et al., 2008)
Sampling with Data Cleaning
Tomek links (Tomek, 1976)
OSS method (Kubat et al., 1997)
Condensed nearest neighbor rule and Tomek Links (Batista et al., 2004)
Neighborhood Cleaning Rule (Laurikkala, 2001)
SMOTE + ENN and SMOTE + Tomek (Batista et al., 2004)
Cluster-Based Sampling
Cluster-based oversampling (Jo & Japkowicz, 2004)
Integration of Sampling and Boosting
SMOTE-Boost (Chawla et al., 2003)
Data-Boost-IM (Guo & Viktor, 2004)
JOUS-Boost (Mease et al., 2007)
Cost-Sensitive Approaches
Cost-Sensitive Learning Framework
Translation theorem (Zadrozny et al., 2003)
Metacost framework (Domingos, 1999)
Cost-Sensitive Dataspace Weighting with Adaptive Boosting
(AdaC1) (AdaC2) (AdaC3) (Sun et al., 2007)
AdaCost (Fan et al., 1999)
Cost-Sensitive Decision Trees and Neural Networks
Kernel-Based Methods
Kernel-Based Learning Framework
SVM (Japkowicz & Stephen, 2002a)
Kernel Methods with Sampling
SMOTE with Different Costs (Akbani et al., 2004)
Ensembles of over/undersampled SVM (Vilariño et al., 2005) (Kang & Cho, 2006) (Liu et al., 2006) (Wang & Japkowicz, 2008)
Granular SVM—Repetitive Undersampling (Tang & Zhang, 2006)
Kernel Modification Methods
Kernel classifier construction (Hong et al., 2007)
(Boundary movement) (Biased penalties) (Class-boundary alignment) (Wu & Chang, 2003)
Kernel-boundary alignment (Wu & Chang, 2004) (Wu & Chang, 2005)
k-category Proximal SVM with Newton refinement (Fung & Mangasarian, 2001)
Support cluster machines (Yuan et al., 2006)
Kernel neural gas for imbalanced clustering (Qin & Suganthan, 2004)
(P2PKNNC algorithm) (P2P communication paradigm) (Yu & Yu, 2007)
Active Learning Methods
Simple active learning heuristic (Doucette & Heywood, 2008)
Supplementary Approaches
One-class SVM (Raskutti & Kowalczyk, 2004)
Autoassociator method (Japkowicz, 2001) (Manevitz & Yousef, 2007) (Japkowicz et al., 2000) (Japkowicz et al., 1995)
Ensemble knowledge for imbalance sample sets (eKISS) (Tan et al., 2003)

Figure 2.4: A flow chart showing the classification of Imbalanced Learning Techniques proposed by He & Garcia (2009).

such as category centers (CC) proposed by Zhang et al. (2018), Large Margin Local Embedding (LMLE) introduced by Huang et al. (2016), and Deep Over Sampling (DOS) developed by Ando & Huang (2017). These methods utilize hybrid approaches that combine transfer learning, deep feature representations, and k-nearest neighbors to tackle imbalanced datasets (Johnson & Khoshgoftaar, 2019a).

Clustering-based methods have become a sophisticated approach to tackle class imbalance by focusing on grouping instances into clusters before applying subsampling methods. This strategy aims to preserve the integrity of information while simultaneously providing a more equitable distribution among different classes (Munguía Mondragón et al., 2023). More precisely, algorithms such as K-means (Kaufman & Rousseeuw, 2009) and Clustering Large Applications (CLARA) (Kaufman & Rousseeuw, 2009) aid in the creation of clusters within classes by choosing instances that are both representative and diverse. A significant progress in this field is the utilization of density-based clustering methods, specifically Density-Based Spatial Clustering of Applications with Noise (DBSCAN) (Ester et al., 1996) and Hierarchical Density-Based Spatial Clustering of Applications with Noise (HDBSCAN) (Campello et al., 2013). These algorithms excel at handling datasets with varying sizes, shapes, and densities by detecting regions with high population density. This allows for reducing the size of the majority class without significant loss of information (Munguía Mondragón et al., 2023).

Munguía Mondragón et al. (2023) makes a comprehensive contribution by presenting a new density-based undersampling technique that exploits the capabilities of DBSCAN and HDBSCAN, in addition to an improved Multi-class imbalanced data classification using SMOTE and cluster-based undersampling (SCUT) (Agrawal et al., 2015) algorithm. This approach, when used with imbalanced and multi-class hyperspectral remote sensing images, employs geometric mean values and Friedman's test (Friedman, 1940) to thoroughly assess its effectiveness. The results emphasize the effectiveness of density-based clustering methods in addressing the difficulties posed by severe class imbalances, representing a notable shift from conventional strategies (Munguía Mondragón et al., 2023).

Munguía Mondragón et al. (2023) introduces methodological advancements that suggest a more precise equilibrium between undersampling and oversampling, depending on the average size of each class sample. This strategy effectively combines density-based clustering with the SCUT method to dynamically modify class sizes. It utilizes DBSCAN (Ester et al., 1996) or HDBSCAN (Campello et al., 2013) for classes that are above the average and employs SMOTE (Chawla et al., 2002) for classes that are below the average. This strategy seeks to achieve both a more equitable representation of classes and improved classifier performance by utilizing geometric mean values as a measure of success. The suggested method effectively addresses the issue of class imbalance by deliberately removing instances from the majority class using density-based clustering and increasing the representation of the minority class through SMOTE. The empirical validation of this method, which shows significant improvements in the accuracy of classifying highly unbalanced datasets, highlights the potential of combining density-based clustering with sampling approaches as an effective solution to the challenges caused by class imbalance.

The research by Munguía Mondragón et al. (2023) contends that conventional oversampling techniques such as SMOTE and its variations aim to achieve balance in the dataset, but they may not adequately address the nuances of class imbalance. On the other hand, clustering methods, especially those that rely on density, provide a strong and reliable alternative.

Research Gap: The emergence of Vision Transformer (ViT) models has had a tremendous impact on the field of computer vision. Despite this advancement, the use of these models to address the common issue of dataset imbalance has not been widely examined. This oversight signifies a crucial research gap, neglecting the potential of ViT models to serve as general-purpose tools for extracting intricate image features. Vision Transformers are distinguished by their ability to assimilate complex visual representations, which provides access to a rich feature space for developing novel techniques to handle the imbalanced learning challenge. This feature space, strengthened by training on large and diverse image datasets, has a wide range of visual properties. Such a collection of information is useful in finding both tiny differences and similarities between images, presenting a viable solution to the imbalance problem. It offers a thorough understanding and representation of minority classes, which are typically excluded and ignored by standard models. The comprehensive representation offered by ViT models opens up opportunities for the advancement of sampling strategies, augmentation techniques, and customized loss functions that are particularly crafted to address the distinct difficulties presented by unbalanced datasets. By innovating techniques that exploit the feature space of Vision Transformers (Dosovitskiy et al., 2020), there lies the potential to inaugurate a new phase of progress in deep learning. This advancement is particularly crucial in achieving equitable and consistent model performance across all class categories within imbalanced datasets.

Within the context of this thesis, Chapter 4 focuses on introducing and explaining such a novel sampling technique called "ClusteredFeatureWeighting" that combines the effective use of feature space of ViT using Transfer Learning, Density-Based Clustering, and Weighted Random Sampling. This novel approach is specifically tailored to rectify the imbalance dataset problem inherent in the drive and act dataset (Martin et al., 2019). Subsequent chapters, particularly Chapters 3 and 4, provide a comprehensive explanation of the working principles and experimental methodologies. Chapter 5 presents the results after each experiment by employing the methodology explained in the chapter 4.

Chapter 3

Background

We looked at the study on driver distraction detection and dataset imbalance in the last chapter. Following the research gaps identified in the previous section and the recommended research alternatives, this chapter will look at the background knowledge and working principles of the algorithms and models employed in this thesis to address the research problems. It covers the working principle of the HDBSCAN clustering algorithm (Campello et al., 2013) and vision transformer Dosovitskiy et al. (2020). Also discussed are the Self-Distillation with no labels (DINO) (Caron et al., 2021) framework and it's improved version DINOv2 (Oquab et al., 2023) for training vision transformer models.

3.1 HDBSCAN Clustering Algorithm

Clusters can be represented as dense regions in the data space, divided by sparse areas. Density-based clustering algorithms use this strategy to identify non-spherical groupings (Han et al., 2011). Density-based clustering detects clusters in data by separating regions of high and low point density (Han et al., 2011). DBSCAN (Ester et al., 1996) is a type of density based clustering algorithm which groups points that are closely packed together, marking points that are in low-density regions as noise. This method is particularly effective for discovering clusters of arbitrary shapes and sizes in datasets with noise (scikit-learn contrib/hdbscan, 2024). While DBSCAN (Ester et al., 1996) is powerful, it has limitations. It requires two parameters: *eps* (maximum distance between neighbors) and *min_samples* (minimum number of points to form a dense region) (Ester et al., 1996; scikit-learn contrib/hdbscan, 2024). Selecting suitable values for these variables might prove problematic, especially for datasets with varying density (scikit-learn contrib/hdbscan, 2024). Moreover, DBSCAN (Ester et al., 1996) does not inherently provide a way to explore the hierarchical structure of clusters (scikit-learn contrib/hdbscan, 2024).

HDBSCAN (Hierarchical Density-Based Spatial Clustering of Applications with Noise) (Campello et al., 2013) extends DBSCAN (Ester et al., 1996) by addressing these limitations. It combines density-based clustering with hierarchical clustering, allowing it to handle data with varying density more effectively and revealing a hierarchy of clusters (scikit-learn contrib/hdbscan, 2024).

Working of HDBSCAN algorithm: Based on the (Campello et al., 2013; 2015) papers, the working of HDBSCAN algorithm can be divided into following steps:

1. **Transforming the Space**:
    - HDBSCAN begins by computing the core distance for each point, which represents the distance to its k-th nearest neighbor (where k is defined by the *min_samples* parameter).
    - It then calculates the mutual reachability distance between pairs of points. This distance accounts for the core distance of the points, ensuring clusters can be identified in regions with varying density.
2. **Building the Minimum Spanning Tree (MST)**:
    - Using the mutual reachability distances, HDBSCAN constructs a minimum spanning tree (MST). This tree connects all points in the dataset with the shortest possible total distance, ensuring every point is reachable from any other point (scikit-learn contrib/hdbscan, 2024).
3. **Condensing the Tree**:
    - The MST is condensed into a hierarchy of connected components by progressively removing the edges with the longest mutual reachability distances. This process reveals a tree structure, or dendrogram, that represents clusters at different density levels (scikit-learn contrib/hdbscan, 2024).
4. **Extracting Stable Clusters**:
    - From the dendrogram, HDBSCAN uses the Excess of Mass (EOM) method to extract clusters. EOM identifies the most stable clusters by measuring their persistence across different density levels. Stable clusters are those that remain consistent over a range of scales, indicating they are meaningful groupings of data points (scikit-learn contrib/hdbscan, 2024).
5. **Outlier Detection**:
    - Points that do not belong to any stable cluster are labeled as noise. These points are in low-density regions and do not fulfill the requirements to be considered as part of a cluster (Campello et al., 2013; 2015).

Key Parameters of HDBSCAN:

- **Minimum Cluster Size (e.g., 25)**: Sets the smallest allowable size for a cluster. Clusters smaller than this are considered noise.
- **min_samples (e.g., 1)**: Specifies the minimum number of points required within a neighborhood for a point to be classified as a core point, hence affecting the computations of local density (Campello et al., 2013; 2015; scikit-learn contrib/hdbscan, 2024).
- **cluster_selection_epsilon (0.0)**: Controls the sensitivity for cluster formation. A value of 0.0 applies the strictest criteria.
- **metric ('euclidean')**: Specifies the distance metric, with 'euclidean' being the default for calculating distances between points (scikit-learn contrib/hdbscan, 2024).
- **cluster_selection_method ('eom')**: Determines how clusters are selected from the hierarchical tree, with 'eom' favoring stable, persistent clusters.

HDBSCAN (Campello et al., 2013) improves upon traditional density-based clustering by combining it with hierarchical clustering techniques. It enables the HDBSCAN algorithm to efficiently process datasets with different levels of density, detect clusters of any shape, and identify significant clusters more reliably than techniques such as DBSCAN (Ester et al., 1996). Through its use of core distances, mutual reachability distances, and hierarchical extraction methods, HDBSCAN (Campello et al., 2013) provides a powerful tool for clustering complex data. While HDBSCAN is relatively less affected by parameter settings compared to other clustering algorithms, it still necessitates the configuration of parameters such as ‘min_cluster_size’ and ‘min_samples’, which can impact the clustering results and is a limitation of this algorithm (scikit-learn contrib/hdbscan, 2024; Campello et al., 2015).

3.2 Vision Transformer

The ViT (Dosovitskiy et al., 2020) has transformed the computer vision domain by demonstrating that CNNs (O’shea & Nash, 2015) are not necessary for achieving high performance in image classification tasks. ViT applies the Transformer architecture (Vaswani et al., 2017), originally intended for sequential data in Natural Language Processing (NLP) (Eisenstein, 2019), to process images by considering them as sequences of patches, akin to tokens in text.

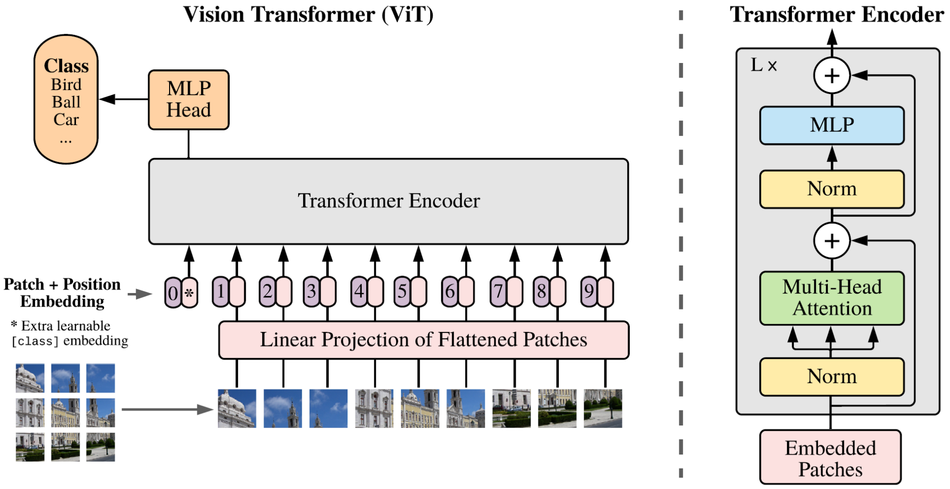


Figure 3.1: Vision Transformer Architecture from (Dosovitskiy et al., 2020).

Vision Transformer Architecture: The Vision transformer architecture consists of key elements including image patches, patch and positional embedding, learnable class embedding, Transformer Encoder, and MLP Head as shown in figure 3.1. Image patches are embedded linearly, with position embeddings for spatial information preservation. The transformer encoder is composed of layers that alternate between multi-headed self-attention and MLP blocks. LayerNorm (LN) is applied before each block and residual connections are applied after each block. The MLP blocks have two layers with a GELU non-linearity to maintain local and translational equivariance. The multi-headed self-attention layers record global dependencies across the image (Dosovitskiy et al., 2020). The MLP Head converts learned features into class output, sometimes it is also referred as the classifier head in classification tasks.

Mathematical Functions and Terminology: This paragraph adheres to the notations and conventions established in the Vision Transformer (ViT) paper by Dosovitskiy et al. (2020). All descriptions and mathematical formulations presented herein are based on those detailed in the original paper.

The process includes transforming image patches into embeddings, adding positional embeddings and a class token, and then passing them through the Transformer encoder, which entails a sequence of essential mathematical operations necessary for understanding the functionality of ViT. The components include linear transformations for patch embedding, softmax normalization in the self-attention mechanism, and the GELU (Hendrycks & Gimpel, 2016) non-linearity in MLP blocks.

Dosovitskiy et al. (2020) considered an RGB image $\mathbf{x}$ with dimensions $H \times W \times C$, where H represents Height, W represents Width, and C represents the color channel of the image. The image has a resolution of $H \times W$. Dosovitskiy et al. (2020) converted this image into a series of 2D image patches by dividing it into N patches with a resolution of P^2.

Dosovitskiy et al. (2020) mathematically transformed an image $\mathbf{x}$ into a 2D sequence of image patches $\mathbf{x}_p$, as shown in equation 3.1.

$$\mathbf{x} \in \mathbb{R}^{H \times W \times C} \to \mathbf{x}_p \in \mathbb{R}^{N \times (P^2 \times C)} \tag{3.1}$$

The lowercase letter “p” represents the patch. The number of patches, N, obtained from this transformation is determined by dividing the resolution of the original image by the resolution of each patch. Dosovitskiy et al. (2020) expressed N as the product of H and W divided by P squared as given in equation 3.2.

$$N = \frac{H \times W}{P^2} \tag{3.2}$$

Afterwards, Dosovitskiy et al. (2020) turned the two-dimensional patch sequence into a one-dimensional sequence using linear projections to align with the input requirements of the conventional Transformer encoder (Vaswani et al., 2017), which only accepts one-dimensional token embeddings. The linearly projected embedding is referred to as Patch embedding and is trainable.

According to Dosovitskiy et al. (2020), the initial patch embeddings and positional encoding of an input image in the Vision Transformer architecture can be defined as:

$$\begin{aligned} \mathbf{z}_0 = &\left[\mathbf{x}_{\text{class}}; \mathbf{x}_p^1\mathbf{E}; \mathbf{x}_p^2\mathbf{E}; \cdots; \mathbf{x}_p^N\mathbf{E}\right] \\ &+ \mathbf{E}_{\text{pos}}, \quad \mathbf{E} \in \mathbb{R}^{\left(P^2 \cdot C\right) \times D}, \quad \mathbf{E}_{\text{pos}} \in \mathbb{R}^{(N+1) \times D} \end{aligned} \tag{3.3}$$

In equation 3.3, $\mathbf{z}_0$ is the initial embedding matrix input into the Transformer encoder. The image is initially divided into N flattened 2D patches, indicated by x_p^i, where i denotes the count of patches from 1 to N. These patches are then linearly projected into a D-dimensional embedding space using a trainable matrix E (Dosovitskiy et al., 2020). The embedding matrix E is shared by all patches, representing the idea that each patch is equivalent to a ‘word’ in NLP tasks. Position embeddings $\mathbf{E}_{\text{pos}}$ are used to maintain positional information, which is crucial given the Transformer’s permutation invariance (Dosovitskiy et al., 2020). A specific class embedding $\mathbf{x}_{\text{class}}$ is prepended to the sequence to serve as a proxy for the overall image representation.

Each Transformer encoder layer comprises a self-attention mechanism Vaswani et al. (2017), succeeded by a MLP. Mathematically, the self-attention mechanism along with residual connection (He et al., 2016b) for layer ℓ is represented as:

$$\mathbf{z}'_\ell = \text{MSA}\left(\text{LN}\left(\mathbf{z}_{\ell-1}\right)\right) + \mathbf{z}_{\ell-1}, \quad \ell = 1 \ldots L \tag{3.4}$$

In the vision transformer architecture, ℓ is a number between 1 and L, where L is the maximum number of stacked transformer encoders. The implementation of the multiheaded self-attention (MSA) function on the layer-normalized embeddings from the preceding layer $\mathbf{z}_{\ell-1}$, is illustrated in equation 3.4. Subsequently, the result of this combined operation, is added in $\mathbf{z}_{\ell-1}$ as a residual connection (He et al., 2016b), thereby enabling gradient flow (Dosovitskiy et al., 2020). The self-attention mechanism's output $\mathbf{z}'_{\ell}$ is subsequently passed through a MLP in the Transformer encoder block as described in equation 3.5:

$$\mathbf{z}_{\ell} = \mathrm{MLP}\left(\mathrm{LN}\left(\mathbf{z}'_{\ell}\right)\right) + \mathbf{z}'_{\ell}, \quad \ell = 1 \ldots L \tag{3.5}$$

The output of the Transformer encoder block is the result of combining the output of the MLP with the output of residual connection (Dosovitskiy et al., 2020). The MLP comprises two dense layers with a GELU (Hendrycks & Gimpel, 2016) activation function.

Ultimately, equation 3.6 produces the ultimate output representation of the image, which is subsequently employed for classification purposes.

$$\mathbf{y} = \mathrm{LN}\left(\mathbf{z}_L^0\right) \tag{3.6}$$

The final image representation, denoted as y, is derived by applying layer normalization (Ba et al., 2016) to the layer that corresponds to the classification token. This representation y contains the aggregated information from all patches and their interactions as distilled through the various Transformer layers. The Vision Transformer's image classification approach replaces traditional convolutional operations with mechanisms that consider image patches as sequences of data points, akin to words in a sentence. This enables the model to learn contextual relationships throughout the image (Dosovitskiy et al., 2020).

Multi-headed Self-Attention Mechanism: The primary functionality of ViT relies on self-attention, enabling each patch to attend to every other patch in the image. This mechanism allows the model to prioritize the most pertinent regions of the image for the given task. The multi-headed self-attention Vaswani et al. (2017) (MHSA) improves the model's ability to understand complex spatial structures and connections by recording several visual components at the same time. The detailed process of self-attention mechanism can be found in paper Vaswani et al. (2017), while the mathematical explanation of the Multi-headed Self-Attention mechanism is given in Appendix A of the Vision transformer paper Dosovitskiy et al. (2020).

3.3 Evolution of Self-Supervised Learning in Vision Transformers: From DINO to DINOv2

DINO stands for "Self-Distillation with NO labels" (Caron et al., 2021). It is an innovative advancement in self-supervised learning for vision transformers. The DINO framework takes a novel approach to self-supervised learning by merging elements of self-training and knowledge distillation, which have previously been employed to increase feature quality by propagating annotations to unlabeled datasets (Caron et al., 2021; Hinton et al., 2015). Traditionally, knowledge distillation is teaching a smaller, simpler student network to mimic the behavior of a larger, pre-trained teacher network, therefore compressing knowledge into a more efficient model (Buciluă et al., 2006; Kim & Kim, 2017; Caron et al., 2021). DINO framework innovates by converting this method into what is known as "self-distillation," in which both the student and the teacher are trained concurrently during the learning phase, with no labeled data. In self-supervised vision transformers, label propagation involves both hard and soft assignments (Lee et al., 2013; Xu et al., 2020; Yalniz et al., 2019; Xie et al., 2020), with soft assignments (Xie et al., 2020) being particularly matched with knowledge distillation concepts (Buciluă et al., 2006; Hinton et al., 2015). This strategy has usually concentrated on model compression by training smaller networks to mimic the outputs of bigger ones, as proven by Xie et al. (2020).

DINO applies these notions to a self-supervised situation in which no true labels are available. Unlike prior techniques, which used a pre-trained, fixed teacher (Chen et al., 2020b; Fang et al., 2021; Shen et al., 2021; Noroozi et al., 2018), DINO framework dynamically constructs the teacher during training, making knowledge distillation an inherent component of the learning objective rather than a post-processing step (Caron et al., 2021). This approach is similar to codistillation (Anil et al., 2018), in which both the student and teacher networks share the same architecture; however, in DINO, the teacher network is updated using an exponential moving average (Polyak & Juditsky, 1992) of the student's parameters rather than mutual distillation as depicted in the figure 3.2 (Caron et al., 2021).

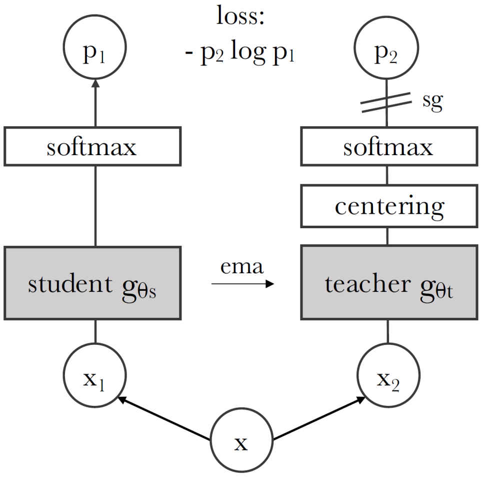


Figure 3.2: A diagram illustrating DINO with one image pair (x1, x2). The image undergoes two random transformations and is passed to both student and teacher networks, which have the same architecture but different parameters. The teacher's output is centered using the batch mean. Both outputs are normalized with temperature softmax and compared using cross-entropy loss. A stop-gradient operator is applied to the teacher, and its parameters are updated using an exponential moving average of the student's parameters (Caron et al., 2021).

The implementation of DINO, as depicted in the figure 3.2, employs two different random transformations of an input image, extracted from a single input image. These views are then processed by student and teacher networks, which have the same architecture but different parameters. The networks generate K-dimensional features that are normalized using a softmax function controlled by a temperature parameter. The similarity between these features is evaluated using cross-entropy loss. A stop-gradient operation is applied to the teacher to ensure that only the student is directly trained, enhancing the focus on feature generation without direct label dependence. For a comprehensive understanding of the implementation, interested readers can refer to section 3 of the DINO paper (Caron et al., 2021), which provides a complete mathematical foundation and explanation. This thesis focuses specifically on utilizing a pre-trained self-supervised vision transformer encoder to detect driver distraction on the DAA image dataset. The evaluation procedure used is a linear evaluation protocol.

The DINO architecture has significant implications for Vision Transformers (Dosovitskiy et al., 2020) since it has the ability to provide novel qualities to ViTs, which have primarily been compared to classical CNNs. The DINO framework aims to assess the potential benefits of self-supervised ViTs, including the ability to explicitly describe semantic segmentation and perform successful k-NN classification directly from the features Caron et al. (2021). In contrast to supervised ViTs and classical CNNs, which do not naturally prioritize certain characteristics, DINO approach stands out and provides remarkable accuracy on ImageNet (Russakovsky et al., 2015; Ridnik et al., 2021) dataset without the need for finetuning Caron et al. (2021).

Self-Distillation with NO labels Version 2 (DINOv2): Building on the fundamental ideas of DINO Caron et al. (2021), the architecture and training technique are enhanced and refined in various ways in DINOv2 Oquab et al. (2023). The architecture of DINOv2 signifies a significant progression in self-supervised learning in the field of computer vision. It integrates discriminative learning techniques to enhance feature extraction and improve model performance. The DINOv2 architecture is centered around a dual approach that consists of a student and a teacher network as introduced intially in the DINO paper Caron et al. (2021). This modified approach utilizes self-distillation techniques to promote strong feature learning without the need for labeled data Oquab et al. (2023).

The fundamental enhancements in the DINOv2 architecture consist of:

- **Discriminative Self-supervised Pre-training:** DINOv2 utilizes a novel combination of loss functions from DINO Caron et al. (2021) and Image bert pre-training with online tokenizer (iBOT) (Zhou et al., 2021), integrated with the centering mechanism from Swapping Assignments between Views (SwAV) (Caron et al., 2020). This configuration aids in stabilizing the feature space, preventing the trivial solutions often encountered in self-supervised learning scenarios (Oquab et al., 2023).
- **Dual-Objective Learning:**
  - **Image-level Objective Caron et al. (2021):** At this level, DINOv2 framework applies a cross-entropy loss between the outputs of the student and the teacher networks, derived from the class token of a Vision Transformer (Dosovitskiy et al., 2020) model. The outputs are obtained from different crops of the same image, encouraging the model to recognize and learn consistent features across varied perspectives and scales (Caron et al., 2021; Oquab et al., 2023).
  - **Patch-level Objective (Zhou et al., 2021):** This objective introduces variability in the input by masking some patches presented to the student network, but not to the teacher. A cross-entropy loss is then computed between the patch features from both networks. This added complexity ensures the model captures detailed textural and structural information at finer granularities, thus enhancing its overall sensitivity to image content (Oquab et al., 2023).
- **Adaptive Weight Management:** To optimize learning at different scales, DINOv2 employs a strategy of untying the head weights for the image-level and patch-level objectives. This approach corrects the tendency of the model to underfit at the patch level or overfit at the image level, thereby balancing the learning focus and improving outcomes at both scales (Oquab et al., 2023).
- **Sinkhorn-Knopp Centering (Caron et al., 2020):** The DINOv2 architecture adopts the Sinkhorn-Knopp (SK) algorithm for batch normalization (Caron et al., 2020) from SwAV framework, a method initially suggested for the DINO and iBOT framework by Ruan et al. (2022). This algorithm, recommended for its efficiency in normalizing features, replaces the softmax centering typically used in DINO and iBOT setups (Ruan et al., 2022). The SK centering is applied in three iterative steps, mainly for the teacher network, while the student employs softmax normalization. This element helps in maintaining a uniform distribution of features across batches, crucial for consistent self-supervised learning (Oquab et al., 2023).
- **Regularization and High-Resolution Training Phase:** In addition to these structural components, DINOv2 incorporates KoLeo regularizer (Sablayrolles et al., 2018) to ensure a diverse spread of features across the learning process and introduces a short high-resolution training phase. These

enhancements allow the model to adapt better to high-resolution tasks, which are particularly relevant in contexts such as detailed image analysis, segmentation, and object detection (Oquab et al., 2023).

The primary distinctions between DINO and DINOv2 are outlined below:

Loss Function Integration:

- DINO: Introduced knowledge distillation emphasizing feature invariance across various perspectives of the same image (Caron et al., 2021).
- DINOv2: Integrated a combination of DINO and iBOT loss functions, along with the centering techniques from SwAV, to stabilize the learning process further (Oquab et al., 2023).

Objective Functions:

- DINO: Primarily focused on image-level objectives to encourage feature learning across different augmentations of input data (Caron et al., 2021).
- DINOv2: Introduced patch-level objectives alongside image-level objectives, involving masking some image patches for the student but not for the teacher, adding complexity and encouraging more detailed feature learning (Oquab et al., 2023).

Weight Management:

- DINO: Did not explicitly address weight tying between different objectives.
- DINOv2: Improved the architecture by untying the weights between image-level and patch-level objectives, enhancing model performance across different scales and preventing overfitting or underfitting (Oquab et al., 2023).

Evaluation Protocols for SSL Models The evaluation of self-supervised learning SSL models, such as our DINOv2-based ViT encoder (Oquab et al., 2023), deviates substantially from conventional supervised approaches. The methods of evaluation for SSL models are essential for determining their effectiveness in downstream tasks like driver distraction detection, as these models are generally evaluated based on their capacity to generate valuable representations without direct supervision.

Types of SSL Evaluation:

1. **K-Nearest Neighbors (KNN):** In the context of SSL, a KNN classifier (Mucherino et al., 2009) uses l_2-normalized features extracted by the model to classify new images based on the closest training examples in feature space. This method is advantageous due to its simplicity, speed, and minimal hyperparameter tuning, making it an ideal quick benchmark for SSL models (Caron et al., 2021; Balestriero et al., 2023).
2. **Linear Evaluation:** The most popular method for evaluating SSL models is the linear evaluation, or linear probing. This method tests the quality of the backbone directly, as the linear classifier has limited capacity to adjust to the data, thereby providing a clear signal of the representational power of the underlying parameters. Typically, this involves appending a linear layer to the frozen backbone and optimizing it for several epochs (usually around 100), which is computationally efficient (Zhang et al., 2016; 2017; Balestriero et al., 2023; Bao et al., 2021).
3. **Full Fine-Tuning:** This method involves training the entire model (both the pre-trained backbone and the newly added classifier) on a downstream task such as driver distraction detection. It is the most thorough evaluation, allowing the model to fully adapt to the new task. However, it is also the

most computationally expensive and may not always correlate with the strength of the initial SSL pre-training, especially in scenarios where the downstream task is substantially different from the pre-training setup (He et al., 2021; Balestriero et al., 2023).

4. **Multi-Layer Perceptron (MLP) Probing:** While less common, MLP probing involves adding a small multi-layer perceptron on top of the frozen features. This can reveal whether the features are non-linearly separable, which may be masked by simpler linear probing. However, this approach is more prone to overfitting and typically requires careful management of model capacity and training duration (Bordes et al., 2023; Balestriero et al., 2023).

Chapter 4

Proposed Methods

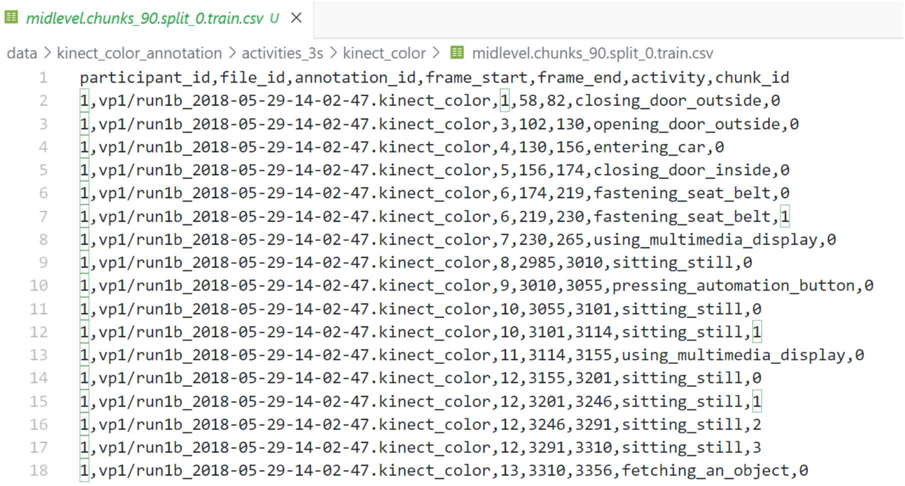
In Chapter 3, we explored clustering, vision transformers, and self-supervised learning frameworks like DINO and DINOv2, including their evaluation in various scenarios. Building on that, Chapter 4 outlines our proposed methods and the steps involved in our experiments. We start with how we created our image datasets using annotation files provided for DAA video dataset, including addressing dataset imbalance and relevant statistics. We introduce a new dataloader designed specifically for imbalanced datasets and describe our clustering process. The chapter also explains our methodologies for comparing new dataloader with traditional one and training and evaluating our models. Additionally, we provide necessary mathematical formulations to aid in understanding our evaluation principles, and we outline our experimental goals, which will be tested in the following chapter.

4.1 Image Dataset Generation

This section explains how we created image datasets from the DAA video dataset (Martin et al., 2019) using the provided annotation files. First, we'll examine the structure of these annotation files to understand how to extract frames from the DAA videos. We'll then go over the data pre-processing steps and frame extraction process. By the end of this section, we'll have prepared the extracted image datasets for an analysis of any imbalances in the next section.

Dataset Annotation Details: The annotation files are integral for indexing and extracting relevant frames. An example of such a file, "midlevel.chunks_90.split_0.train.csv", provided by DAA (Martin et al., 2019) video dataset, is depicted in the figure 4.1. It includes headers like participant_id, file_id, annotation_id, frame_start, frame_end, activity, and chunk_id. For instance, entries in this file indicate specific activities such as 'closing_door_outside' and 'opening_door_outside', with precise frame ranges provided for effective localization of the activity within the video because each activity is captured for 3 seconds and corresponds to 1 video sample (Martin et al., 2019).

Data Preprocessing and Frame Extraction: The methodology's initial step involves extracting image frames from the video files of the DAA dataset. This process forms foundational image datasets corresponding to each considered modality and camera view. Table 4.1 contains information about different modalities and views offered by DAA dataset Martin et al. (2019). Specifically, two camera views are utilized for this research: the 'Right Top View' and the 'Front Top View' as depicted in the table 4.2. Two 'Right Top View' datasets come from color and infrared videos captured by the Kinect camera and one 'Front-top


midlevel.chunks_90.split_0.train.csv U ×
data > kinect_color_annotation > activities_3s > kinect_color > midlevel.chunks_90.split_0.train.csv

```
participant_id,file_id,annotation_id,frame_start,frame_end,activity,chunk_id
1,vp1/run1b_2018-05-29-14-02-47.kinect_color,1,58,82,closing_door_outside,0
1,vp1/run1b_2018-05-29-14-02-47.kinect_color,3,102,130,opening_door_outside,0
1,vp1/run1b_2018-05-29-14-02-47.kinect_color,4,130,156,entering_car,0
1,vp1/run1b_2018-05-29-14-02-47.kinect_color,5,156,174,closing_door_inside,0
1,vp1/run1b_2018-05-29-14-02-47.kinect_color,6,174,219,fastening_seat_belt,0
1,vp1/run1b_2018-05-29-14-02-47.kinect_color,6,219,230,fastening_seat_belt,1
1,vp1/run1b_2018-05-29-14-02-47.kinect_color,7,230,265,using_multimedia_display,0
1,vp1/run1b_2018-05-29-14-02-47.kinect_color,8,2985,3010,sitting_still,0
1,vp1/run1b_2018-05-29-14-02-47.kinect_color,9,3010,3055,pressing_automation_button,0
1,vp1/run1b_2018-05-29-14-02-47.kinect_color,10,3055,3101,sitting_still,0
1,vp1/run1b_2018-05-29-14-02-47.kinect_color,10,3101,3114,sitting_still,1
1,vp1/run1b_2018-05-29-14-02-47.kinect_color,11,3114,3155,using_multimedia_display,0
1,vp1/run1b_2018-05-29-14-02-47.kinect_color,12,3155,3201,sitting_still,0
1,vp1/run1b_2018-05-29-14-02-47.kinect_color,12,3201,3246,sitting_still,1
1,vp1/run1b_2018-05-29-14-02-47.kinect_color,12,3246,3291,sitting_still,2
1,vp1/run1b_2018-05-29-14-02-47.kinect_color,12,3291,3310,sitting_still,3
1,vp1/run1b_2018-05-29-14-02-47.kinect_color,13,3310,3356,fetching_an_object,0
```

Figure 4.1: This figure depicts the annotation structure in the "midlevel.chunks_90.split_0.train.csv" file provided by Martin et al. (2019) for the train dataset of the split 0 of the Kinect Color DAA video dataset.It includes headers like participant_id, file_id, annotation_id, frame_start, frame_end, activity, and chunk_id. These details are utilized to extract exact image frames corresponding to each activity in the DAA video dataset.

View' dataset comes from NIR camera recordings. Table 4.3 lists the different image datasets obtained as a results of frame extraction procedure on the DAA video dataset. Additionally, it includes details regarding the specific perspective captured in the dataset, as well as the designated name for the image dataset within the context of this thesis. The number of channels in the table 4.3 includes information about the number of channels in each image, where for gray scale images there are 3 channels in which each channel contains the same information or in other words one gray scale channel is repeated thrice. This thesis focuses on training models using the 'Kinect Right Top View' color DAA dataset and evaluating their performance on Infrared (Grayscale) datasets to assess their ability to generalize across different modalities and viewpoints.

Dataset Categorization: This thesis focuses on driver distraction, which necessitates a streamlined approach to data categorization. The extracted drive and act image datasets, consisting of 34 distinct midlevel activities that detail various driver behaviors, are further reorganised into two main classes: '_non_distracted' and 'distracted' driver. The '_non_distracted' class includes activities that indicate the driver's full attention to driving. For example, the 'sitting_still' activity is clearly aligned with a non-distracted state. Similarly, activities like 'entering_car' and 'exiting_car', which occur outside the active driving period, are also grouped under 'non_distracted'. Alternatively, these activities could be excluded altogether, allowing the analysis to focus strictly on a binary classification: 'sitting_still' versus all other activities.

In contrast, the 'distracted' class comprises the remaining 31 fine-grained activities depicted in figure 2.3, representing potential distractions from the driving task. This binary categorization simplifies the dataset, enhancing its compatibility with the PyTorch Image Folder class (PyTorch, 2024).

4.1.1 Dataset Statistics

After extracting the image datasets into 34 fine grained activities, there is evidence of significant disparities in class distribution within the resulted image datasets. Figure 4.2 illustrates the disparities among the 34

Table 4.1: Different views and modalities in Drive and Act Dataset. Source: (Martin et al., 2019)

Camera Type	Modality Type	View Type
NIR	Infra-red (Gray scale)	Front Top
NIR	Infra-red (Gray scale)	Right Top
NIR	Infra-red (Gray scale)	Back
NIR	Infra-red (Gray scale)	Face view
NIR	Infra-red (Gray scale)	Left Top
Kinect	Color (RGB)	Right Top
Kinect	Depth	Right Top
Kinect	Infra-red (Gray scale)	Right Top

Table 4.2: Different views and modalities from Drive and Act Dataset used in this thesis. Source: (Martin et al., 2019)

Camera Type	Modality Type	View Type
NIR	Infra-red (Gray scale)	Front Top
Kinect	Color (RGB)	Right Top
Kinect	Infra-red (Gray scale)	Right Top

fine-grained activities in 'split 0' of the 'Kincet Color Right Top Image DAA' train dataset. The dataset for this split contains a total of 259,865 images. Of these, the 'sitting_still' class alone comprises 78,227 images, which account for 30.10% of the dataset. This starkly contrasts with categories such as 'closing_door_outside,' which represents a mere 0.087% with only 226 images. Such imbalances highlight the challenges in training models that can accurately recognize less frequent activities. However, this view gives us multi-class classification perspective of the image datasets and we need to further analyse the statistics for the binary classification task 'non distracted' driver versus 'distracted' driver. The imbalance ratio (Buda et al., 2018) is used as a standard metric to calculate the imbalance with respect to minority and majority classes rather than class ratios with respect to whole dataset size.

Method to calculate the class ratios Suppose we have a dataset with 34 distinct classes. Let N be the total number of images in the train set of split 0 of the 'Kinect Color Right Top DAA' dataset, and let N_i be the number of images in each class C_i for $i = 1, 2, \ldots, 34$.

The class ratio R_i for each class C_i with respect to the total dataset is defined as:

$$R_i = \frac{N_i}{N} \tag{4.1}$$

which gives the proportion of elements in each class relative to the entire dataset (Johnson & Khoshgoftaar, 2019a; Buda et al., 2018).

To convert this class ratio into a proportional percentage, which expresses the proportion of each class as a percentage of the total dataset, we multiply the class ratio by 100 (Bennett et al., 2003). Thus, the proportional percentage P_i is defined as:

$$P_i = R_i \times 100 \tag{4.2}$$

This provides the percentage of the dataset that belongs to each class, facilitating easier comparison and visualization of class distribution.

Table 4.3: Different version of Drive and Act (DAA) Image Datasets formed in this thesis. Source: (Martin et al., 2019)

Camera and Modality Type	**View Type**	**Dataset Name**	**Number of Channels**
Near Infra-red (Gray scale)	Front Top	NIR Front Top Image DAA	1 x 3 (duplicated)
Kinect Color (RGB)	Right Top	Kinect Color Right Top Image DAA	3 (RGB)
Kinect Infra-red (Gray scale)	Right Top	Kinect IR Right Top Image DAA	1 x 3 (duplicated)

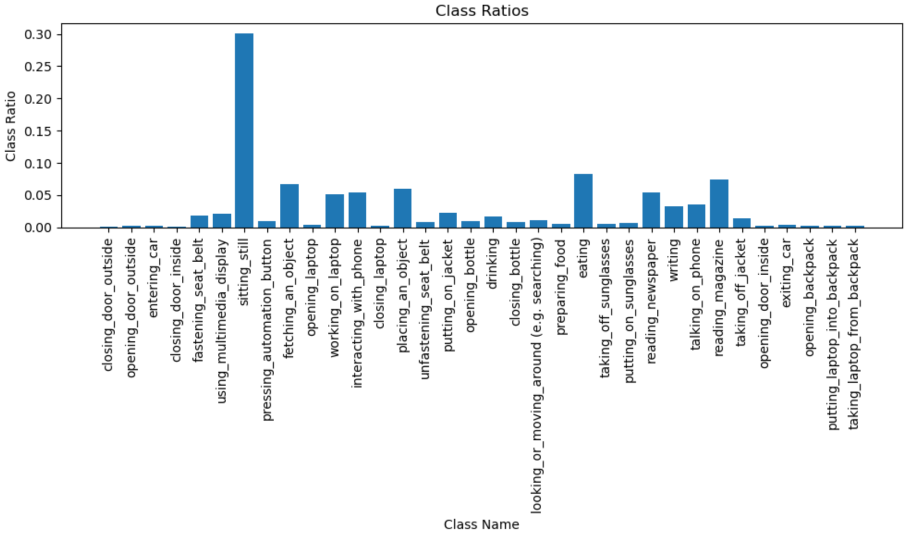


Figure 4.2: Illustrative image depicting the class imbalance in the split 0 of the Kincet Color right Top Image DAA train dataset with 34 fine grained activities. The y-axis represents the class ratios and the x-axis represents 34 fine-grained activities. This extracted image dataset can be used for multi-class classification tasks like driver action recognition.

Furthermore, for any two different classes C_i and C_j with N_i and N_j denoting the number of elements in each class , the pairwise class ratio R_{ij} can be defined as:

$$R_{ij} = \frac{N_i}{N_j} \tag{4.3}$$

which compares the relative sizes of any two classes.

For example, the class ratio for 'sitting_still' class can be calculated as follows:

$$R_i = \frac{78227}{259865} = 0.3010$$

And, the proportional percentage for 'sitting_still' class can be calculated as follows:

$$P_i = 0.3010 \times 100 = 30.10\%$$

4.1.2 Imbalance Across Different Splits and Classes

The datasets further contains the imbalance between ‘_non-distracted’ and ‘distracted’ driver classes across each dataset split. Figure 4.3 shows the distribution of ‘_non-distracted’ and ‘distracted’ driver classes in the Kinect color Right Top Image DAA Dataset. Tables 4.5 to 4.6, presents the imbalance ratio (ImR) for all generated image datasets. The Imbalance Ratio (ImR) is the ratio of image count for the majority class (‘distracted’) to the image count for the minority class (‘non-distracted’) (Johnson & Khoshgoftaar, 2019a; Buda et al., 2018). This metric shows the significant skew in data distribution, affecting the training and performance of detection models.

For a binary classification dataset, mathematically, the Imbalance Ratio can be written as follows:

$$ImR = \frac{N_{Majority}}{N_{Minority}} \tag{4.4}$$

where $N_{Majority}$ represents the total number of images in the majority class and $N_{Minority}$ represents the total number of images in the minority class (Johnson & Khoshgoftaar, 2019a; Buda et al., 2018).

Table 4.4: Imbalance in Kinect Color Right Top Image DAA Dataset

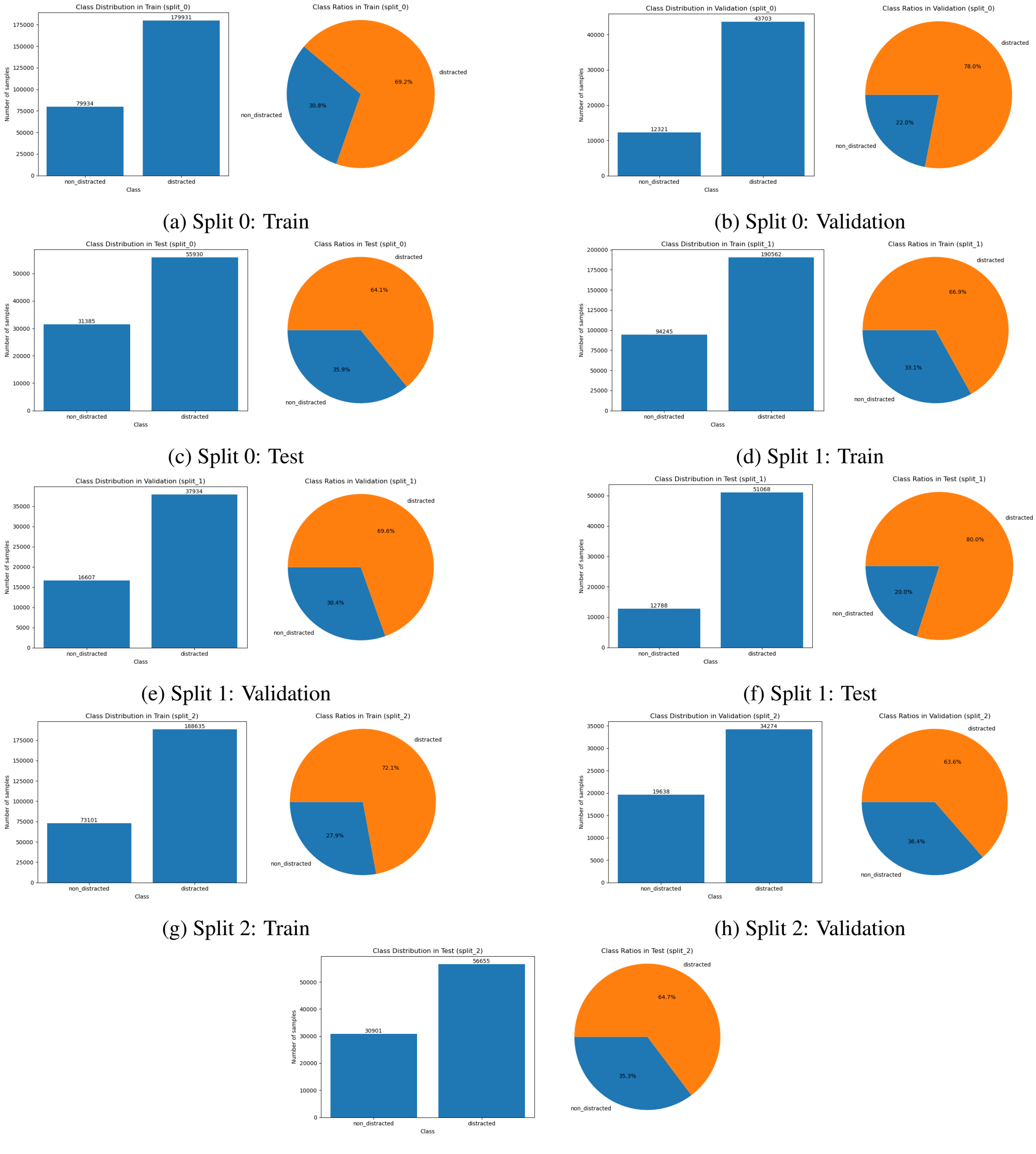
Dataset Split	**Modality**	**View**	**ImR Train**	**ImR Validation**	**ImR Test**
Split_0	Kinect RGB	Right Top	$\frac{179931}{79934} = 2.25$	$\frac{43703}{12321} = 3.54$	$\frac{55930}{31385} = 1.78$
Split_1	Kinect RGB	Right Top	$\frac{190562}{94245} = 2.02$	$\frac{37934}{16607} = 2.28$	$\frac{51068}{12788} = 3.99$
Split_2	Kinect RGB	Right Top	$\frac{188635}{73101} = 2.58$	$\frac{34274}{19638} = 1.74$	$\frac{56655}{30901} = 1.83$

Table 4.5: Imbalance in Kinect IR Right Top Image DAA Dataset

Dataset Split	**Modality**	**View**	**ImR Train**	**ImR Validation**	**ImR Test**
Split_0	Kinect Infra Red	Right Top	$\frac{209827}{87758} = 2.39$	$\frac{50465}{13721} = 3.67$	$\frac{64799}{34347} = 1.88$
Split_1	Kinect Infra Red	Right Top	$\frac{221575}{103027} = 2.15$	$\frac{44258}{18448} = 2.399$	$\frac{59258}{14351} = 4.129$
Split_2	Kinect Infra Red	Right Top	$\frac{218780}{80867} = 2.70$	$\frac{40313}{21490} = 1.87$	$\frac{65998}{33469} = 1.97$

4.1.3 Proposed Evaluation Metrics

As we have imbalance in the extracted image datasets, we can no longer rely on the traditional evaluation metrics like accuracy as already discussed in the chapter 2. Accuracy is the ratio of correctly predicted observations to the total observations. It can be misleading in evaluating performance of a model on our imbalanced datasets. So, following the recommendations of Johnson & Khoshgoftaar (2019a); Wang et al.



(a) Split 0: Train

(b) Split 0: Validation

(c) Split 0: Test

(d) Split 1: Train

(e) Split 1: Validation

(f) Split 1: Test

(g) Split 2: Train

(h) Split 2: Validation

(i) Split 2: Test

Figure 4.3: Class distribution of all three splits of Kinect Color Right Top Image DAA Dataset. In the bar plot, the x-axis corresponds to the class, while the y-axis reflects the number of images in each class. The pie chart displays the relative proportion, expressed as a percentage, of each class inside each split of the dataset. These proportions are generated based on the class ratios.

Table 4.6: Imbalance in Near Infra Red Front Top Image DAA Dataset

Dataset Split	**Modality**	**View**	**ImR Train**	**ImR Validation**	**ImR Test**
Split_0	Near Infra Red	Front Top	$\frac{351345}{156522} = 2.24$	$\frac{85400}{24101} = 3.54$	$\frac{109493}{61492} = 1.78$
Split_1	Near Infra Red	Front Top	$\frac{371987}{184436} = 2.01$	$\frac{74304}{32598} = 2.27$	$\frac{99947}{25081} = 3.98$
Split_2	Near Infra Red	Front Top	$\frac{369144}{143272} = 2.57$	$\frac{67123}{38541} = 1.74$	$\frac{109971}{60302} = 1.82$

(2016), this thesis employs the balanced accuracy score instead of the standard accuracy score for evaluating model performance. Such metrics offer a more nuanced insight into how well models perform across all classes, irrespective of their frequency. Below is the definition and mathematical formulation of the balanced accuracy score.

Balanced Accuracy: It is the average of the recall obtained on each class, ensuring equal treatment of each class's performance (Brodersen et al., 2010; Kelleher et al., 2015).

For a binary classification problem with classes C_1 and C_2, where C_1 is considered the positive or minority class and C_2 the negative or majority class, the balanced accuracy can be mathematically expressed as follows:

Let:

- TP denote the number of true positives.
- TN denote the number of true negatives.
- FP denote the number of false positives.
- FN denote the number of false negatives.

The recall for class C_1 (True Positive Rate) is:

$$\text{Recall}_{C_1} = \frac{TP}{TP + FN} \tag{4.5}$$

The recall for class C_2 (True Negative Rate) is:

$$\text{Recall}_{C_2} = \frac{TN}{TN + FP} \tag{4.6}$$

Therefore, the balanced accuracy is given by:

$$\text{Balanced Accuracy} = \frac{\text{Recall}_{C_1} + \text{Recall}_{C_2}}{2} \tag{4.7}$$

With respect to this thesis, class C_1 represents the '_non_distracted' driver class whereas class C_2 represents the 'distracted' driver class. This formula ensures that both classes are equally represented in the accuracy metric, which is particularly important in cases of class imbalance (Johnson & Khoshgoftaar, 2019a; Wang et al., 2016).

4.2 Novel Dataloader for Imbalanced Dataset

To address the first research question of this thesis, we propose ‘Clustered Feature Weighting’ a data loading strategy to improve imbalance in batches during model training. The methodology opted for the novel data loading is depicted in the flowchart 4.4. This flowchart shows the step by step procedure opted for the creation of the novel dataloader titled “ClusteredFeatureWeighting”. The novel dataloader designed for this thesis aims to address the inherent bias towards the majority class in imbalanced datasets by introducing balanced batches during model training. This enhancement not only aims to mitigate model bias but also aims to improve model performance by feeding balanced batches during model training.

4.2.1 Methodological Steps

The methodology is divided into four main parts starting from model selection to final dataloader comparison as shown below. Each part of the methodology contains different design choices and verification procedures. First we will start with the explanation of the steps involved in implementing the novel dataloader, which are as follows:

1. **Model Selection**: Choose a vision transformer model or encoder that has been pre-trained on the ImageNet-21K (Russakovsky et al., 2015) dataset in order to obtain embeddings from the Kinect Right Top Color Drive and Act image dataset.
2. **Variance Analysis**: Perform variance analysis on the collected features to assess the suitability of the selected model for feature extraction. Understanding the degree of separation between features within the same class and across different classes in the feature space is highly important. If the model is compatible with the imbalanced dataset under consideration, then move on to next step else select a suitable encoder again and verify its compatibility with the dataset under consideration by repeating the variance analysis.
3. **Clustering**: Arrange the features batchwise and Apply HDBSCAN (Campello et al., 2013) clustering algorithm on the cosine distance between the extracted features batchwise. This step is explained in detail in the section 4.2.2.
4. **Weight Generation**: Generate weights for a weighted random sampler based on the clustering results, to ensure diverse and representative samples in each batch.
5. **Comparison**: Compare the effectiveness of the novel dataloader with traditional dataloader to validate improvements in batchwise imbalance. If the results are satisfactory then move on to model training or else try different weights for outliers in order to see its impact on the sampling and imbalance in batches.

Before progressing with the methodological steps of the novel dataloader, it is essential to first establish the mathematical formulations that underpin these processes. These formulations not only guide the development and implementation of the dataloader but also serve as critical verification procedures for evaluating the efficacy of the selected model.

Mathematical Definitions and Formulations: To quantitatively analyze the features extracted by a model, several mathematical measures are employed. These measures are pivotal for evaluating how well a model can capture and differentiate features within and between different classes, which directly impacts the effectiveness of classification algorithms and in our case the novel dataloading startegy.

Intra-class Variance (σ^2_{intra}): Intra-class variance quantifies the variability or spread of feature vectors within a single class. It measures how much the features of individual samples deviate from the mean feature

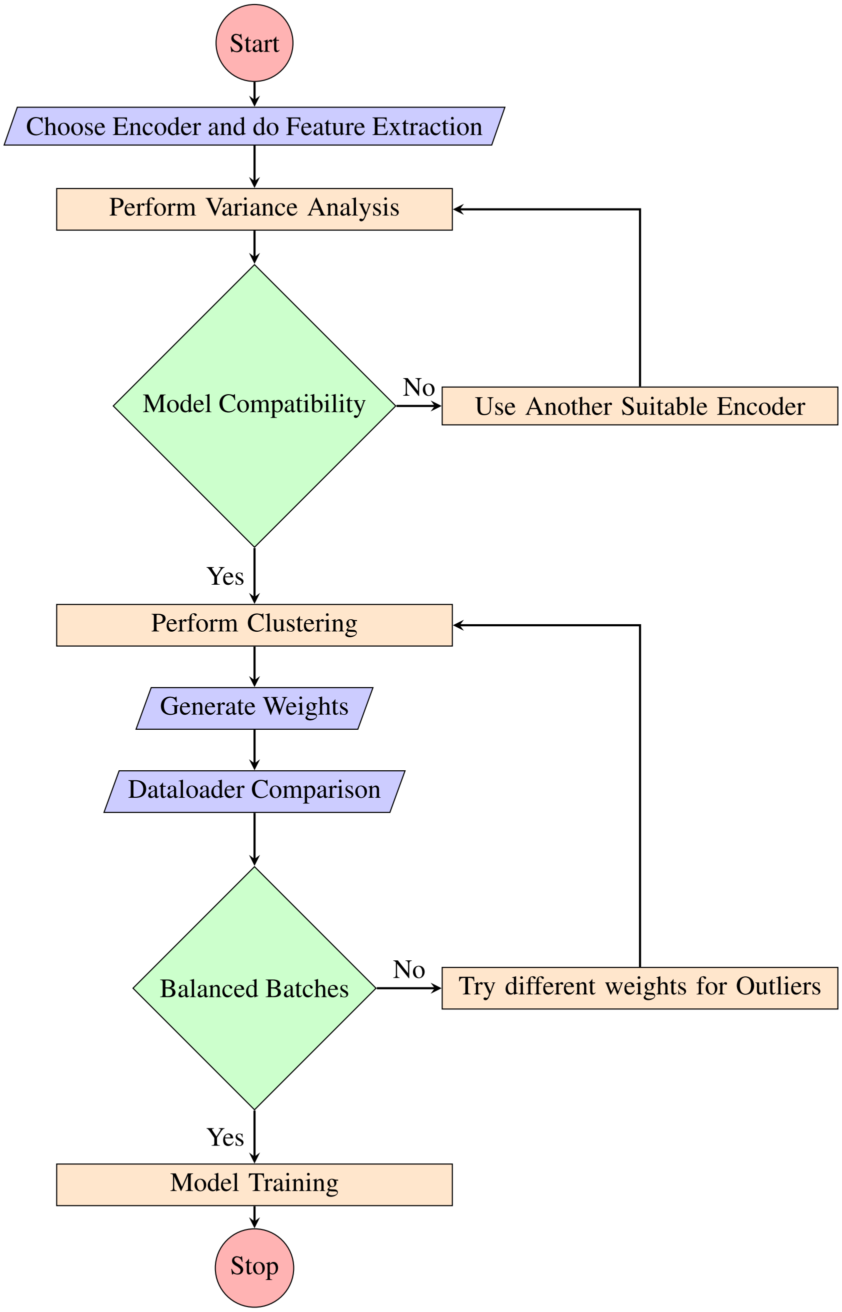

Start
Choose Encoder and do Feature Extraction
Perform Variance Analysis
Model Compatibility
No
Use Another Suitable Encoder
Yes
Perform Clustering
Generate Weights
Dataloader Comparison
Balanced Batches
No
Try different weights for Outliers
Yes
Model Training
Stop

Figure 4.4: Methodology for Novel Dataloader for Imbalanced Dataset.

vector of their respective class. A lower intra-class variance indicates a high degree of similarity among the samples within the class, suggesting that the model is consistent in capturing features for a specific class (Pilarczyk & Skarbek, 2019).

Mathematically,

$$\sigma^2_{intra} = \frac{1}{N}\sum_{i=1}^{N}(\boldsymbol{x}_i - \mu)^2 \tag{4.8}$$

where: $\boldsymbol{x}_i$ represents the feature vector of the i-th image in a class, μ denotes the mean feature vector of that class, and N is the total number of samples in the class (Pilarczyk & Skarbek, 2019).

Calculation of Class Centers: To calculate the class centers in a feature space extracted using a pretrained Vision Transformer model, the mean feature vector (center) for each class is computed by averaging the feature vectors of all samples within each class (Pilarczyk & Skarbek, 2019). As there are two classes (A) 'non_distracted' and (B) distracted in the dataset under consideration, we can calculate the class centers for each class in the feature space as follows: If N is the total number of sample in each class, then,

For class A:

$$\text{center}_A = \text{Mean Feature Vector}_A = \mu_0 = \frac{1}{N}\sum_{i=1}^{N}\boldsymbol{x}_i^A \tag{4.9}$$

where $\boldsymbol{x}_i^A$ represents the feature vector of the i-th sample in class A.

For class B:

$$\text{center}_B = \text{Mean Feature Vector}_B = \mu_1 = \frac{1}{N}\sum_{i=1}^{N}\boldsymbol{x}_i^B \tag{4.10}$$

where $\boldsymbol{x}_i^B$ represents the feature vector of the i-th sample in class B.

Inter-class Variance (σ^2_{inter}): Inter-class variance measures the differences between the mean feature vectors of different classes. This metric is crucial for determining how distinct the classes are from each other in the feature space. Higher values of inter-class variance suggest that the classes are more distinguishable, indicating that the model is effective in capturing diverse features necessary for differentiating between classes (Yu et al., 2022). Mathematically,

$$\sigma^2_{inter} = (\mu_0 - \mu_1)^2 \tag{4.11}$$

where: μ_0 and μ_1 are the mean feature vectors of two different classes.

Distance Between Class Centers: The distance between class centers is an additional metric used to evaluate the separability of classes within the feature space. This measure complements inter-class variance by providing a direct assessment of the spatial separation between class centers, which can be interpreted as an indicator of the model's ability to partition the feature space effectively. Once the centers of each class are computed, the Euclidean distance (Wikipedia contributors, 2024b) between these two centers can be calculated using the following formula:

$$d(\text{center}_A, \text{center}_B) = \sqrt{\sum_{i=1}^{M}(\text{center}_A[i] - \text{center}_B[i])^2} \tag{4.12}$$

where, M is the size of the feature vector obtained after feature extraction or in other words M is the embedding size of the encoder used for feature extraction. For our case, M is 1280.

These metrics together create a strong foundation for evaluating the model's ability to accurately differentiate across classes using the information it extracts. They are essential for validating the feature extraction capabilities of the model, ensuring that it captures both the nuances within classes and the distinctions between different classes.

Cosine Similarity: Cosine similarity measures the cosine of the angle between two non-zero vectors in an inner product space. This measure is a reflection of the cosine of the angle between the two vectors and is calculated using the dot product of the vectors and the magnitudes (norms) of each vector (Wikipedia contributors, 2024a).

Given two vectors, $\boldsymbol{x}$ and $\boldsymbol{y}$, their cosine similarity, similarity($\boldsymbol{x}$, $\boldsymbol{y}$), is defined as:

$$\text{similarity}(\boldsymbol{x}, \boldsymbol{y}) = \cos(\theta) = \frac{\boldsymbol{x} \cdot \boldsymbol{y}}{\|\boldsymbol{x}\|\|\boldsymbol{y}\|} \tag{4.13}$$

where:

- $\boldsymbol{x} \cdot \boldsymbol{y}$ is the dot product of vectors $\boldsymbol{x}$ and $\boldsymbol{y}$.
  If $\boldsymbol{x} = [x_1, x_2, \ldots, x_n]$ and $\boldsymbol{y} = [y_1, y_2, \ldots, y_n]$, then:

$$\boldsymbol{x} \cdot \boldsymbol{y} = x_1 y_1 + x_2 y_2 + \ldots + x_n y_n \tag{4.14}$$

- $\|\boldsymbol{x}\|$ is the Euclidean norm (or magnitude) of the vector $\boldsymbol{x}$. The Euclidean norm of a vector $\boldsymbol{x}$ is defined as:

$$\|\boldsymbol{x}\| = \sqrt{x_1^2 + x_2^2 + \ldots + x_n^2} \tag{4.15}$$

Cosine Distance: Cosine distance is a measure derived from cosine similarity and is used to quantify the dissimilarity between two vectors. Thus, it ranges from 0 to 2, where 0 indicates identical vectors and 2 indicates completely opposite vectors (Wikipedia contributors, 2024a). It is defined as:

$$\text{distance}(\boldsymbol{x}, \boldsymbol{y}) = 1 - \text{similarity}(\boldsymbol{x}, \boldsymbol{y}) \tag{4.16}$$

This formula essentially inverts the cosine similarity to provide a distance metric: when the cosine similarity is 1 (meaning $\boldsymbol{x}$ and $\boldsymbol{y}$ are identical), the cosine distance is 0; conversely, when the cosine similarity is 0 (meaning $\boldsymbol{y}$ and $\boldsymbol{y}$ are orthogonal), the cosine distance is 1.

Practical Use in a Matrix: When utilizing the pretrained vision transformer encoder model, a feature matrix $\boldsymbol{X}$ is produced for one batch. This matrix has a dimension of [1024 x 1280], where each row represents a sample vector. The batch size is 1024 and the embedding size is 1280. In this scenario, the cosine similarity can be calculated for each pair of vectors. This leads to the creation of a similarity matrix in which each member (i, j) represents the cosine similarity between the i-th and j-th vectors in $\boldsymbol{X}$. The cosine distance matrix is obtained by subtracting the similarity matrix from one. This approach for computing distances and similarities is particularly valuable in high-dimensional spaces, where the traditional Euclidean distance may lose its significance due to the curse of dimensionality. This thesis employs this approach to compute the cosine distance matrix for the HDBSCAN clustering algorithm.

Next, we will delve into the implementation details of the novel dataloader. The following section presents the pseudocode for the 'Clustered Feature Weighting' data loading strategy and thoroughly explains the corresponding algorithmic workflow. This includes the strategy's implications and advantages. Following this, the section outlines the clustering process, detailing its benefits and drawbacks, and discusses its possible impact on data variability in batches and overall model training.

4.2.2 ClusteredFeatureWeighting Data Loading Strategy:

This thesis presents a new approach to address the difficulties encountered due to imbalanced image datasets. The proposed solution is a unique data loading technique, outlined in algorithm 1 named “ClusteredFeatureWeighting”.This approach leverages advanced machine learning techniques, including clustering and weighted sampling, to enhance the training of vision models on skewed image datasets. The following section details the technical components and operational flow of this strategy, emphasizing its integration into the training process.

4.2.3 Algorithmic Workflow

The data loading strategy is structured around several critical phases, each tailored to optimize the model’s exposure to under-represented classes in an imbalanced dataset. The algorithm 1 operates as follows:

1. **Model Initialization**:
   - Initialize a pre-trained vision transformer model using the ImageNet-21K dataset (Ridnik et al., 2021). This model serves as the foundation for feature extraction, leveraging its broad, generic feature representation capabilities gained through rigorous pre-training.
2. **Feature Extraction**:
   - For each batch of images in the training dataset, features are extracted using the pre-trained ViT specified for feature extraction. This step converts raw image data into a high-dimensional feature space where semantic similarities and differences are more visible.
   - Extracted features, alongside true class labels and image paths, are recorded for subsequent processing.
3. **Clustering and Weight Assignment**:
   - Within each batch, a cosine distance matrix is computed to measure the dissimilarities between all feature embeddings.
   - HDBSCAN (Campello et al., 2013), a robust clustering algorithm suitable for handling varying cluster densities and sizes, is applied to this distance matrix. This method effectively identifies natural clusters and outliers within the data.
   - Weights are assigned to features based on cluster membership. Each feature within a cluster receives a weight inversely proportional to the cluster size, promoting equal representation across clusters. Outliers are assigned a minimal weight of 0.01, maintaining their presence in the dataset without dominating the training process. This outlier weight requires fine tuning and can be tuned by choosing a value between 0 to 1 where 0 being the lowest weight corresponding to 0 probability in the weighted random sampler and 1 being the highest probability in the weighted random sampler for sampling.
4. **Weights Aggregation**:
   - Weights from all batches are aggregated and paired with their corresponding image paths. This consolidated list forms the basis for the customized sampling strategy in subsequent training iterations.
5. **Custom Dataset and DataLoader Configuration**:
   - A custom dataset class, `WeightedImageDataset`, is defined to handle the storage and retrieval of images, weights, labels, and paths.
   - The `getitem` method is tailored to return weighted samples, ensuring that each data retrieval aligns with the predetermined weights.

Algorithm 1 Pseudocode: ClusteredFeatureWeighting Data Loading Strategy

```
def extract_features(pretrained_encoder, dataset):
    features, labels, paths = [ ], [ ], [ ]
    for image, label, path in dataset:
        feature = pretrained_encoder(image)
        features.append(feature)
        labels.append(label)
        paths.append(path)
    return features, labels, paths
def assign_weights(features):
    weights = [ ]
    for feature_batch in features:
        cosine_matrix = 1 - cosine_similarity(feature_batch)
        clusters, outliers = HDBSCAN(cosine_matrix)
        for cluster in clusters:
            weight = 1.0 / len(cluster)
            for index in cluster:
                weights[index] = weight
        for outlier in outliers:
            weights[outlier] = 0.01
    return weights
class WeightedImageDataset:
    def __init__(self, imagepath_list, weights_list, labels_list):
        self.imagepaths = [path for imagepath in imagepath_list]
        self.weights = [weight for weight in weights_list]
        self.labels = [label for label in labels_list]
    def load_image(self, image_path):
        return Image.open(image_path)
    def __getitem__(self, idx):
        image = self.load_image(image_path)
        return image, self.imagepath, self.weights[idx], self.labels[idx]
def create_dataloader(image_paths, weights, labels, batch_size):
    dataset = WeightedImageDataset(image_paths, weights, labels)
    sampler = WeightedRandomSampler(weights, len(weights), rep=True)
    return DataLoader(dataset, batch_size, sampler)
def train_model(dataloader, epochs):
    for epoch in range(epochs):
        for images, weights, labels in dataloader:
            Perform training step with images and labels
model = VisionTransformerModel('ImageNet-21K')
dataset = LoadDataset()
features, labels, paths = extract_features(model.encoder, dataset)
weights = assign_weights(features)
dataloader = create_dataloader(paths, weights, labels, batch_size=1024)
train_model(dataloader, 100)
```

    - A `WeightedRandomSampler` is employed with the calculated weights, set with replacement to true, allowing for repeated selection of underrepresented samples, thereby enhancing their influence on the model training.

6. **Training Loop**:

    - The training process iterates over the dataset using a PyTorch `DataLoader` configured with the custom dataset and weighted random sampler. This setup ensures that each batch is reflective of the weighted sampling strategy, focusing model training on a balanced representation of the dataset.

4.2.4 Clustering Procedure

We utilized HDBSCAN (Campello et al., 2013), a density based clustering algorithm known for its effectiveness in identifying clusters without predefining the number of clusters. This method is especially well-suited for data with a high number of dimensions, such as ours. Our data consists of a feature matrix with dimensions of [1024 x 1280] per batch, with a batch size of 1024. In this matrix, each row represents a sample and each column represents a feature. Clustering was based on the cosine distance matrix derived batch wise from the dataset. This metric emphasizes the angle between feature vectors, grouping together samples that are directionally similar. Post-clustering, each sample was assigned a weight. Samples within dense clusters received higher weights to emphasize their representativeness of common data patterns, while outliers were given lower weights to decrease their selection frequency in future sampling, yet keeping them within the analytical scope.

Advantages of the Clustering Approach Using HDBSCAN offers several advantages:

- **Adaptability and Effectiveness**: The algorithm excels in managing diverse data shapes and densities, crucial for our complex, high-dimensional features.
- **Focus on Angular Similarity**: By using cosine distances, the approach is sensitive to directional similarities, which is particularly useful in datasets where traditional measures like Euclidean distance may be less informative.
- **Enhanced Data Sampling**: Integrating weighted random sampling ensures that the data batches used for model training are not just random collections but are reflective of the underlying data structure, potentially improving the learning process.

Limitations and Challenges

Despite its strengths, the clustering method faces some challenges:

- **Dimensional Sensitivity**: Relying solely on angular differences might overlook other important aspects of data similarity or dissimilarity.
- **Batch Consistency**: HDBSCAN's flexibility can sometimes lead to different clustering outcomes for different batches, which might affect the consistency of sample weights and training outcomes.
- **Complexity in Outlier Management**: Handling outliers by assigning them low weights reduces their impact but might neglect valuable anomalous patterns that could be essential for certain predictions.

Impact on Data Variability and Model Training The strategy of weighted random sampling, particularly with replacement, ensures a comprehensive representation of data patterns:

- **Balanced Variability**: This approach maintains essential data variability, crucial for preventing model overfitting. It shows the model images with central and unusual data features.
- **Inclusive Sampling**: Each batch includes a mix of both highly representative samples and outliers, providing a well-rounded dataset for training and enhancing the model's ability to generalize.

4.3 Methodology for DataLoader Comparison

The versions of the DAA image datasets, exhibit imbalance across their three splits. To address this, "Clustered Feature Weighting" strategy is proposed in this thesis. However, this approach need validation for its effectiveness in producing balanced batches and increasing model performance and generalisation. This section provides details about the comparison of two data loading strategies: a traditional dataloader and the novel dataloader using a"Clustered Feature Weighting" strategy, designed to produce more balanced training batches.

Data Loading Strategies:

- **Traditional DataLoader:** This loader samples data directly from the imbalanced dataset, reflecting inherent class imbalances. It is used as a baseline for comparison.
- **Clustered Feature Weighting DataLoader:** This approach adjusts sampling probabilities to achieve a balanced class representation within each batch, thus enhancing the model's exposure to under-represented classes during training.

Methodology for Comparison The effectiveness of each dataloader on balanced distribution inside batches is evaluated based on their ability to distribute class samples evenly across batches. Ideally, for a batch size of 1024, each class should be represented by 512 samples, given that we have a binary dataset with two classes 'non distracted' and 'distracted' driver. The comparison focuses on how well each dataloader approximates this ideal distribution. The Kullback-Leibler (KL) divergence (Kullback, 1997) is employed to quantitatively measure the discrepancy between the ideal and the actual distributions provided by the dataloaders. Kullback-Leibler divergence is a statistical metric that estimates the extent to which one probability distribution deviates from a second, reference probability distribution (Kullback, 1997).

Mathematical Formulation: The KL divergence (Kullback, 1997) from a true probability distribution P to an approximate distribution Q over discrete variables is defined as follows:

$$D_{KL}(P \parallel Q) = \sum_i P(i) \log \left(\frac{P(i)}{Q(i)} \right) \tag{4.17}$$

where, P, the uniform distribution per batch, is:

$$P = \left[\frac{1}{2}, \frac{1}{2} \right] \tag{4.18}$$

and Q, the observed distribution from a dataloader for each batch, is calculated as:

$$Q = \left[\frac{\text{number of `non\_distracted' samples}}{1024}, \frac{\text{number of `distracted' samples}}{1024} \right] \tag{4.19}$$

Criteria for Comparison: A lower KL divergence indicates a closer approximation to the ideal distribution, signifying a more effective dataloader in terms of managing class balance (Kullback, 1997). The KL divergence values against batch numbers for both dataloaders has been plotted. This visual analysis helps identify which dataloader consistently provides a more balanced class distribution across batches.

4.4 Methodology for Model Training and Evaluation

This section details the methodologies used across experiments to enhance driver distraction detection through supervised and self-supervised learning-based pre-trained encoders. We explore supervised and self-supervised learning-based pre-trained encoders, grayscale augmentation, and novel data loading to assess their impact on model accuracy and generalizability. Each experiment tests the models under different conditions, focusing on their adaptability across visual modalities and camera perspectives, which is critical for real-world automotive applications. In the following chapter, we will conduct thorough experiments to gain a deeper understanding of how various methods of training encoders accurately identify drivers who are distracted.

4.4.1 Experiment 1: Supervised Learning Based Encoder

We employed the Kinect Color DAA Image dataset, captured from a right-top camera view, for training and evaluation. Figure 4.5 illustrates the workflow for this experiment. We utilized a vision transformer encoder (vit_b_16) (PyTorch Contributors, 2024), pre-trained on the Imagenet-1K (Russakovsky et al., 2015) dataset using a supervised learning approach accessed from the torchvision library. This model, serving as a frozen backbone, extracts features for the downstream task of driver distraction detection. Our objective is to assess the effectiveness of a pre-trained encoder, using supervised learning, in identifying distracted drivers. We added a linear layer, distinguishing between distracted and non-distracted drivers, atop the encoder. This layer underwent fine-tuning on the dataset for 100 epochs, guided by hyperparameters derived from our search discussed in the subsequent chapter. Pre-trained model specific data transformations were applied during the training and evaluation phases. We employed the cross-entropy loss (Mao et al., 2023) as our loss function, utilizing a batch size of 1024. We evaluated the performance of the trained model by testing it on previously unseen datasets from the Kinect Color Right Top view Image DAA test datasets.

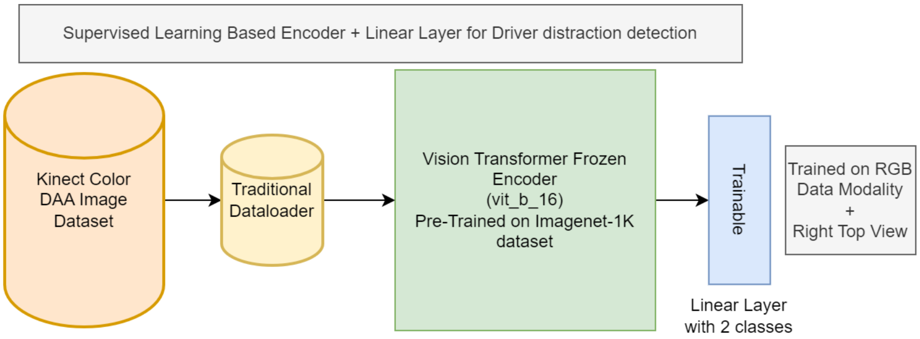

Supervised Learning Based Encoder + Linear Layer for Driver distraction detection
Kinect Color DAA Image Dataset
Traditional Dataloader
Vision Transformer Frozen Encoder (vit_b_16) Pre-Trained on Imagenet-1K dataset
Trainable
Linear Layer with 2 classes
Trained on RGB Data Modality + Right Top View

Figure 4.5: Methodology for supervised learning based pre-trained encoder for downstream task of driver distraction detection.

4.4.2 Experiment 2: Supervised Learning Based Encoder with Gray scale Augmentation

Figure 4.6 outlines the methodology for this experiment, which mirrors Experiment 1 with a significant variation: the addition of grayscale augmentation. This experiment was designed to explore the impact of grayscale augmentation on model generalization, particularly under low-light or night time driving conditions where color images may offer limited information. We hypothesized that grayscale images might yield better generalizability in such scenarios. Thus, we trained our hybrid classifier with grayscale augmentation for 100 epochs and subsequently evaluated its effectiveness.

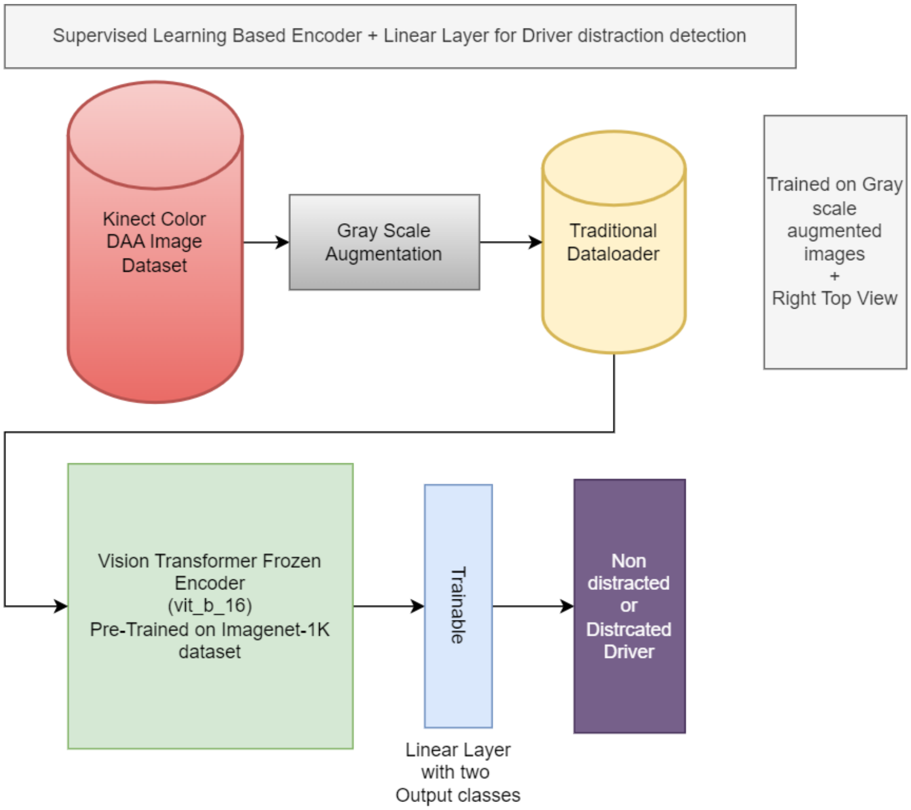

Supervised Learning Based Encoder + Linear Layer for Driver distraction detection
Kinect Color DAA Image Dataset
Gray Scale Augmentation
Traditional Dataloader
Trained on Gray scale augmented images + Right Top View
Vision Transformer Frozen Encoder (vit_b_16) Pre-Trained on Imagenet-1K dataset
Trainable
Linear Layer with two Output classes
Non distracted or Distrcated Driver

Figure 4.6: Methodology for supervised learning using a pre-trained encoder with grayscale augmentation for driver distraction detection. The figure shows grayscale transformations applied during data loading, incorporating IR modality knowledge to fine-tune a linear layer on a frozen encoder.

4.4.3 Experiment 3: Self-Supervised Learning (SSL) Based Encoder

The methodology for this experiment is depicted in figure 4.7. We utilized a vision transformer encoder, vit_b_14 (Facebook AI Research, 2023), trained with the DINOv2 SSL method on the extensive, unlabeled LVD-142M (Oquab et al., 2023) dataset. The encoder, frozen to preserve its feature-extracting capabilities, processes image batches to serve as inputs to a linear classifier designed for detecting driver distractions. This setup allowed us to train the linear layer of the model on the color modality with a right top view and evaluate its performance on unseen test datasets. This experiment aims to compare the performance of linear layer fine tuned on top of self-supervised learning-based frozen encoder against Experiments 1 and 2. For a balanced comparison, we will evaluate the 100th checkpoint of the finely-tuned linear layers across all experiments, assessing both validation and test balanced accuracy scores.

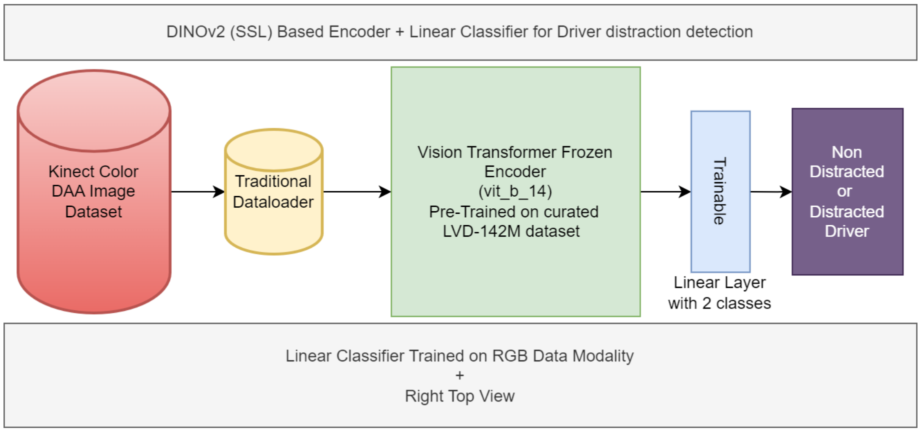

DINOv2 (SSL) Based Encoder + Linear Classifier for Driver distraction detection
Kinect Color DAA Image Dataset
Traditional Dataloader
Vision Transformer Frozen Encoder (vit_b_14) Pre-Trained on curated LVD-142M dataset
Trainable
Linear Layer with 2 classes
Non Distracted or Distracted Driver
Linear Classifier Trained on RGB Data Modality + Right Top View

Figure 4.7: Methodology for self-supervised learning based pre-trained encoder (DINOv2 vit_b_14) for downstream task of driver distraction detection. The figure depicts the traditional dataloading of the Kinect Color DAA dataset and fine-tuning of a linear layer on top of frozen encoder for downstream driver distraction detection task.

Choice of Linear Evaluation protocol

The authors of the (Oquab et al., 2023) have used both kNN and linear evaluation protocols in their research. But in this thesis, the DINOv2 based 'vit_b_14' encoder (Oquab et al., 2023), pretrained on curated LVD-142M unlabeled dataset, is tested using the linear probing on the Kinect color right top drive and act image dataset. This decision is guided by several factors:

- **Computational Efficiency:** Linear evaluation requires considerably less computational resources compared to full fine-tuning and can be executed relatively quickly (Balestriero et al., 2023).
- **Direct Assessment of Representational Quality:** Since linear probing focuses purely on the discriminative power of the pre-trained features without allowing significant model adaptation, it provides a clear indication of the quality of the SSL-induced features (Zhang et al., 2016; Balestriero et al., 2023).
- **Practical Relevance:** Training a linear classifier on a fixed backbone roughly replicates real-world scenarios, where SSL models are often used as feature extractors in larger systems (Zhang et al., 2017; Balestriero et al., 2023).

4.4.4 Experiment 4: SSL Based Encoder with Clustered Feature Weighting

Figure 4.8 details this experiment's methodology, highlighting the use of a clustered feature weighting data loading technique, a deviation from traditional data loading approaches. This method is expected to enhance model training and generalization across unseen datasets and different modalities or views. The same encoder and linear layer configuration from Experiment 3 is employed, but with the clustered feature weighting strategy during model training.

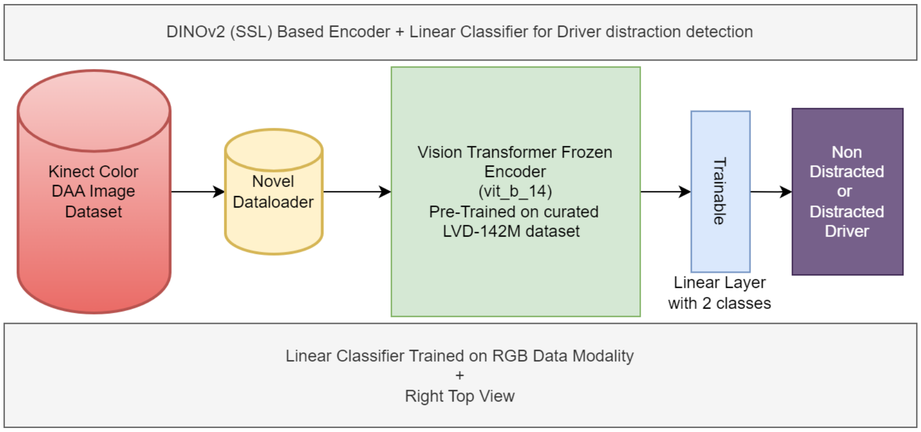

DINOv2 (SSL) Based Encoder + Linear Classifier for Driver distraction detection
Kinect Color DAA Image Dataset
Novel Dataloader
Vision Transformer Frozen Encoder (vit_b_14) Pre-Trained on curated LVD-142M dataset
Trainable
Linear Layer with 2 classes
Non Distracted or Distracted Driver
Linear Classifier Trained on RGB Data Modality + Right Top View

Figure 4.8: Methodology for self-supervised learning based pre-trained encoder (DINOv2 vit_b_14) with Clustered Feature Weighting Data-loading for downstream task of driver distraction detection. In the figure novel dataloader corresponds to Clustered Feature Weighting based dataloader.

Data Transformations in Experiment 3 and 4: Data transformation strategies are crucial for training robust models. We employed the same data transformation strategy for both training and evaluation phases as described and used in (Oquab et al., 2023) for linear probing.

Training Transforms: The training transforms includes a series of transformations to simulate diverse viewing conditions:

- **RandomResizedCrop**: Applied with bicubic interpolation to preserve image quality while adjusting sizes. The crop size used is 224.
- **RandomHorizontalFlip**: Conditionally applied based on a 0.5 probability to introduce horizontal asymmetry.
- **MaybeToTensor**: Ensured all inputs were converted to tensors to accommodate different formats.
- **Normalization**: Used predefined ImageNet (Russakovsky et al., 2015) mean and standard deviation values to standardize inputs.

Evaluation Transforms: The evaluation transforms aimed for consistency and reproducibility:

- **Resize and CenterCrop**: First the image is resized to a size 256 using bicubic interpolation to ensure high-quality image representation at standard dimensions. Then the CenterCrop with crop size 224 is applied.
- **MaybeToTensor and Normalization**: Consistently prepared and standardized inputs, similar to training transforms.

4.4.5 Methodology for Cross-Modality Generalization Evaluation

Figure 4.9 illustrates the cross-modality generalization approach. We evaluated the 100th epoch checkpoints of the trained encoders on the KIR Right Top and NIR Front Top IR datasets, which present different imaging modalities (IR) compared to the color data used in model training. This experiment seeks to understand the adaptability of models to different sensory information, relevant in diverse lighting conditions.

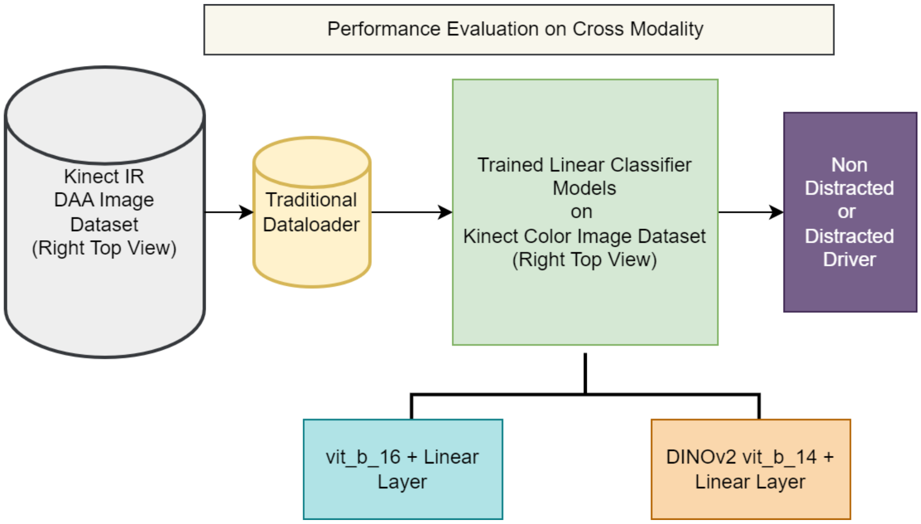

Performance Evaluation on Cross Modality
Kinect IR DAA Image Dataset (Right Top View)
Traditional Dataloader
Trained Linear Classifier Models on Kinect Color Image Dataset (Right Top View)
Non Distracted or Distracted Driver
vit_b_16 + Linear Layer
DINOv2 vit_b_14 + Linear Layer

Figure 4.9: Methodology for cross-modality generalisation for downstream task of driver distraction detection. KIR Right Top view dataset only differs in the modality of data when compared with Kinect Color Right Top view dataset. Hence, using Kinect IR DAA test sets, the cross-modality evaluation across all four experimental scenarios is performed.

4.4.6 Methodology for Cross-View Generalization Evaluation

Figure 4.10 shows the methodology for cross-view generalization. Here, we assess the models trained on the right view Kinect color image DAA dataset against the NIR Front Top dataset, which provides a different camera perspective. This experiment aims to verify the models' robustness to varying viewpoints, simulating real-world scenarios where camera angles can differ unexpectedly.

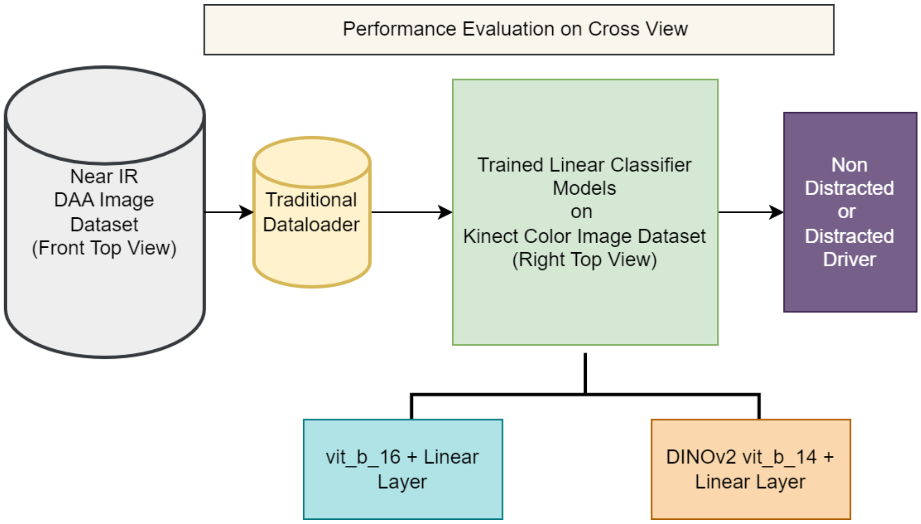

Performance Evaluation on Cross View
Near IR DAA Image Dataset (Front Top View)
Traditional Dataloader
Trained Linear Classifier Models on Kinect Color Image Dataset (Right Top View)
Non Distracted or Distracted Driver
vit_b_16 + Linear Layer
DINOv2 vit_b_14 + Linear Layer

Figure 4.10: Methodology for cross-view generalisation for downstream task of driver distraction detection. In the figure, the NIR DAA dataset is used with front top view and Ir modality for cross-view and cross modality generalisation. This evaluation done in all four experiments using both types of encoder under consideration as depicted in the figure.

Chapter 5

Experiments and Results

In Chapter 4, we outlined the methodologies employed across various experiments integral to this thesis. Building on that foundation, Chapter 5 presents the empirical evidence validating the methods and concepts previously discussed. We begin by detailing the experimental setup, including a description of the datasets and models utilized, as well as the computational resources involved. This is followed by an analysis of experiments focused on our novel data loading technique, presenting specific results from these experiments.

Subsequent sections delve into the outcomes of Experiments 1 through 4, examining their effectiveness in cross-modality and cross-view generalization. Each experiment is contextualized with comprehensive results, facilitating a thorough understanding of their impact and significance.

This chapter aims to solidify the empirical groundwork for the conclusions drawn in the subsequent final chapter, where these results are synthesized into a cohesive conclusion regarding the thesis objectives.

5.1 Experimental Setup

5.1.1 Datasets

For our experiments we have utilized three distinct datasets derived from the DAA video dataset as listed below:

1. **Kinect Color Right Top Image DAA Dataset**: Consists of RGB images used to train and evaluate the model's ability to detect distracted and non-distracted drivers. The train dataset from split 0 of this dataset is utilized for dataloader experiments. The train and validation datasets from split 0 of this dataset are used for hyperparameter search.

2. **Kinect IR Right Top Image DAA Dataset**: Comprises grayscale images where each channel replicates the same grayscale information, used primarily to test the models' generalization across different imaging modalities.

3. **NIR Front Top Image DAA Dataset**: Contains grayscale images similar to the Kinect IR dataset but from a front top viewpoint. It is used to assess the models' capacity to generalize across front top view and IR modality.

5.1.2 Model Architectures

Three primary models are utilized in this thesis:

- **Feature Extraction Model**: The 'vit_huge_patch14_224.orig_in21k' (Face, 2024) model is used for feature extraction in the 5.2, where the novel 'Clustered Feature Weighting' data loading startegy is evaluated. The model is accessed from Huggingface librray. Equipped with 658.7 million parameters and pretrained on the ImageNet-21K dataset, this model efficiently extracts features from image datasets. It expacts an input image of size 224 x 224 pixels and extract a [1 x 1280] feature vector for a batch of single image.
- **Supervised Learning based pre-trained Vision Transformer**: The 'vit_b_16' (PyTorch Contributors, 2024) model accessed from the Torchvision library, pretrained on imagenet-1K dataset is used as a supervised encoder. It served as a pretrained backbone, on top of which a zero initialized linear layer with two output classes is used to classify 'non distracted' and 'distracted' driver classes in experiment 1 and 2. Total parameters in the resulted model are 85,800,194 out of which total trainable parameters are 1538.
- **Self-Supervised learning based pre-trained Vision Transformer (vit_b_14)**: The DINOv2 'vit_b_14' (Facebook AI Research, 2023) model is accessed from Pytorch hub, it is pretrained on the unlabeled curated LVD-142M dataset via the DINOv2 (Oquab et al., 2023) self-supervised learning approach. It served as a backbone in experiments 3 and 4. A linear classifier is defined with frozen backbone to utilize the pre-trained weights of the 'vit_b_14' encoder on the driver distraction detection task. The linear layer is initialised with weights drawn from a normal distribution with (mean = 0 and standard deviation = 0.01) and bias = 0. Total parameters in the resulted model are 86,582,018 out of which total trainable parameters are 1538.

5.1.3 Computational Resources

The experiments from 1 to 3 utilized dual NVIDIA Tesla V-100-SXM2-32 GB GPUs setup. Most training was facilitated via the Distributed Data Parallel (DDP) (Li et al., 2020b; PyTorch Contributors, 2023a) algorithm to enhance computational efficiency across two GPUs. The effective batch size used is 1024 which means a batch size of 512 per gpu in DDP setup. The experiment 4, using 'ClusteredFeatureWeighting' data loading strategy, utilized a single GPU with the same effective batch size for fair comparison.

5.2 Results of Dataloader Experiments

The novel data loader proposed in this thesis promotes a fairer evaluation of model performance across different classes by ensuring a more equitable class representation in each training batch. It also aims at improving the trained models' overall robustness and reliability.

5.2.1 Assessment of the Vision Transformer Model as a Feature Extractor

This section details the performance of the Vision Transformer model 'vit_huge_patch14_224.orig_in21k' (Face, 2024) in extracting features. Equipped with 658.7 million parameters and pretrained on the ImageNet-21K (Ridnik et al., 2021) dataset, this model efficiently extracts features from the Kinect Color Right Top Image dataset. We resize each image to 224 x 224 pixels and extract a [1 x 1280] feature vector, culminating in a [1024 x 1280] embedding for batches of 1024 images.

The primary aim of this experiment was to evaluate the variance of features within and across classes after extraction. We organized features by class from the Kinect Color DAA's training dataset (split 0). The feature dimensions for the ‘non-distracted’ class were [79934 x 1280], and for the ‘distracted’ class were [179931x1280]. We calculated intra-class variance vectors for each class (Equations 4.8), determined class centers (Equations 4.9 and 4.10), assessed inter-class variance (Equation 4.11), and measured the distance between class centers (Equation 4.12), which was found to be 0.366.

Figure 5.1 presents the variance spread across the 1280 dimensions, normalized to the maximum variance value observed. This normalization facilitates direct comparison across variances, aiding in identifying the most variable and potentially discriminative features. The analysis is crucial for understanding how effectively the feature extractor can differentiate between ‘distracted’ and ‘non-distracted’ categories.

Intra-Class Variance: The visualization indicates that Class 0 (blue) generally displays lower variance, suggesting more homogeneity within this class compared to Class 1 (red), which shows higher variance due to its larger sample size and greater diversity.

Inter-Class Variance: Represented by a green line, the inter-class variance indicates that the average features of the two classes are similar across many dimensions, highlighting the need for a more effective feature extraction approach to enhance class distinction.

Notably, the higher intra-class variances (peaks in red and blue lines) pinpoint dimensions where data points are more dispersed, providing insights into class characteristics. Features showing higher inter-class variance are critical for distinguishing between classes. Our analysis indicates moderate separability, suggesting that more sophisticated models or feature transformations may be necessary to improve class distinction. However, the existing feature distinction suffices for evaluating the novel dataloader approach.

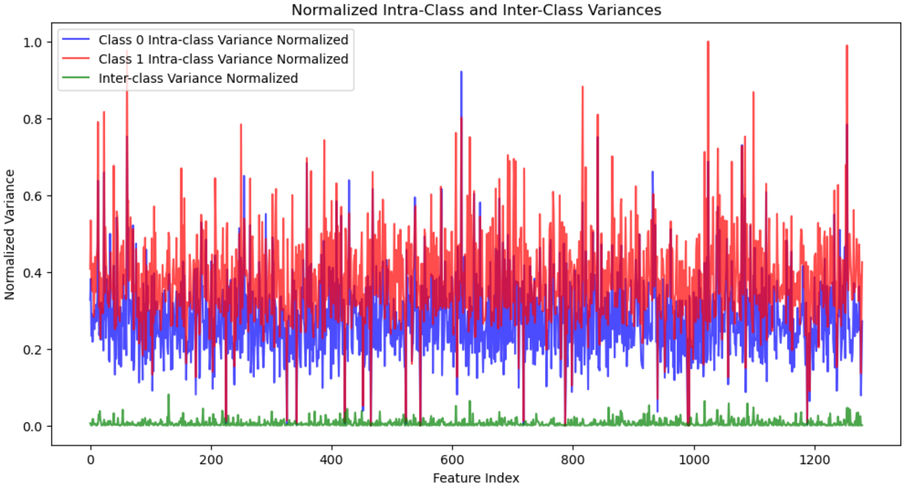


Figure 5.1: Feature variance analysis using the vision transformer model. Displays intra-class variance curves for ‘non-distracted’ (blue) and ‘distracted’ (red) drivers, with the green line depicting inter-class variance. The x-axis denotes the 1280 features extracted, and the y-axis shows normalized variance values, using the highest observed variance of 0.00423 for normalization.

Challenges in Class Separability: The analysis indicates a low inter-class variance of approximately 0.01 and a distance of 0.366 between class centers, suggesting the model’s limitations in distinct class differentiation. Nonetheless, these limitations are less critical as the model’s primary function is to enrich the input for clustering and weighting rather than direct classification. The HDBSCAN algorithm leverages the broad spectrum of extracted features to generate meaningful clusters, which are more valuable than outright class separability for this application.

This strategy enhances training batch diversity, crucial for improving generalization in the subsequent model training. A weighting strategy that assigns lower weights to outliers (0.001) ensures that these atypical features do not disproportionately influence the training process, mitigating potential biases from high intra-class variance.

Overall, while the feature extraction model may not excel in class discrimination, it captures an extensive range of class features essential for effective clustering and weighted sampling, thus addressing challenges in training with imbalanced and complex datasets.

5.2.2 Dataloader Comparison

In addressing the first research question concerning the issue of data imbalance in the DAA dataset, this section evaluates whether unsupervised learning techniques can effectively rectify this imbalance. Previous studies, as outlined in the chapter 2, provide various methods to tackle dataset imbalances, each with distinct advantages and limitations. This thesis proposes an unsupervised learning-based data loading strategy termed “Clustered Feature Weighting,” aiming to enhance batch balance during data loading for training deep learning models.

Experiment Setup: To assess the efficacy of the “Clustered Feature Weighting” strategy against traditional data loading methods, where training data is loaded without addressing imbalances, an experimental comparison was conducted. Detailed methodologies are presented in the methodology section; here, we focus exclusively on the experimental results.

Settings for Clustering & Weighting:

- **Algorithm**: HDBSCAN
- **Metric**: Batchwise Cosine Distance Matrix
- **Minimum Cluster Size**: 25
- **Minimum Samples (min_samples)**: 1
- **Cluster Selection Epsilon**: 0
- **Metric Type**: Precomputed
- **Cluster Selection Method**: EOM
- **Allow Single Cluster**: No
- **Sample Weighting**: $\frac{1}{\text{Number of samples in the cluster}}$
- **Outlier Weight**: 0.001

Results: Figure 5.2 illustrates the KL divergence for dataloader comparison. This plot compares the KL divergence between the ideal uniform distribution per batch per category (shown in red) and the distributions achieved by Traditional Dataloader A (blue) and Clustered Feature Weighting Dataloader B (orange). The y-axis represents the KL divergence value, while the x-axis shows the number of batches with a batch size of 1024 for the split 0 train dataset of Kinect Color Right Top Image DAA, containing 259,865 total image samples. A lower KL divergence value indicates a closer approximation to the ideal uniform distribution.

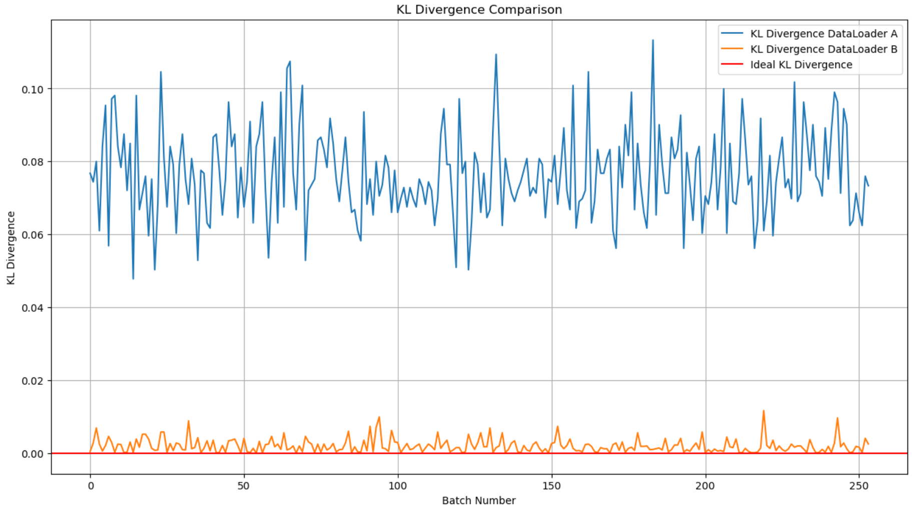


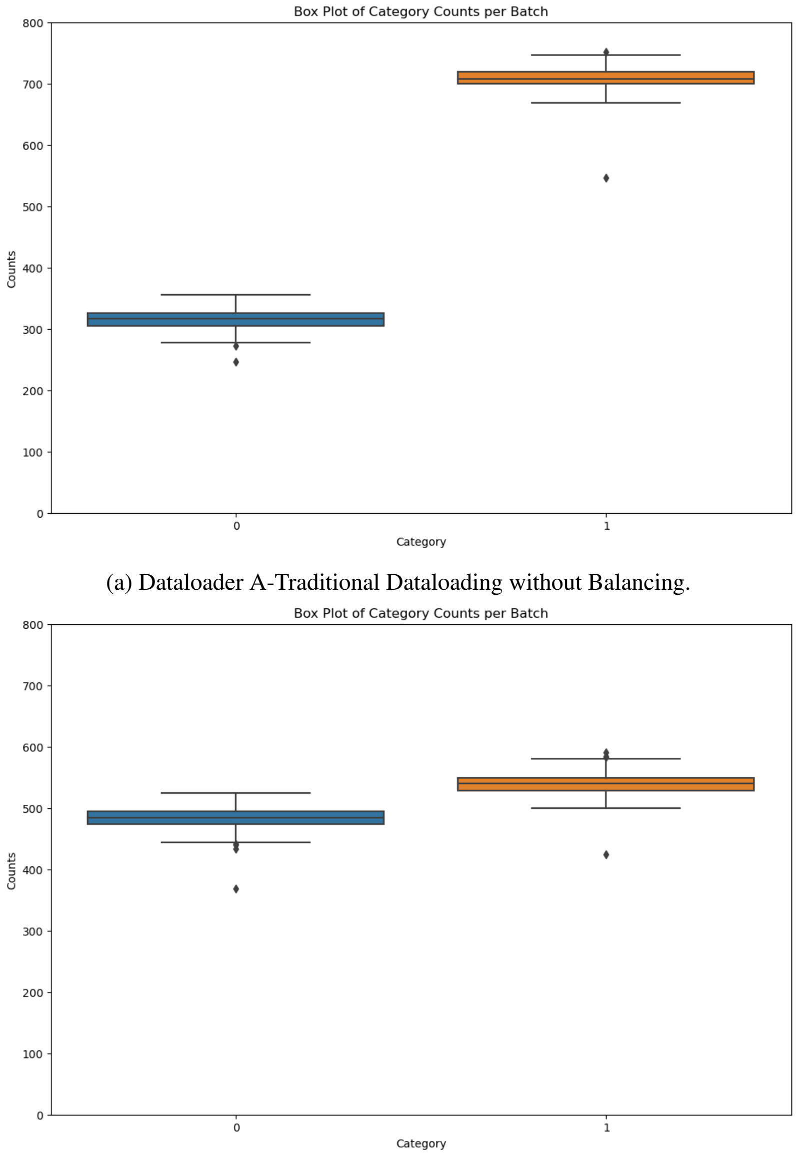
Figure 5.2: Comparative KL Divergence Analysis Across Multiple Batches for Two Dataloading Strategies. This plot visualizes the KL divergence values on the y-axis against the batch numbers on the x-axis, comparing the uniformity of category distribution per batch between the traditional dataloading approach (Dataloader A, shown in blue) and the novel clustered feature weighting strategy (Dataloader B, shown in orange). The plot demonstrates how closely each strategy approximates the ideal uniform distribution across categories, highlighting the effectiveness of the clustered feature weighting in achieving more balanced data loading.

Analysis of KL Divergence Plot:

- **General Trend**: Both lines on the plot denote the KL divergence for each batch, with Dataloader A depicted in blue and Dataloader B in orange.
- **Dataloader A (Blue Line)**: The values fluctuate around 0.08 and higher, suggesting significant divergence from the uniform distribution. This indicates imbalanced class representation within each batch, particularly with over-representation of the ‘distracted’ driver category.
- **Dataloader B (Orange Line)**: The values generally remain below 0.01, close to the ideal zero, which indicates a more balanced class representation within batches.

Interpretation and Assessment:

- **Balanced Sampling**: Dataloader B’s performance, characterized by lower and more stable KL divergence values, demonstrates its effective balanced sampling across categories. This contrasts with Dataloader A, which shows greater batch-to-batch category imbalance.
- **Consistency and Predictability**: Dataloader B exhibits consistent and predictable sampling behavior, essential for stable deep learning training. In contrast, the imbalance observed in Dataloader A could potentially lead to less effective model training and generalization.
- **Suitability for Deep Learning Experiments**: Given the goal of achieving a balanced and unbiased representation of categories in training batches, Dataloader B is preferable for training robust deep learning models. It potentially reduces the risk of overfitting to dominant categories.



(a) Dataloader A-Traditional Dataloading without Balancing.

(b) Dataloader B-With Clustered Feature Weighting Strategy for Balancing.

Figure 5.3: Box Plot Analysis of Sample Counts per Batch for Two Dataloading Strategies. This figure displays box plots that illustrate the distribution of sample counts per batch for nondistracted (Category 0) and distracted (Category 1) driver categories, using both (a) Dataloader A and (b) Dataloader B. The box plots show the spread, central tendency, and outliers for each category within the batches. Dataloader B, which employed an outlier weight of 0.001, demonstrates how this parameter influences the balance and uniformity of sample distribution compared to Dataloader A.

Analysis of Category Counts per Batch Using Box Plots: Figure 5.3 presents a box plot visualization of the counts of samples per batch for two categories of driver distraction: nondistracted (Category 0) and distracted (Category 1) across two different dataloaders under comparison, A and B. These box plots are instrumental in assessing the balance of sample distribution across categories within each batch.

In Dataloader A, see 5.3-(a), the median count for Category 0 (non-distracted) is notably lower at approximately 310 counts, whereas for Category 1 (distracted), it is significantly higher at about 710 counts. This discrepancy indicates a skewed distribution in each batch, which could lead to biased model training due to the over-representation of distracted drivers.

Conversely, Dataloader B, see 5.3-(b), shows a more balanced approach, with the median counts for Category 0 and Category 1 being much closer to each other, around 480 and 530 counts, respectively. These values suggest a more equitable distribution of categories within batches, which is closer to the ideal scenario where each category would ideally comprise half of the batch size of 1024, meaning 512 samples from Category 0 and 512 samples from Category 1. The comparison of these medians between the two dataloaders indicates that Dataloader B is more effective in creating balanced batches.

Impact of Outlier Weight on Sampling: The selection of an outlier weight of 0.001 in our experiments was a deliberate decision informed by extensive empirical testing. This parameter was fine-tuned through a systematic trial-and-error process, exploring a range of values from 0 to 1. The impact of the outlier weight on the sampling process is crucial, as it directly influences the KL divergence values, which measure the discrepancy between the actual data distribution in each batch and the ideal uniform distribution across categories.

Figure 5.4 illustrates the effects of utilizing a higher outlier weight of 0.02 in Dataloader B. The results indicate a significant increase in the KL divergence values, averaging around 0.07, which suggests a deviation from the ideal uniform distribution. This deviation is substantiated by the increased KL divergence, highlighting the sensitivity of the sampling process to the outlier weight setting.

Moreover, the corresponding box plots in Figure 5.5b for categories 0 and 1 reveal a noticeable difference between the medians of counts of each category. This difference underscores a pronounced imbalance in the batch compositions of Dataloader B when a higher outlier weight is employed. The divergence of the medians from each other at this higher outlier weight (0.2) confirms that the distribution of samples across categories becomes significantly skewed, detracting from the efficacy of the dataloading process in maintaining balanced batches.

Dataloader Experiments Conclusion: The comprehensive evaluation of dataloaders in this study highlights the superior performance of Dataloader B, which utilizes a clustered feature weighting strategy proposed in this thesis. This strategy significantly improves the data imbalance in training deep learning models, as demonstrated by the KL divergence and box plot analyses.

Dataloader B consistently achieved lower KL divergence values, indicating a more uniform distribution that aligns closely with the ideal uniform distribution across categories per batch. Additionally, the box plot analysis 5.3 confirmed that Dataloader B provides a more balanced distribution of both ‘non-distracted’ and ‘distracted’ categories within each batch. The median counts of these categories are nearly even, closely approaching the ideal split of the batch size.

In conclusion, Dataloader B ensures a more equitable representation of categories. However, this proposed novel data loading strategy still needs a verification for gain in model learning and generalisation compared to traditional dataloading.

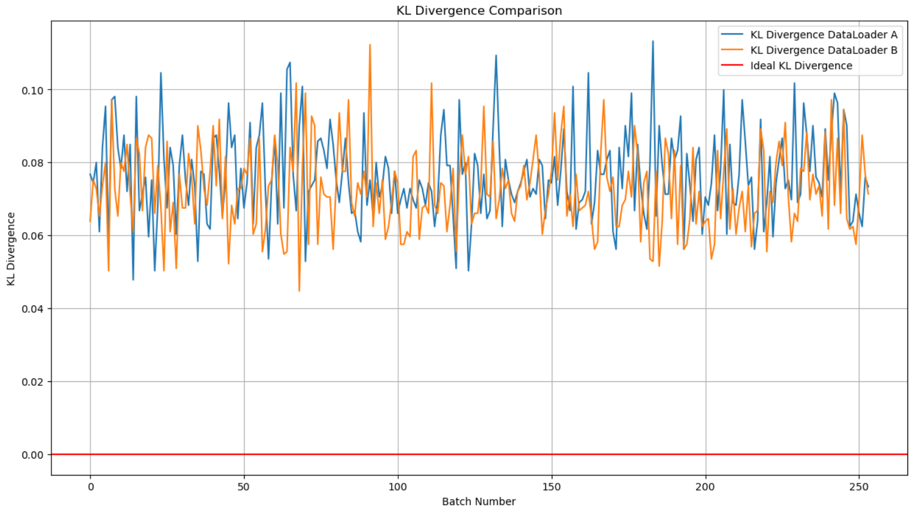


Figure 5.4: KL Divergence Analysis with Increased Outlier Weight. This plot displays the KL divergence values on the y-axis versus the number of batches on the x-axis, comparing the traditional dataloading approach (Dataloader A) and the novel clustered feature weighting strategy (Dataloader B), evaluated with an outlier weight of 0.02. The divergence from the ideal uniform distribution across categories per batch is indicated, with Dataloader B (shown in orange) and Dataloader A (shown in blue), highlighting how the increased outlier weight affects the effectiveness of each dataloading strategy in achieving category balance.

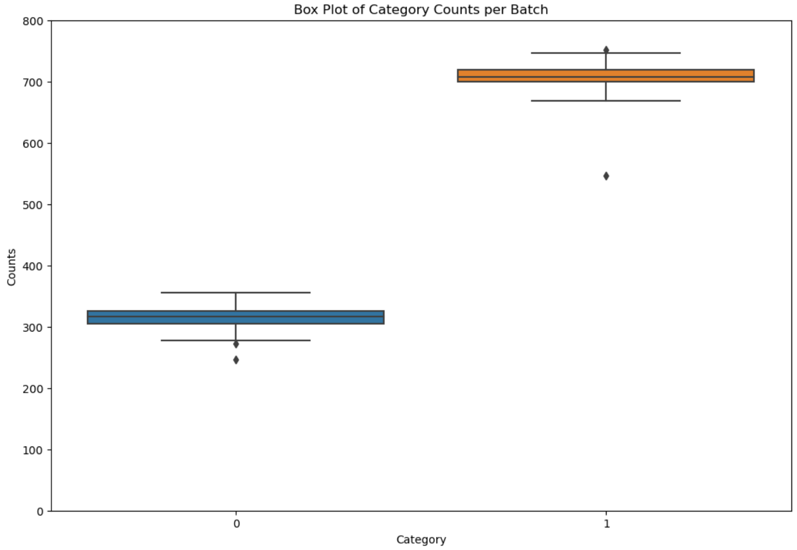
5.3 Experiments based on Model Training and Evaluation

5.3.1 Hyperparameter Grid Search Results

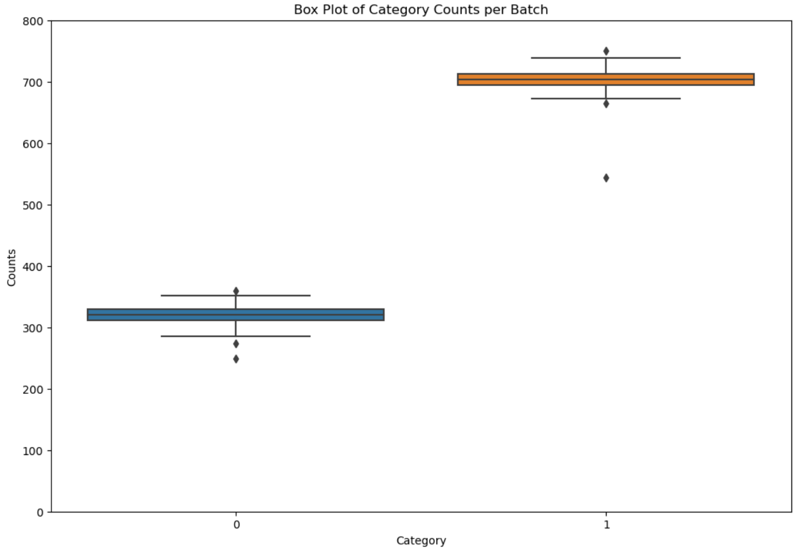
A rigorous hyperparameter search was conducted using a 10% subset of the split 0 of Kinect Color Image DAA dataset, sampled through stratified sampling method to reflect the inherent imbalance of the split 0 of the Kinect Color image DAA dataset. This process aimed to optimize model performance by identifying the most effective hyperparameters for this specific context. Dosovitskiy et al. (2020) recommended the Stochastic Gradient Descent (SGD) (PyTorch Contributors, 2023b) optimizer for transfer learning on vision transformer models due to its suitability in handling pretrained architectures.

The primary objective of the hyperparameter grid search was to ascertain the optimal learning rate and validate the superiority of SGD over Adam (Kingma & Ba, 2014) for fine-tuning linear layer on top of frozen pre-trained vision transformers on the Kinect Color Right Top DAA dataset. Key elements of the initial setup included:

- **Learning Rate Sweep:** A systematic exploration of learning rates—0.003, 0.01, 0.03, 0.06—is conducted to determine the optimal rate for training the linear layer atop the frozen encoder, gauged over 20,000 steps using validation set performance.
- **Steps and Epochs Calculation:** Given the 259,865 images in the train dataset of split 0 of the Kinect Color DAA Image dataset and a batch size of 1024, it takes 254 steps to complete one epoch. To achieve 20,000 steps, approximately 79 epochs are necessary. Consequently, all experiments are extended to 100 epochs to ensure adequate model convergence.



(a) Dataloader A- Traditional Dataloading without Balancing.



(b) Dataloader B- With Clustered Feature Weighting Strategy for Balancing.

Figure 5.5: Box Plot Analysis of Category Counts per Batch with an Increased Outlier Weight for Dataloader B. This figure presents box plots depicting the distribution of sample counts per batch for nondistracted (Category 0) and distracted (Category 1) driver categories, utilizing both Dataloader A and Dataloader B. These plots illustrate the spread, central tendency, and identification of outliers within each category's distribution across batches. Notably, Dataloader B is evaluated using an increased outlier weight of 0.02, highlighting the impact of this weight adjustment on the balance and uniformity of sample distribution compared to Dataloader A.

- **Additional Parameters:** The search also included fine-tuning parameters such as a momentum of 0.9, no weight decay, and gradient clipping at a global norm of 1 as used in the (Dosovitskiy et al., 2020) for fine tuning. The resolution for fine-tuning was set at 224 pixels.

To ensure stable training conditions, the hyperparameter search was expanded to include various learning rate schedulers like Linear Decay, Step Decay, Exponential Decay, and Constant LR, alongside necessary adjustments for each strategy. Appendix section provides more details about the results of these experiments.

Experiment results, as shown in Table 5.2, indicate that training balanced accuracies are consistently high (greater than 92%), whereas validation balanced accuracies are considerably lower (83.45% to 83.87%), suggesting a potential overfitting issue. Notably, the learning rate adjustments between experiments varied, with Experiment 22 starting at 4.00×10^{-4} and reducing to 2.00×10^{-4}, and Experiment 23 implementing a higher initial rate that decreased more significantly to 1.00×10^{-4}. These configurations resulted in the smallest gaps between training and validation accuracies (8.18% and 8.17%, respectively), indicating better generalization compared to Experiment 25, which had a larger gap of 9.18%. These findings underscore the importance of finely tuned learning rate schedules in balancing between achieving high training accuracy and ensuring good generalization to unseen data.

Using the Adam optimizer with cosine annealing significantly lowered performance in terms of validation accuracy; however, utilizing Adam with a linear decay scheduler, as depicted in Experiment 24 in appendix section in Table 6.2, yielded a validation balanced accuracy of 80.86%, with a pronounced discrepancy of 15.80% between training and validation accuracies. In total, 16 experiments combining the Adam and SGD optimizers with the Cosine Annealing scheduler were conducted. The maximum number of iterations (T_{max}) set for the cosine annealing was 100 and 10, as shown in the appendix section in Table 6.1. These experiments consistently resulted in a balanced accuracy gap exceeding 10%, indicative of substantial overfitting. Based on the hyperparameter settings and experiments conducting using Adam and SGD optimizer, SGD optimizer performed better than Adam optimizer which aligns with the choice of (Dosovitskiy et al., 2020) for fine tuning vision transformer on downstream tasks like image classification on custom datasets.

Further analysis involving the SGD optimizer paired with a Linear Decay scheduler and varying initial and final learning rates, detailed in appendix section in Table 6.2, revealed that configurations with lower starting and ending rates effectively decreased the overfitting issue. Similarly, using the Step Decay scheduler with a decay factor of 0.1 every 20 epochs also led to overfitting, with a gap exceeding 10%. This pattern persisted in four experiments utilizing an Exponential Decay scheduler with the SGD optimizer, refer to appendix section in Table 6.4, each also exhibiting significant overfitting. To assess the impact of a constant learning rate on overfitting, two additional experiments were conducted with learning rates of 0.001 and 0.003, refer to appendix section in Table 6.5, resulting in gaps of 13.70% and 14.64%, respectively, and validation accuracies of 81.07% and 81.39%.

Considering these outcomes, the hyperparameter settings from Experiment 22, which demonstrated a smaller gap between training and validation balanced accuracies, are adopted as the baseline for main experiments, as illustrated in the table 5.1.

5.3.2 Results of Experiment 1: Supervised Learning Based Encoder

Following the methodology explained in section 4.4.1, in this experiment, we adapted a pre-trained Vision Transformer (vit_b_16) model, pre-trained on the ImageNet-1K dataset, for the driver distraction detection task using the Kinect Color Right Top view dataset. We replaced the (vit_b_16) classifier layer with a linear layer, training only this component to leverage the model's transfer learning and feature extraction capabilities. The hyperparameters selected in the previous section played a crucial role in training our model. These included a linear decay scheduler for the learning rate, starting at 4.00×10^{-4} and decreasing to 2.00×10^{-4} over 100 epochs, as illustrated in Figure 5.6. The model was trained with an effective batch

Table 5.1: Chosen hyperparameter configurations

PARAMETER	VALUE
Experiment Number	22
Epochs	100
Effective Batch Size	1024
Number of GPUs	2
Batch Size per GPU	512
Optimizer	SGD
Scheduler	LinearDecay
Initial LR	4.00×10^{-4}
End LR	2.00×10^{-4}
Train Balanced Accuracy	92.05%
Validation Balanced Accuracy	83.87%
Balanced Accuracy Gap	8.18%

Table 5.2: Top competing hyperparameter configurations

Experiment Details		Learning Rate	Balanced Accuracy		
Exp No	**Optimizer-Scheduler**	**Initial LR - End LR**	**Train**	**Validation**	**Gap**
22	SGD - LinearDecay	$4.00 \times 10^{-4} - 2.00 \times 10^{-4}$	92.05%	83.87%	8.18%
23	SGD - LinearDecay	$5.00 \times 10^{-4} - 1.00 \times 10^{-4}$	92.00%	83.83%	8.17%
25	SGD - LinearDecay	$4.50 \times 10^{-4} - 1.50 \times 10^{-4}$	92.62%	83.45%	9.18%

size of 1024, distributed across two GPUs using the Distributed Data-Parallel (DDP) algorithm. The results of this experiment are presented in Table 5.3.

The average training balanced accuracy across all splits is notably high at 95.88%, demonstrating the model's robust ability to learn from the training data. However, the validation balanced accuracies are considerably lower, averaging 78.16%, which indicates overfitting. The average test accuracy is somewhat higher at 81.38%, indicating a reasonable generalization to unseen data.

Validation-balanced accuracies are much lower than training-balanced accuracies, indicating that the model struggles to generalize across dataset subsets. This discrepancy may be due to the unique Kinect Color Right Top view image dataset or limits in the transferability of pre-trained features to driver distraction detection task and requires further research in this direction. Similarly, the significant difference between training and validation balanced accuracy in Table 5.3, especially in Split 1, shows overfitting.

Table 5.3: Supervised learning based encoder results on driver distraction detection task. The linear layer on top of pretrained frozen encoder (vit_b_16) is trained and evaluated on all three splits of Kinect Color Right Top DAA image dataset for 100 epochs using DDP algorithm on dual GPU setup with an effective batch of 1024. The results are avearged for 100th epoch for comparison with subsequent experiments.

Splits	View	Encoder	Epoch	Train	Val	Test
Split_0	Right Top	vit_b_16	100	96.10%	80.32%	85.12%
Split_1	Right Top	vit_b_16	100	96.12%	72.69%	79.49%
Split_2	Right Top	vit_b_16	100	95.42%	81.48%	79.53%
Average	Right Top	vit_b_16	100	95.88%	78.16%	81.38%

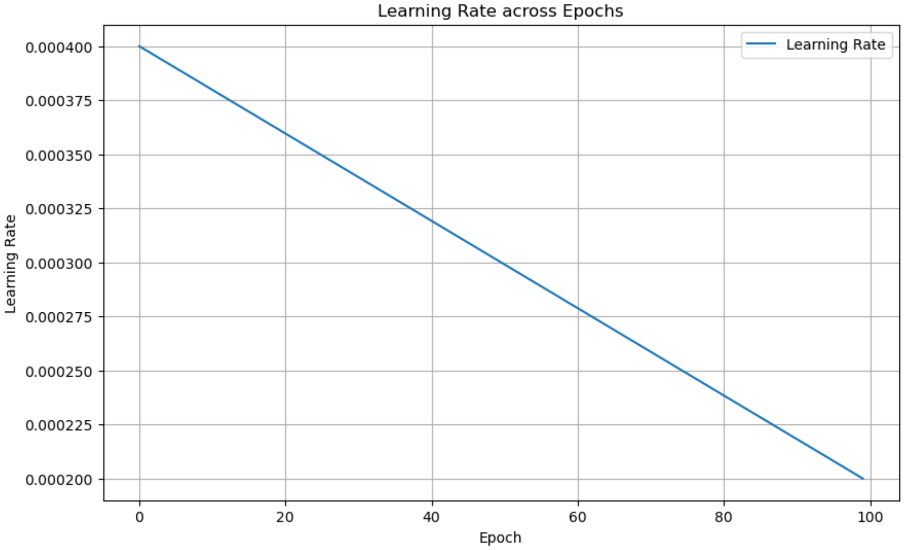


Figure 5.6: Linear learning rate decay: the learning rate schedule starts at initial learning rate of 0.0004 and ends at 0.0002 after 100 epochs. The epochs on the x-axis are 0 indexed.

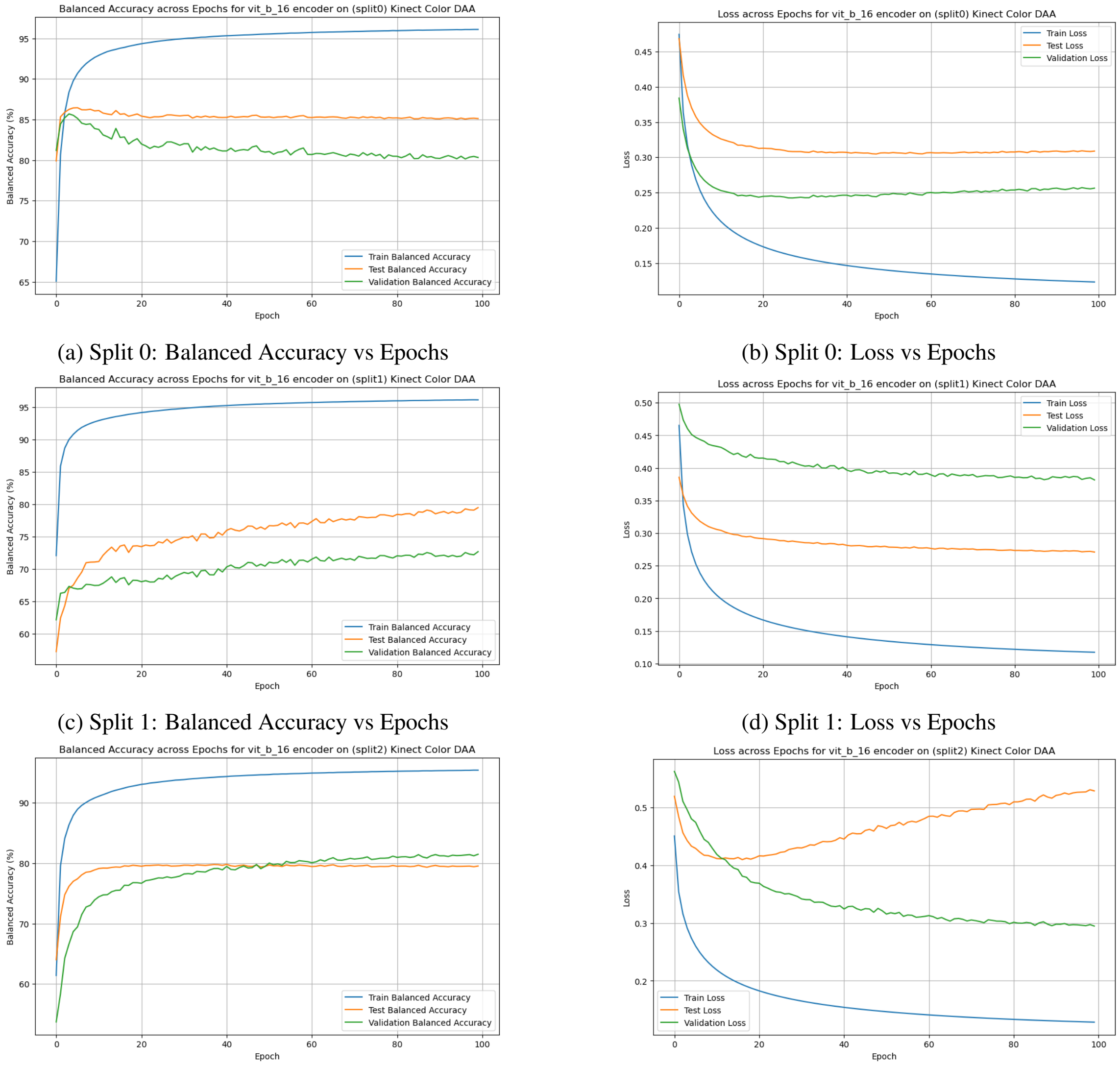
Figure 5.7 shows this experiment's balanced accuracy and loss curves. It can be seen in the figure that each split exhibits high initial training balanced accuracies, quickly reaching a plateau. This behavior underscores the effective utilization of the pre-trained features of the ViT-B/16, which is adept at adapting quickly to the training data. However, the validation and test accuracies across the splits show varied patterns. Test-balanced accuracies generally stabilize at higher levels than validation, suggesting better generalization on the test set. Notably, Split 2 shows a closer convergence between validation and test balanced accuracies, indicating more effective generalization compared to Splits 0 and 1.

Similarly, the loss curves in the figure 5.7 show sharp initial declines in training loss in all splits, reflecting efficient error minimization on training data. However, the validation loss behaviors vary, with Split 2 displaying an increasing trend after initial stabilization—signaling potential overfitting or inadequate model tuning for generalization.

The trained linear layer on top of the frozen supervised learning-based encoder (vit_b_16) in this experiment is further tested for cross-modality and cross-view generalization on the Kinect IR and NIR DAA test datasets across all splits, as described in sections 4.4.5 and 4.4.6. Tables 5.4 and 5.5 shows the model's performance on Kinect IR and NIR DAA image datasets with different camera view and image modality than the training dataset.

Cross-Modality and Cross-View Generalization: The performance on the Kinect IR Right Top view dataset (Table 5.4) shows that the model achieves an average test balanced accuracy of 53.40%. This result is considerably lower than the training balanced accuracies observed on the Kincet Color DAA dataset, highlighting challenges in the model's ability to generalize to grayscale IR images. Individual splits show variability, with the highest being 57.83% in Split 1 and the lowest at 49.96% in Split 0, suggesting some inconsistency in the model's performance across different segments of the Kinect IR dataset.

Further evaluating the model's generalization capabilities, the NIR Front Top view dataset (Table 5.5) shows even more uniform results with a narrow range of test balanced accuracies hovering around 49%. The aver-



(a) Split 0: Balanced Accuracy vs Epochs

(b) Split 0: Loss vs Epochs

(c) Split 1: Balanced Accuracy vs Epochs

(d) Split 1: Loss vs Epochs

(e) Split 2: Balanced Accuracy vs Epochs

(f) Split 2: Loss vs Epochs

Figure 5.7: Comparison of training and evaluation based on balanced accuracy an loss curves plotted across epochs for all three splits of Kinect Color Right Top view DAA Image dataset for supervised learning based encoder in experiment 1.

Table 5.4: Performance of the Linear Layer trained on top of the Frozen Supervised Encoder on Kinect IR Right Top View Dataset: This table shows the results from evaluating the 100th checkpoint of the model, initially trained on color images, on the grayscale test sets of the Kinect IR Right Top view dataset.

Splits	View	Encoder	Checkpoint	Test
Split_0	Right Top	vit_b_16	100	49.96%
Split_1	Right Top	vit_b_16	100	57.83%
Split_2	Right Top	vit_b_16	100	52.42%
Average	KIR Right Top	vit_b_16	100	53.40%

age balanced accuracy across the splits stands at 49.24%, which is slightly lower than the Kinect IR dataset results. This indicates additional challenges when the model encounters not only a different modality but also a different viewing angle, emphasizing the specificity of the learned features to the training conditions.

Table 5.5: Performance of the Linear Layer trained on top of the Frozen Supervised Encoder on NIR Front Top View DAA Dataset: This table shows the results from evaluating the 100th checkpoint of the model, initially trained on color images, on the grayscale test sets of the NIR Front Top view dataset. Here model evaluation is performed on gray scale images and front top view resulting in cross-modality and cross-view generalization evaluation.

Splits	View	Encoder	Checkpoint	Test
Split_0	Front Top	vit_b_16	100	49.79%
Split_1	Front Top	vit_b_16	100	49.53%
Split_2	Front Top	vit_b_16	100	48.41%
Average	NIR Front Top	vit_b_16	100	49.24%

These results demonstrate the limitations of applying models trained on specific datasets and modalities to other dataset without adaptation or training. The model's feature extraction abilities appear to be highly related to the training data's features, as color to grayscale images and viewing perspectives significantly decrease performance. Domain adaptation, fine-tuning on target domain datasets, and more diversified training data including many modalities and perspectives may improve cross-modality and cross-view generalization.

In conclusion, while the linear layer trained on top of frozen encoder vit_b_16 demonstrates robust learning capabilities within its training data modality, its application to significantly different data modality without prior adaptation highlights the critical need for models that better generalize across diverse input conditions. Further research into transfer learning and domain generalization techniques would be beneficial to address these challenges and improve the practical utility of such models in real-world applications like driver distraction detection.

5.3.3 Results of Experiment 2: Supervised Learning Based Encoder with Gray scale Augmentation

The results of previous experiment show that without prior training on grayscale image data, the linear layer trained on top of a frozen supervised learning-based encoder (vit_b_16) does not generalize effectively on cross-modality and cross-view dataset images. This experiment uses the approach described in section 4.4.2 to assess the model's generalization capabilities while providing grayscale transforms during training. This experiment employed the same training setup as experiment 1, with the main variation being that grayscale transforms were applied to the images fed into the model training. In other words, the color images in the Kinect Color Right Top view dataset are first transformed to grayscale before being given into the model for further training of the linear layer on top of the (vit_b_16) encoder. Table 5.6 summarizes the findings of this experiment.

Table 5.6: Results of Experiment 2: Supervised Learning Based Encoder with Gray scale augmentation

Splits	View	Augmentation	Encoder	Train	Val	Test
Split_0	Right Top	Gray Scale	vit_b_16	95.61%	68.43%	62.25%
Split_1	Right Top	Gray Scale	vit_b_16	95.88%	64.28%	86.64%
Split_2	Right Top	Gray Scale	vit_b_16	95.12%	87.73%	84.52%
Average	Right Top	Gray Scale	vit_b_16	95.53%	73.48%	77.80%

Table 5.6 shows a high average training balanced accuracy of 95.53% across all splits, indicating the model's strong capacity to learn from augmented grayscale training data. However, the validation balanced accuracies are significantly lower, averaging 73.48 percent, indicating overfitting. The average test balanced accuracy is slightly better at 77.80%, demonstrating a decent generalization to unseen augmented grayscale data.

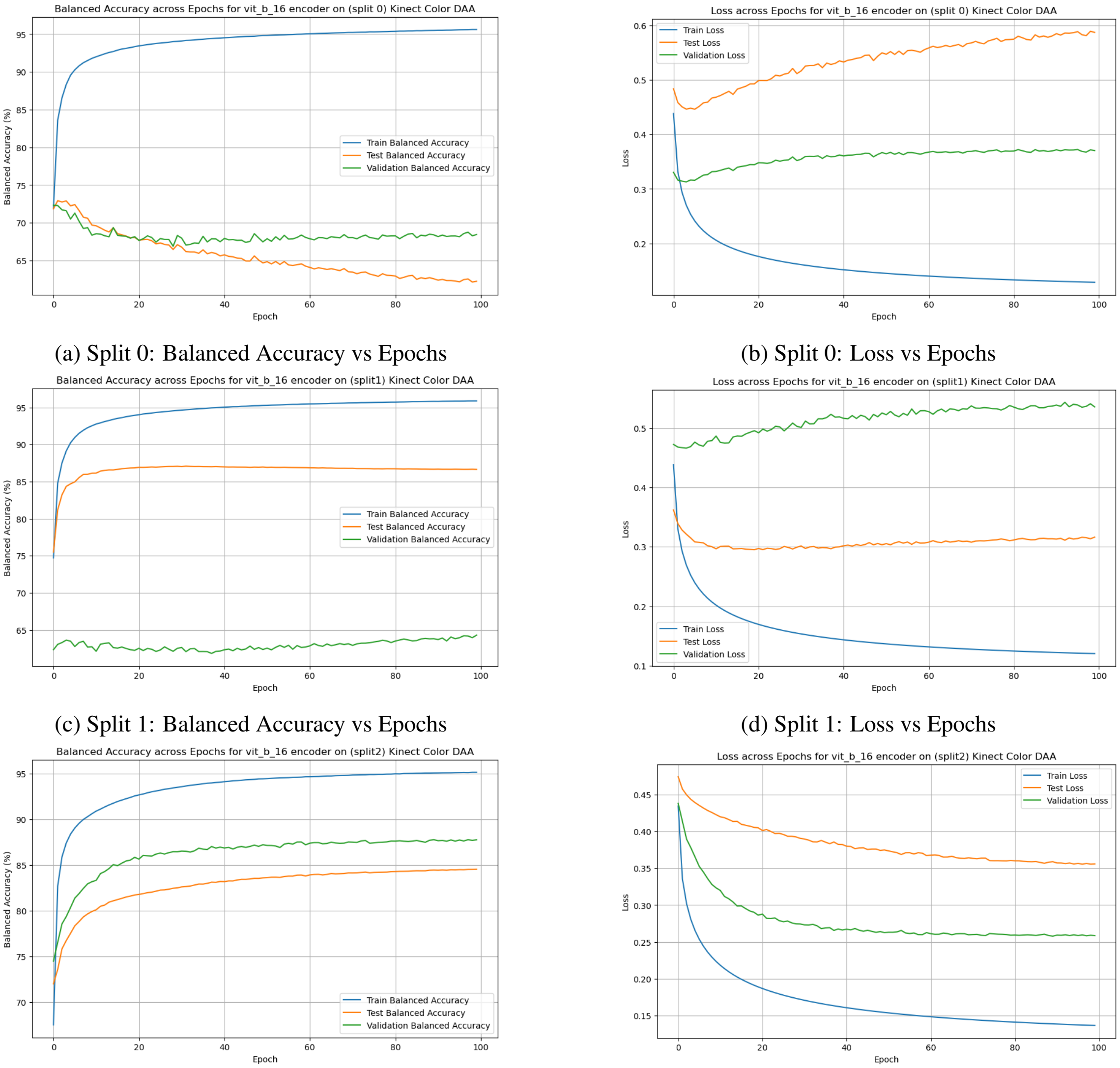
Validation-balanced accuracies are much lower than training-balanced accuracies, indicating that the model trained using grayscale augmentations struggles to generalize across dataset subsets except for split 2, where the validation-balanced accuracy, at 87.73%, is slightly better than the other two splits. Similarly, the test-balanced accuracy for split 1 and split 2 are much higher than those for split 0.

Figure 5.8 depicts the balanced accuracy and loss curves from this experiment. The figure shows that each split demonstrates high initial training balanced accuracies before shortly plateauing. This behavior highlights the effective exploitation of the pre-trained features of the vit_b_16 encoder, which can quickly adjust to the augmented training data. However, the validation and test accuracies throughout the splits exhibit different patterns. Split 1 test-balanced accuracy tends to stabilize at a greater level than validation. On the contrary, for split 2, validation-balanced accuracy stabilizes at a higher level than the test. For split 1, both validation and test-balanced accuracies show a decreasing trend, which indicates overfitting. Notably, Split 2 shows better convergence for validation and test-balanced accuracies, indicating more effective generalization compared to splits 0 and 1.

The loss curves in Figure 5.8 show sharp early drops in training loss in all splits, indicating efficient error minimization on the augmented grayscale training data. However, the validation loss (green) behaviors differ, with Split 1 showing an increasing trend from the start, indicating potential overfitting. On the other hand, for split 1, the test loss (orange) curve is much better and much closer to the train loss curve than the validation loss curve. In the split 0 loss curve plot, the test loss curve increases after the first few epochs, indicating poor generalization, as shown in the split 0 balanced accuracy plot.

Like the last, this experiment evaluates the model trained on a grayscale augmented Kinect Color Right Top DAA image dataset across cross-modality and views. The data sets utilized for this evaluation are identical to those used in previous experiment. Tables 5.7 and 5.8 demonstrate the model's performance on the Kinect IR and NIR DAA image datasets.

Cross-modality and cross-view generalization: The performance on the Kinect IR Right Top view dataset (Table 5.7) shows that the model achieves an average test balanced accuracy of 54.86%. This result is considerably lower than the train and test balanced accuracies observed on the augmented Kinect Color DAA dataset. However, compared to the cross-modality generalization results from experiment 1 on the Kinect IR Right Top dataset, there is an increment of 1.46% from 53.40% to 54.86%. This slight increment highlights the model's challenges in generalizing to grayscale IR images. Individual splits show variability, with the highest being 60.66% in Split 2 and the lowest at 50.58% in Split 0, suggesting some inconsistency in the model's performance across different segments of the Kinect IR dataset.



(a) Split 0: Balanced Accuracy vs Epochs

(b) Split 0: Loss vs Epochs

(c) Split 1: Balanced Accuracy vs Epochs

(d) Split 1: Loss vs Epochs

(e) Split 2: Balanced Accuracy vs Epochs

(f) Split 2: Loss vs Epochs

Figure 5.8: Comparison of training and evaluation based on balanced accuracy an loss curves plotted across epochs for all three splits of Kinect Color Right Top view DAA Image dataset for supervised learning based encoder with Gray Scale Augmentation in experiment 2.

Table 5.7: Results of Experiment 2 for Cross Modality Generalisation on test sets of Kinect IR DAA Image Dataset

Splits	View	Encoder	Checkpoint	Test
Split_0	Right Top	vit_b_16	100	50.58%
Split_1	Right Top	vit_b_16	100	53.34%
Split_2	Right Top	vit_b_16	100	60.66%
Average	KIR Right Top	vit_b_16	100	54.86%

Table 5.8: Results of Experiment 2 for Cross Modality and Cross view Generalisation on test sets of NIR DAA Image Dataset

Splits	View	Encoder	Checkpoint	Test
Split_0	Front Top	vit_b_16	100	49.68%
Split_1	Front Top	vit_b_16	100	49.97%
Split_2	Front Top	vit_b_16	100	49.17%
Average	NIR Front Top	vit_b_16	100	49.60%

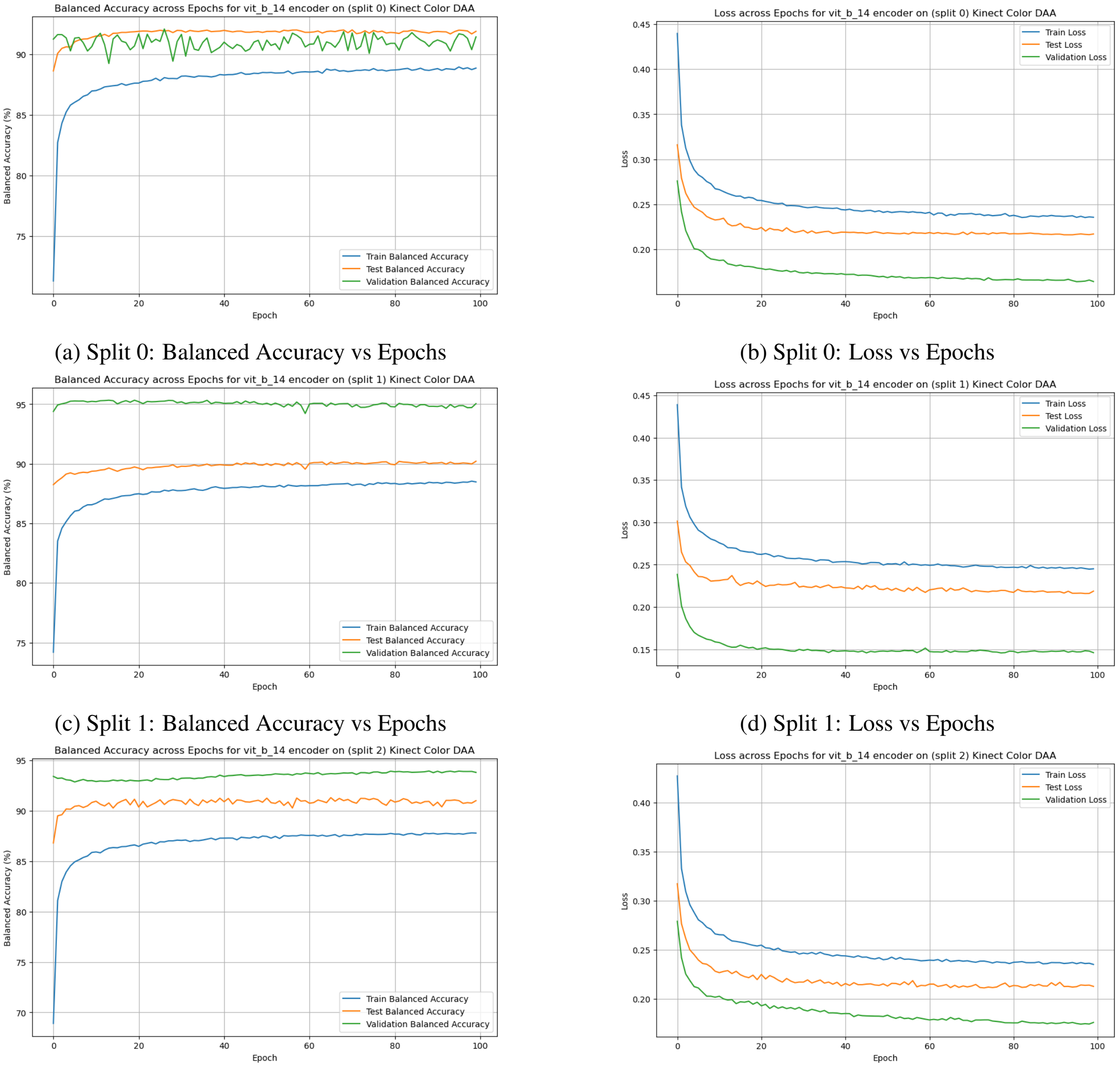
The NIR Front Top view dataset (Table 5.8) demonstrates consistent results with test balanced accuracies around 49%. The average balanced accuracy across splits is 49.60%, somewhat higher than experiment 1 findings on the NIR Front top view DAA dataset. This finding suggests that cross-view generalization remains challenging when the model faces a new viewing angle, stressing the relevance of the learnt features in the pre-trained encoders to the training conditions. This also emphasizes the importance of a foundational vision model for such complicated generalization tasks, which leads this thesis to encoders trained utilizing self-supervised learning approaches on huge curated datasets without labels, as done in the (Oquab et al., 2023).

5.3.4 Results of Experiment 3: Self-Supervised Learning Based Encoder

As demonstrated by the results of experiment 2, by providing prior knowledge about the modality, such as augmenting color images to grayscale prior to model training, we can progress toward increasing cross-modality generalization to some extent; however, cross-view generalization still requires a robust encoder whose knowledge can be transferred to the cross-view generalization. Following the methodology in section 4.4.3, in this experiment, we have replaced the supervised encoder with the self-supervised encoder (vit_b_14), which is trained using the DINOv2 (Oquab et al., 2023) self-supervised learning approach on a huge curated dataset LVD-142M (Oquab et al., 2023) with 142 million images without labels. This experiment's setup is identical to that of experiment 1, using the same hyperparameters to provide a fair performance comparison.

Table 5.9 shows an average training balanced accuracy of 88.36% across all splits, indicating the model's strong capacity to learn from Kinect Right Top view color training data. However, the validation balanced accuracies are significantly higher, averaging 93.42 percent, indicating underfitting. The average test-balanced accuracy is slightly lower than validation-balanced accuracy at 91.02%, demonstrating a decent generalization to unseen data.

Validation-balanced accuracies are much higher than training-balanced accuracies, indicating that the model trained using Kinect color right top view dataset generalizes across the dataset well. Similarly, the test-balanced accuracy also supports this argument by showing higher test-balanced accuracies than the training-balanced accuracies across all splits for the Kinect color right top view dataset. Figure 5.9 depicts the



(a) Split 0: Balanced Accuracy vs Epochs

(b) Split 0: Loss vs Epochs

(c) Split 1: Balanced Accuracy vs Epochs

(d) Split 1: Loss vs Epochs

(e) Split 2: Balanced Accuracy vs Epochs

(f) Split 2: Loss vs Epochs

Figure 5.9: Comparison of training and evaluation based on balanced accuracy an loss curves plotted across epochs for all three splits of Kinect Color Right Top view DAA Image dataset for self supervised learning based encoder in experiment 3.

Table 5.9: Results of Experiment 3: Self-Supervised Learning Based Encoder

Splits	**View**	**Encoder**	**Train**	**Val**	**Test**
Split_0	Right Top	DINOv2_vit_b_14	88.83%	91.41%	91.87%
Split_1	Right Top	DINOv2_vit_b_14	88.48%	95.04%	90.21%
Split_2	Right Top	DINOv2_vit_b_14	87.79%	93.81%	91.00%
Average	Right Top	DINOv2_vit_b_14	88.36%	93.42%	91.02%

balanced accuracy and loss curves from this experiment. The figure shows that each split demonstrates high initial validation and test balanced accuracies before shortly plateauing.

The train balanced accuracy curves in the figure 5.9 for all three splits reveals that although the model is underfitting, there is a lot higher generalization on unseen test data than in the prior two experiments. This demonstrates the robustness and powerful features that the DINOv2-based vit_b_14 encoder has. The loss curves in the figure 5.9 also support the underfitting on all three splits. This suggests the need for different hyperparameter configurations for this experiment, which contributes to the future scope of this thesis.

Similar to the earlier two experiments, this experiment evaluates the model trained on a Kinect Color Right Top view DAA image dataset across cross-modality and views. The data sets utilized for this evaluation are identical to those used in earlier two experiments. Tables 5.10 and 5.11 demonstrate the model's performance on the Kinect IR and NIR DAA image datasets.

Cross-Modality and Cross-View Generalization: The performance on the Kinect IR Right Top view dataset (Table 5.10) shows that the model achieves an average test balanced accuracy of 60.57%. This result is considerably lower than the test balanced accuracies observed on the Kinect Color Right Top view DAA dataset. However, compared to the cross-modality generalization results from experiment 1 and experiment 2 on the Kinect IR Right Top view dataset, there is an increment of 7.17% from 53.40% in experiment to 60.57% in experiment 3 and 5.71% from 54.86% in experiment 1 to 60.57% in experiment 3. This shows the superiority of the self-supervised learning-based encoder over the supervised learning-based encoder under consideration in this thesis on cross-modality generalization.

Table 5.10: Results of Experiment 3 for Cross Modality Generalisation on test sets of Kinect IR DAA Image Dataset

Splits	**View**	**Encoder**	**Checkpoint**	**Test**
Split_0	Right Top	DINOv2_vit_b_14	100	71.80%
Split_1	Right Top	DINOv2_vit_b_14	100	56.64%
Split_2	Right Top	DINOv2_vit_b_14	100	53.27%
Average	KIR Right Top	DINOv2_vit_b_14	100	60.57%

Table 5.11: Results of Experiment 3 for Cross Modality and Cross view Generalisation on test sets of NIR DAA Image Dataset

Splits	**View**	**Encoder**	**Checkpoint**	**Test**
Split_0	Front Top	DINOv2_vit_b_14	100	50.05%
Split_1	Front Top	DINOv2_vit_b_14	100	51.10%
Split_2	Front Top	DINOv2_vit_b_14	100	50.49%
Average	NIR Front Top	DINOv2_vit_b_14	100	50.54%

Further evaluating the model's generalization capabilities, the NIR Front Top view dataset (Table 5.11) shows test balanced accuracies hovering around 50%. The average balanced accuracy across the splits on the NIR Front top view DAA dataset stands at 50.54%, which is 1.3% greater than the one in experiment 1 and 0.94% greater than the one in experiment 2. However, no significant improvement has been obtained, even using a very powerful encoder on cross-view and cross-modality generalization on the NIR Front Top view dataset. This shows that in order to improve the cross-view and cross-modality generalization on the NIR Front Top view dataset, the model needs prior training on the train sets of this dataset. To further enhance the performance on the NIR Front Top view dataset, the encoder can be completely fine-tuned on the datasets by unfreezing its pre-trained parameters for training, which will be computationally expensive given the large size and three splits of the DAA datasets.

5.3.5 Results of Experiment 4: Self-Supervised Learning Based Encoder with Clustered Feature Weighting Data-loading

In our previous three experiments, we assessed the efficacy of various training paradigms using a pre-trained vision transformer encoder obtained from supervised and self-supervised learning paradigms. The experiments examined the impact of pre-trained encoders on driver distraction detection. The datasets considered in this thesis were analyzed from a cross-modality and cross-view generalization perspective for driver distraction detection, utilizing the traditional imbalanced data-loading approach.

Nevertheless, this thesis suggests a data loading approach called "Clustered Feature Weighting" to enhance the imbalance in loading data in batches. This experiment evaluates the impact of modifying the dataloading strategy on driver distraction detection performance. Specifically, it focuses on three aspects: the driver distraction detection performance on the Kinect Color Right Top view DAA image dataset, the cross-modality generalization for driver distraction detection on the Kinect IR Right Top view DAA image dataset, and the cross-view and cross-modality generalization for driver distraction detection on the NIR Front Top-view DAA image dataset.

So, following the methodology in section 4.4.4, we have used clustered feature data loading with the self-supervised encoder (vit_b_14) in this experiment. This experiment's setup is identical to that of experiment 1, using the same hyperparameters to provide a fair performance comparison.

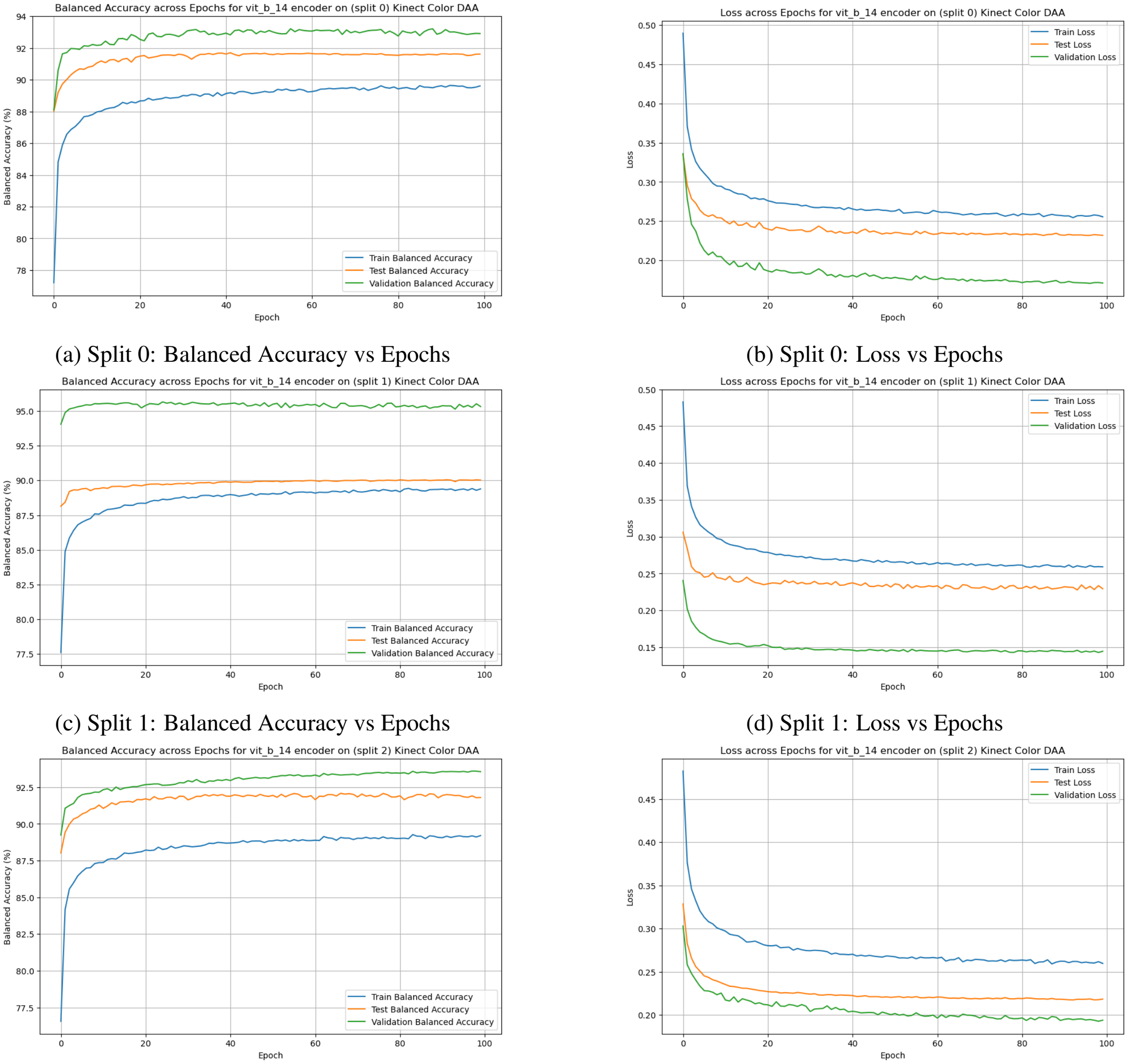
Table 5.12: Results of Experiment 4: Self-Supervised Learning Based Encoder with Clustered Feature Weighting Data-loading

Splits	View	Encoder	Train	Val	Test
Split_0	Right Top	DINOv2_vit_b_14	89.61%	92.91%	91.62%
Split_1	Right Top	DINOv2_vit_b_14	89.38%	95.32%	90.02%
Split_2	Right Top	DINOv2_vit_b_14	89.20%	93.55%	91.79%
Average	Right Top	DINOv2_vit_b_14	89.39%	93.92%	91.14%

Table 5.12 shows driver distraction detection performance on the Kinect Color Right Top view dataset with an average training balanced accuracy of 89.39% across all splits with an improvement of 1.03% from experiment 3.The validation balanced accuracies average 93.92 percent, indicating underfitting using novel data-loading, too. The balanced accuracy and loss curves in figure 5.10 further confirm the phenomenon of underfitting. The average test-balanced accuracy is slightly lower than validation-balanced accuracy at 91.14%, demonstrating a decent generalization to unseen test set data of Kinect Color dataset. In comparison to experiment 3, the train balanced accuracy in this experiment on the Kinect Color Right Top view dataset is 1.03% higher. Additionally, the validation balanced accuracy is 0.5% higher, and the test balanced accuracy is 0.08% higher than in experiment 3. These results indicate that the model requires different hyperparameter configuration to prevent underfitting and assess the overall effectiveness and limitations of the 'Clustered Feature Weighting' data-loading technique on the model's performance on driver distraction detection task.

Cross-Modality and Cross-View Generalization: The performance on the Kinect IR Right Top view dataset (Table 5.13) shows that the model achieves an average test balanced accuracy of 54.37%.

The results from the Kinect IR Right Top view dataset (Table 5.13) indicate that the model attains an average test balanced accuracy of 54.37%. Upon further evaluation of the model's ability to generalize, the NIR Front Top view dataset (Table 5.14) demonstrates an average test balanced accuracy of 50.58% across the splits on the NIR Front top view DAA dataset. This accuracy is 1.34% higher than that of experiment 1, 0.98% higher than that of experiment 2, and 0.08% higher than that of experiment 3. Despite using a powerful encoder and balanced data-loading, there has been no notable increase in comparison to experiment 3 in terms of cross-view and cross-modality generalization on the NIR Front Top view dataset. This demonstrates that,



(a) Split 0: Balanced Accuracy vs Epochs

(b) Split 0: Loss vs Epochs

(c) Split 1: Balanced Accuracy vs Epochs

(d) Split 1: Loss vs Epochs

(e) Split 2: Balanced Accuracy vs Epochs

(f) Split 2: Loss vs Epochs

Figure 5.10: Comparison of training and evaluation based on balanced accuracy an loss curves plotted across epochs for all three splits of Kinect Color Right Top view DAA Image dataset for self supervised learning based encoder with Clustered Feature Weighting Data-loading in experiment 4.

Table 5.13: Results of Experiment 4 for Cross Modality Generalization on test sets of Kinect IR DAA Image Dataset

Splits	View	Encoder	Checkpoint	Test
Split_0	Right Top	DINOv2_vit_b_14	100	59.57%
Split_1	Right Top	DINOv2_vit_b_14	100	52.75%
Split_2	Right Top	DINOv2_vit_b_14	100	50.81%
Average	KIR Right Top	DINOv2_vit_b_14	100	54.37%

Table 5.14: Results of Experiment 4 for Cross Modality and Cross view Generalization on test sets of NIR DAA Image Dataset

Splits	View	Encoder	Checkpoint	Test
Split_0	Front Top	DINOv2_vit_b_14	100	50.05%
Split_1	Front Top	DINOv2_vit_b_14	100	51.00%
Split_2	Front Top	DINOv2_vit_b_14	100	50.69%
Average	NIR Front Top	DINOv2_vit_b_14	100	50.58%

in order to improve the ability to generalize across multiple views and modality in the NIR Front Top view dataset, the problem of underfitting must be addressed. Furthermore, the model must be pre-trained using the training sets from this dataset, as previously described in experiment 3.

5.4 Answers to Research Questions

This section provides detailed answers based on the experimental findings for the central research questions guiding this thesis.

5.4.1 Practical Challenges

The issue of data imbalance in the DAA dataset was effectively addressed by employing the novel ‘Clustered Feature Weighting’ dataloading technique. This technique leverages unsupervised learning, specifically using the HDBSCAN algorithm for clustering based on features extracted by a pretrained vision transformer. The inclusion of a weighted random sampler resulted in balanced batches, indicating consistent training and indications of improvement in the performance of the deep learning models.

5.4.2 Effectiveness of SSL Models

The use of vision transformer encoders pretrained with the SSL method, specifically DINOv2 (Oquab et al., 2023), showed significant benefits in detecting driver distraction. These encoders provided robust feature extraction capabilities, which improved generalization over the supervised learning encoder. The main drawback observed was a slight underfitting in the fixed experimental setup, suggesting that these models might require further tuning of hyperparameters or training procedures to fully capitalize on their potential.

5.4.3 Generalization Capabilities

The different image views, namely the right top view and the front top view from the DAA dataset, demonstrated that the vision transformer encoder pretrained using the SSL approach maintains a relatively consistent performance on driver distraction detection task. However, generalization across these views was not completely uniform, indicating that while the SSL models can handle view variability to a certain extent, there is still room for improvement in model training or architecture to enhance view-invariant feature extraction.

5.4.4 Data Modality Impact

The impact of varying data modalities, such as RGB and infrared (IR) images, was significant on the detection of driver distractions. The DINOv2 (Oquab et al., 2023) pretrained vision transformer encoder exhibited better generalization across these modalities compared to models trained with supervised learning approaches. Although the performance on NIR datasets was less impressive, it still marked an improvement over traditional models, emphasizing the potential of SSL models in handling diverse input types. This suggests a promising direction for future research in using self-supervised learning to develop more adaptable and effective driver distraction detection systems.

Chapter 6

Conclusions and Future Work

This thesis summarizes the findings from chapter 5, based on the initial research questions provided in section 1.3. The subsequent sections present the conclusions and possibilities for future research to stimulate subsequent progress in the discipline.

6.1 Conclusions

This thesis addressed key questions surrounding improved driver distraction detection using self-supervised learning techniques, emphasizing data imbalance solutions, benefits of pretrained vision transformers, and the impact of different image views and data modalities. Our research introduced the 'Clustered Feature Weighting' dataloading technique using HDBSCAN and a weighted random sampler, which proved effective in balancing the data distribution in training batches, enhancing model robustness against overfitting, and improving overall model performance across various datasets and views. The findings showed that using a vision transformer encoder pretrained with self-supervised learning, specifically DINOv2, significantly benefits model performance due to better feature extraction capabilities compared to supervised learning methods. This was particularly evident in the superior cross-modality generalization results observed with self-supervised learning-based encoders.

The experiments conducted highlighted the potential of self-supervised learning to provide high initial balanced accuracies and a more consistent performance across different modalities and views. However, despite the improvements, challenges remain in cross-view generalization, especially in the NIR Front Top view dataset. These results confirm the essential role of customized training and the necessity for domain-specific adaptations in leveraging the full capabilities of advanced deep learning models for practical applications, such as detecting driver distraction.

6.2 Future Work

This thesis has established a foundational approach to using self-supervised learning for driver distraction detection. However, several avenues for enhancement and further exploration remain:

- **Enhancement of Clustered Feature Weighting:** The clustered feature weighting strategy could benefit from the integration of a DINOv2-based encoder in place of the current supervised learning based encoder. This change would likely enhance the quality of feature extraction, which is crucial for effective clustering and subsequently for the performance of the ‘Clustered Feature Weighting’ technique. Improved feature extraction through the DINOv2 encoder may lead to more distinct and informative clusters, potentially increasing the accuracy of the proposed ‘Clustered Feature Weighting’ technique.

- **Hyperparameter Optimization:** Experiment 3 suggested potential underfitting, indicating that different hyperparameter configurations might enhance the model’s learning capacity. Future studies could explore optimization techniques to refine these parameters for better model performance.
- **Extended Pretraining and Fine-Tuning:** Given the challenges in cross-view generalization, particularly with the NIR Front Top view, it may be beneficial to extend pretraining phases or employ fine-tuning on specific datasets to improve the encoder’s adaptability to diverse conditions.
- **Adoption of Hierarchical Transformers:** Modifying the DINOv2 (Oquab et al., 2023) self-supervised approach to incorporate a hierarchical vision transformer model, such as ’Hiera’ (Ryali et al., 2023), as a backbone could provide deeper and more structured feature representations. This adaptation, combined with further fine-tuning on image datasets extracted from the DAA video datasets (Martin et al., 2019), could significantly enhance driver distraction detection capabilities.
- **Improving Generalization Across Modalities and Views:** To advance the cross-modality and cross-view generalization, integrating different views and modalities present in the DAA dataset could be beneficial. Additionally, employing a fusion of vision models with text-guided classification might offer a more comprehensive approach to detect driver distractions by leveraging multiple data types and their inherent correlations.
- **Integration with Cognitive and Audio Data:** For a holistic approach to detecting driver distraction, combining the visual models with audio models and EEG data (which records the brain’s electrical activity) could provide insights into the cognitive state of drivers, thereby enhancing the detection accuracy. This comprehensive approach enables the models to evaluate cognitive component as well as the physical component of driver distraction. The audio models can be used in a feedabck ADAS system to alert the distracted driver.
- **Optimization for Real-Time Applications:** Ensuring the models are practical for in-vehicle use involves optimizing them for size and response time. This optimization includes refining the models to operate efficiently within the computational constraints of real-time systems, ensuring they can function seamlessly in live environments without lag.

Bibliography

Yehya Abouelnaga, Hesham M Eraqi, and Mohamed N Moustafa. Real-time distracted driver posture classification. *arXiv preprint arXiv:1706.09498*, 2017a.

Yehya Abouelnaga, Hesham M. Eraqi, and Mohamed N. Moustafa. Real-time distracted driver posture classification. *CoRR*, abs/1706.09498, 2017b. URL `http://arxiv.org/abs/1706.09498`.

Abien Fred Agarap. Deep learning using rectified linear units (relu). *CoRR*, abs/1803.08375, 2018. URL `http://arxiv.org/abs/1803.08375`.

Astha Agrawal, Herna L Viktor, and Eric Paquet. Scut: Multi-class imbalanced data classification using smote and cluster-based undersampling. In *2015 7Th international joint conference on knowledge discovery, knowledge engineering and knowledge management (IC3k)*, volume 1, pp. 226–234. IEEE, 2015.

Rehan Akbani, Stephen Kwek, and Nathalie Japkowicz. Applying support vector machines to imbalanced datasets. In *Machine Learning: ECML 2004: 15th European Conference on Machine Learning, Pisa, Italy, September 20-24, 2004. Proceedings 15*, pp. 39–50. Springer, 2004.

Hossam Almahasneh, Weng-Tink Chooi, Nidal Kamel, and Aamir Saeed Malik. Deep in thought while driving: An eeg study on drivers' cognitive distraction. *Transportation research part F: traffic psychology and behaviour*, 26:218–226, 2014.

Shin Ando and Chun Yuan Huang. Deep over-sampling framework for classifying imbalanced data. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2017, Skopje, Macedonia, September 18–22, 2017, Proceedings, Part I 10*, pp. 770–785. Springer, 2017.

Rohan Anil, Gabriel Pereyra, Alexandre Passos, Robert Ormandi, George E Dahl, and Geoffrey E Hinton. Large scale distributed neural network training through online distillation. *arXiv preprint arXiv:1804.03235*, 2018.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Randall Balestriero, Mark Ibrahim, Vlad Sobal, Ari Morcos, Shashank Shekhar, Tom Goldstein, Florian Bordes, Adrien Bardes, Gregoire Mialon, Yuandong Tian, et al. A cookbook of self-supervised learning. *arXiv preprint arXiv:2304.12210*, 2023.

Hangbo Bao, Li Dong, Songhao Piao, and Furu Wei. Beit: Bert pre-training of image transformers. *arXiv preprint arXiv:2106.08254*, 2021.

Gustavo EAPA Batista, Ronaldo C Prati, and Maria Carolina Monard. A study of the behavior of several methods for balancing machine learning training data. *ACM SIGKDD explorations newsletter*, 6(1):20–29, 2004.

Jeffrey O Bennett, William L Briggs, and Mario F Triola. *Statistical reasoning for everyday life*. Addison-Wesley Boston, MA, 2003.

Florian Bordes, Randall Balestriero, and Pascal Vincent. Towards democratizing joint-embedding self-supervised learning. *arXiv preprint arXiv:2303.01986*, 2023.

Anna Bosch, Andrew Zisserman, and Xavier Munoz. Image classification using random forests and ferns. In *2007 IEEE 11th international conference on computer vision*, pp. 1–8. Ieee, 2007.

Kay Henning Brodersen, Cheng Soon Ong, Klaas Enno Stephan, and Joachim M Buhmann. The balanced accuracy and its posterior distribution. In *2010 20th international conference on pattern recognition*, pp. 3121–3124. IEEE, 2010.

Cristian Buciluă, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 535–541, 2006.

Mateusz Buda, Atsuto Maki, and Maciej A Mazurowski. A systematic study of the class imbalance problem in convolutional neural networks. *Neural networks*, 106:249–259, 2018.

Ricardo JGB Campello, Davoud Moulavi, and Jörg Sander. Density-based clustering based on hierarchical density estimates. In *Pacific-Asia conference on knowledge discovery and data mining*, pp. 160–172. Springer, 2013.

Ricardo JGB Campello, Davoud Moulavi, Arthur Zimek, and Jörg Sander. Hierarchical density estimates for data clustering, visualization, and outlier detection. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 10(1):1–51, 2015.

Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in neural information processing systems*, 33:9912–9924, 2020.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9650–9660, 2021.

João Carreira and Andrew Zisserman. Quo vadis, action recognition? A new model and the kinetics dataset. *CoRR*, abs/1705.07750, 2017. URL `http://arxiv.org/abs/1705.07750`.

Navoneel Chakrabarty and Sanket Biswas. Navo minority over-sampling technique (nmote): a consistent performance booster on imbalanced datasets. *Journal of Electronics*, 2(02):96–136, 2020.

Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

Nitesh V Chawla, Aleksandar Lazarevic, Lawrence O Hall, and Kevin W Bowyer. Smoteboost: Improving prediction of the minority class in boosting. In *Knowledge Discovery in Databases: PKDD 2003: 7th European Conference on Principles and Practice of Knowledge Discovery in Databases, Cavtat-Dubrovnik, Croatia, September 22-26, 2003. Proceedings 7*, pp. 107–119. Springer, 2003.

Shijie Chen, Yu Zhang, and Qiang Yang. Multi-task learning in natural language processing: An overview. *ACM Computing Surveys*, 2021.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pp. 1597–1607. PMLR, 2020a.

Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey E Hinton. Big self-supervised models are strong semi-supervised learners. *Advances in neural information processing systems*, 33:22243–22255, 2020b.

Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 15750–15758, 2021.

Yunpeng Chen, Haoqi Fan, Bing Xu, Zhicheng Yan, Yannis Kalantidis, Marcus Rohrbach, Shuicheng Yan, and Jiashi Feng. Drop an octave: Reducing spatial redundancy in convolutional neural networks with octave convolution. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 3435–3444, 2019.

François Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1251–1258, 2017.

Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20:273–297, 1995.

Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pp. 886–893. Ieee, 2005.

Li Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

Ketan Ramesh Dhakate and Ratnakar Dash. Distracted driver detection using stacking ensemble. In *2020 IEEE International Students' Conference on Electrical, Electronics and Computer Science (SCEECS)*, pp. 1–5. IEEE, 2020.

Xiaohan Ding, Chunlong Xia, Xiangyu Zhang, Xiaojie Chu, Jungong Han, and Guiguang Ding. Repmlp: Re-parameterizing convolutions into fully-connected layers for image recognition. *arXiv preprint arXiv:2105.01883*, 2021.

Pedro Domingos. Metacost: A general method for making classifiers cost-sensitive. In *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 155–164, 1999.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *ArXiv*, abs/2010.11929, 2020. URL `https://api.semanticscholar.org/CorpusID:225039882`.

John Doucette and Malcolm I Heywood. Gp classification under imbalanced data sets: Active sub-sampling and auc approximation. In *European Conference on Genetic Programming*, pp. 266–277. Springer, 2008.

Jacob Eisenstein. *Introduction to natural language processing*. MIT press, 2019.

Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, volume 96, pp. 226–231, 1996.

Hugging Face. google/vit-huge-patch14-224-in21k. `https://huggingface.co/google/vit-huge-patch14-224-in21k`, 2024. Accessed: 2024-05-15.

Facebook AI Research. Dinov2: Pytorch code and models for the dinov2 self-supervised learning method. `https://github.com/facebookresearch/dinov2`, 2023. [Accessed 14-05-2024].

Wei Fan, Salvatore J Stolfo, Junxin Zhang, and Philip K Chan. Adacost: misclassification cost-sensitive boosting. In *Icml*, volume 99, pp. 97–105, 1999.

Zhiyuan Fang, Jianfeng Wang, Lijuan Wang, Lei Zhang, Yezhou Yang, and Zicheng Liu. Seed: Self-supervised distillation for visual representation. *arXiv preprint arXiv:2101.04731*, 2021.

Tom Fawcett. An introduction to roc analysis. *Pattern recognition letters*, 27(8):861–874, 2006.

Alberto Fernández, Salvador García, Mikel Galar, Ronaldo Cristiano Prati, B. Krawczyk, and Francisco Herrera. Learning from imbalanced data sets. In *Cambridge International Law Journal*, 2018. URL `https://api.semanticscholar.org/CorpusID:53046396`.

Milton Friedman. A comparison of alternative tests of significance for the problem of m rankings. *The Annals of Mathematical Statistics*, 11(1):86–92, 1940. ISSN 00034851. URL `http://www.jstor.org/stable/2235971`.

K. Fukushima. A neural network for visual pattern recognition. *Computer*, 21(3):65–75, 1988. doi: 10.1109/2.32.

Glenn Fung and Olvi L Mangasarian. Proximal support vector machine classifiers. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 77–86, 2001.

Valentina Giglioni, Enrique García-Macías, Ilaria Venanzi, Laura Ierimonti, and Filippo Ubertini. The use of receiver operating characteristic curves and precision-versus-recall curves as performance metrics in unsupervised structural damage classification under changing environment. *Engineering Structures*, 246: 113029, 2021.

Margherita Grandini, Enrico Bagli, and Giorgio Visani. Metrics for multi-class classification: an overview. *arXiv preprint arXiv:2008.05756*, 2020.

Hongyu Guo and Herna L Viktor. Learning from imbalanced data sets with boosting and data generation: the databoost-im approach. *ACM Sigkdd Explorations Newsletter*, 6(1):30–39, 2004.

Hui Han, Wen-Yuan Wang, and Bing-Huan Mao. Borderline-smote: a new over-sampling method in imbalanced data sets learning. In *International conference on intelligent computing*, pp. 878–887. Springer, 2005.

Jiawei Han, Micheline Kamber, and Jian Pei. *Data Mining: Concepts and Techniques*. Elsevier, 2011.

Trevor Hastie, Robert Tibshirani, Jerome H Friedman, and Jerome H Friedman. *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer, 2009.

Simon Haykin. *Neural networks and learning machines, 3/E.* Pearson Education India, 2009.

Haibo He and Edwardo A. Garcia. Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, 21(9):1263–1284, 2009. doi: 10.1109/TKDE.2008.239.

Haibo He, Yang Bai, Edwardo A Garcia, and Shutao Li. Adasyn: Adaptive synthetic sampling approach for imbalanced learning. In *2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence)*, pp. 1322–1328. Ieee, 2008.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016a.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016b.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Doll'ar, and Ross B Girshick. Masked autoencoders are scalable vision learners. 2022 ieee. In *CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15979–15988, 2021.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Tin Kam Ho. Random decision forests. In *Proceedings of 3rd international conference on document analysis and recognition*, volume 1, pp. 278–282. IEEE, 1995.

Xia Hong, Sheng Chen, and Chris J Harris. A kernel-based two-class classifier for imbalanced data sets. *IEEE Transactions on neural networks*, 18(1):28–41, 2007.

Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7132–7141, 2018.

Chen Huang, Yining Li, Chen Change Loy, and Xiaoou Tang. Learning deep representation for imbalanced classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5375–5384, 2016.

Chen Huang, Xiaochen Wang, Jiannong Cao, Shihui Wang, and Yan Zhang. Hcf: A hybrid cnn framework for behavior detection of distracted drivers. *IEEE Access*, 8:109335–109349, 2020. doi: 10.1109/ACCESS.2020.3001159.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167, 2015. URL `http://arxiv.org/abs/1502.03167`.

B Janet, U Srinivasulu Reddy, et al. Real time detection of driver distraction using cnn. In *2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT)*, pp. 185–191. IEEE, 2020.

Nathalie Japkowicz. Supervised versus unsupervised binary-learning by feedforward neural networks. *Machine Learning*, 42:97–122, 2001.

Nathalie Japkowicz and Shaju Stephen. The class imbalance problem: A systematic study. *Intelligent data analysis*, 6(5):429–449, 2002a.

Nathalie Japkowicz and Shaju Stephen. The class imbalance problem: A systematic study. *Intell. Data Anal.*, 6:429–449, 11 2002b. doi: 10.3233/IDA-2002-6504.

Nathalie Japkowicz, Catherine Myers, Mark Gluck, et al. A novelty detection approach to classification. In *IJCAI*, volume 1, pp. 518–523. Citeseer, 1995.

Nathalie Japkowicz et al. Learning from imbalanced data sets: a comparison of various strategies. In *AAAI workshop on learning from imbalanced data sets*, volume 68, pp. 10–15. AAAI Press, Menlo Park, 2000.

Taeho Jo and Nathalie Japkowicz. Class imbalances versus small disjuncts. *ACM Sigkdd Explorations Newsletter*, 6(1):40–49, 2004.

Justin Johnson and Taghi Khoshgoftaar. Survey on deep learning with class imbalance. *Journal of Big Data*, 6:27, 03 2019a. doi: 10.1186/s40537-019-0192-5.

Justin Johnson and Taghi Khoshgoftaar. Survey on deep learning with class imbalance. *Journal of Big Data*, 6:27, 03 2019b. doi: 10.1186/s40537-019-0192-5.

Pilsung Kang and Sungzoon Cho. Eus svms: Ensemble of under-sampled svms for data imbalance problems. In *International conference on neural information processing*, pp. 837–846. Springer, 2006.

Leonard Kaufman and Peter J Rousseeuw. *Finding groups in data: an introduction to cluster analysis*. John Wiley & Sons, 2009.

John D Kelleher, Brian Mac Namee, and Aoife D'Arcy. Fundamentals of machine learning for predictive data analytics: algorithms. *Worked examples, and case studies*, 262029448, 2015.

Seung Wook Kim and Hyo-Eun Kim. Transferring knowledge to smaller network with class-distance loss. 2017.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014. URL `https://api.semanticscholar.org/CorpusID:6628106`.

Bartosz Krawczyk. Learning from imbalanced data: open challenges and future directions. *Progress in Artificial Intelligence*, 5(4):221–232, 2016a.

Bartosz Krawczyk. Learning from imbalanced data: Open challenges and future directions. *Progress in Artificial Intelligence*, 5, 04 2016b. doi: 10.1007/s13748-016-0094-0.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

Miroslav Kubat, Stan Matwin, et al. Addressing the curse of imbalanced training sets: one-sided selection. In *Icml*, volume 97, pp. 179. Citeseer, 1997.

Ajay Kulkarni, Deri Chong, and Feras A. Batarseh. Foundations of data imbalance and solutions for a data democracy. *CoRR*, abs/2108.00071, 2021. URL `https://arxiv.org/abs/2108.00071`.

Solomon Kullback. *Information theory and statistics*. Courier Corporation, 1997.

M Rajya Lakshmi, TV Prasad, and Dr V Chandra Prakash. Survey on eeg signal processing methods. *International journal of advanced research in computer science and software engineering*, 4(1), 2014.

Jorma Laurikkala. Improving identification of difficult small classes by balancing class distribution. In *Artificial Intelligence in Medicine: 8th Conference on Artificial Intelligence in Medicine in Europe, AIME 2001 Cascais, Portugal, July 1–4, 2001, Proceedings 8*, pp. 63–66. Springer, 2001.

Dong-Hyun Lee et al. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, volume 3, pp. 896. Atlanta, 2013.

Bing Li, Jie Chen, Zhixiang Huang, Haitao Wang, Jianming Lv, Jingmin Xi, Jun Zhang, and Zhongcheng Wu. A new unsupervised deep learning algorithm for fine-grained detection of driver distraction. *IEEE Transactions on Intelligent Transportation Systems*, 23(10):19272–19284, 2022a.

Li Li, Boxuan Zhong, Clayton Hutmacher Jr, Yulan Liang, William J Horrey, and Xu Xu. Detection of driver manual distraction via image-based hand and ear recognition. *Accident Analysis & Prevention*, 137:105432, 2020a.

Penghua Li, Yifeng Yang, Radu Grosu, Guodong Wang, Rui Li, Yuehong Wu, and Zeng Huang. Driver distraction detection using octave-like convolutional neural network. *IEEE Transactions on Intelligent Transportation Systems*, 23(7):8823–8833, 2022b. doi: 10.1109/TITS.2021.3086411.

Shen Li, Yanli Zhao, Rohan Varma, Omkar Salpekar, Pieter Noordhuis, Teng Li, Adam Paszke, Jeff Smith, Brian Vaughan, Pritam Damania, and Soumith Chintala. Pytorch distributed: Experiences on accelerating data parallel training. *CoRR*, abs/2006.15704, 2020b. URL `https://arxiv.org/abs/2006.15704`.

Ming Liang and Xiaolin Hu. Recurrent convolutional neural network for object recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3367–3375, 2015.

Chin-Teng Lin, Ruei-Cheng Wu, Sheng-Fu Liang, Wen-Hung Chao, Yu-Jie Chen, and Tzyy-Ping Jung. Eeg-based drowsiness estimation for safety driving using independent component analysis. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 52(12):2726–2738, 2005.

Chin-Teng Lin, Shi-An Chen, Li-Wei Ko, and Yu-Kai Wang. Eeg-based brain dynamics of driving distraction. In *The 2011 international joint conference on neural networks*, pp. 1497–1500. IEEE, 2011.

Min Lin, Qiang Chen, and Shuicheng Yan. Network in network. *CoRR*, abs/1312.4400, 2013. URL `https://api.semanticscholar.org/CorpusID:16636683`.

Xu-Ying Liu, Jianxin Wu, and Zhi-Hua Zhou. Exploratory undersampling for class-imbalance learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 39(2):539–550, 2008.

Yang Liu, Aijun An, and Xiangji Huang. Boosting prediction accuracy on imbalanced datasets with svm ensembles. In *Advances in Knowledge Discovery and Data Mining: 10th Pacific-Asia Conference, PAKDD 2006, Singapore, April 9-12, 2006. Proceedings 10*, pp. 107–118. Springer, 2006.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 10012–10022, 2021.

Mohammed S Majdi, Sundaresh Ram, Jonathan T Gill, and Jeffrey J Rodríguez. Drive-net: Convolutional network for driver distraction detection. In *2018 IEEE Southwest Symposium on Image Analysis and Interpretation (SSIAI)*, pp. 1–4. IEEE, 2018.

Larry Manevitz and Malik Yousef. One-class document classification via neural networks. *Neurocomputing*, 70(7-9):1466–1481, 2007.

Inderjeet Mani and I Zhang. knn approach to unbalanced data distributions: a case study involving information extraction. In *Proceedings of workshop on learning from imbalanced datasets*, volume 126, pp. 1–7. ICML, 2003.

Anqi Mao, Mehryar Mohri, and Yutao Zhong. Cross-entropy loss functions: Theoretical analysis and applications. In *International Conference on Machine Learning*, pp. 23803–23828. PMLR, 2023.

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

David Mease, Abraham J Wyner, and Andreas Buja. Boosted classification trees and class probability/quantile estimation. *Journal of Machine Learning Research*, 8(3), 2007.

Anna Montoya, Dan Holman, SF_data_ science, Taylor Smith, and Wendy Kan. State farm distracted driver detection. https://kaggle.com/competitions/state-farm-distracted-driver-detection, 2016. Accessed: 2024-05-13.

Negar Moslemi, Mohsen Soryani, and Reza Azmi. Computer vision-based recognition of driver distraction: A review. *Concurrency and Computation: Practice and Experience*, 33(24):e6475, 2021.

Antonio Mucherino, Petraq J. Papajorgji, and Panos M. Pardalos. *k-Nearest Neighbor Classification*, pp. 83–106. Springer New York, New York, NY, 2009. ISBN 978-0-387-88615-2. doi: 10.1007/978-0-387-88615-2_4. URL https://doi.org/10.1007/978-0-387-88615-2_4.

Julio Cesar Munguía Mondragón, Eréndira Rendón Lara, Roberto Alejo Eleuterio, Everardo Efrén Granda Gutirrez, and Federico Del Razo López. Density-based clustering to deal with highly imbalanced data in multi-class problems. *Mathematics*, 11(18):4008, 2023.

National Highway Traffic Safety Administration. Distracted driving dangers and statistics. https://www.nhtsa.gov/risky-driving/distracted-driving, 2023. [Accessed 18-04-2024].

Mehdi Noroozi, Ananth Vinjimoor, Paolo Favaro, and Hamed Pirsiavash. Boosting self-supervised learning via knowledge transfer. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 9359–9367, 2018.

Eshed Ohn-Bar and Mohan Manubhai Trivedi. Hand gesture recognition in real time for automotive interfaces: A multimodal vision-based approach and evaluations. *IEEE Transactions on Intelligent Transportation Systems*, 15:2368–2377, 2014. URL https://api.semanticscholar.org/CorpusID:2212467.

Ofonime Dominic Okon and Li Meng. Detecting distracted driving with deep learning. In *Interactive Collaborative Robotics: Second International Conference, ICR 2017, Hatfield, UK, September 12-16, 2017, Proceedings 2*, pp. 170–179. Springer, 2017.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

Keiron O'shea and Ryan Nash. An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458*, 2015.

Rafał Pilarczyk and Władysław Skarbek. On intra-class variance for deep learning of classifiers. *Foundations of Computing and Decision Sciences*, 44(3):285–301, 2019.

Boris T Polyak and Anatoli B Juditsky. Acceleration of stochastic approximation by averaging. *SIAM journal on control and optimization*, 30(4):838–855, 1992.

PyTorch. torchvision.datasets.imagefolder, 2024. URL https://pytorch.org/vision/main/generated/torchvision.datasets.ImageFolder.html. Accessed: 2024-05-15.

PyTorch Contributors. torch.nn.parallel.distributeddataparallel. https://pytorch.org/docs/stable/generated/torch.nn.parallel.DistributedDataParallel.html, 2023a. Accessed: 2024-05-15.

PyTorch Contributors. torch.optim.sgd. `https://pytorch.org/docs/stable/generated/torch.optim.SGD.html`, 2023b. Accessed: 2024-05-15.

PyTorch Contributors. Weightedrandomsampler. `https://pytorch.org/docs/stable/_modules/torch/utils/data/sampler.html#WeightedRandomSampler`, 2023c. [Accessed 14-05-2024].

PyTorch Contributors. vit_b_16. `https://pytorch.org/vision/main/models/generated/torchvision.models.vit_b_16.html`, 2024. [Accessed 14-05-2024].

A Kai Qin and Ponnuthurai N Suganthan. Kernel neural gas algorithms with application to cluster analysis. In *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, volume 4, pp. 617–620. IEEE, 2004.

Binbin Qin, Jiangbo Qian, Yu Xin, Baisong Liu, and Yihong Dong. Distracted driver detection based on a cnn with decreasing filter size. *IEEE transactions on intelligent transportation systems*, 23(7):6922–6933, 2021.

Bhavani Raskutti and Adam Kowalczyk. Extreme re-balancing for svms: a case study. *ACM Sigkdd Explorations Newsletter*, 6(1):60–69, 2004.

Satyendra Singh Rawat and Amit Kumar Mishra. Review of methods for handling class-imbalanced in classification problems, 2022.

Michael A Regan, Charlene Hallett, and Craig P Gordon. Driver distraction and driver inattention: Definition, relationship and taxonomy. *Accident Analysis & Prevention*, 43(5):1771–1781, 2011.

Bryan Reimer, Bruce Mehler, Joseph F Coughlin, Nick Roy, and Jeffery A Dusek. The impact of a naturalistic hands-free cellular phone task on heart rate and simulated driving performance in two age groups. *Transportation research part F: traffic psychology and behaviour*, 14(1):13–25, 2011.

Tal Ridnik, Emanuel Ben-Baruch, Asaf Noy, and Lihi Zelnik-Manor. Imagenet-21k pretraining for the masses. *arXiv preprint arXiv:2104.10972*, 2021.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, pp. 234–241. Springer, 2015.

Yangjun Ruan, Saurabh Singh, Warren Morningstar, Alexander A Alemi, Sergey Ioffe, Ian Fischer, and Joshua V Dillon. Weighted ensemble self-supervised learning. *arXiv preprint arXiv:2211.09981*, 2022.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. doi: 10.1007/s11263-015-0816-y.

Chaitanya Ryali, Yuan-Ting Hu, Daniel Bolya, Chen Wei, Haoqi Fan, Po-Yao Huang, Vaibhav Aggarwal, Arkabandhu Chowdhury, Omid Poursaeed, Judy Hoffman, et al. Hiera: A hierarchical vision transformer without the bells-and-whistles. *arXiv preprint arXiv:2306.00989*, 2023.

Alexandre Sablayrolles, Matthijs Douze, Cordelia Schmid, and Hervé Jégou. Spreading vectors for similarity search. *arXiv preprint arXiv:1806.03198*, 2018.

scikit-learn contrib/hdbscan. Comparing python clustering algorithms. https://hdbscan.readthedocs.io/en/latest/comparing_clustering_algorithms.html, 2024. [Accessed 14-05-2024].

Allianz SE. Drivers are too distracted by modern technology, Mar 2023. URL https://www.allianz.com/en/press/news/studies/230301_Allianz-Drivers-are-too-distracted-by-modern-technology.html.

Amir Shahroudy, Jun Liu, Tian-Tsong Ng, and Gang Wang. NTU RGB+D: A large scale dataset for 3d human activity analysis. *CoRR*, abs/1604.02808, 2016. URL http://arxiv.org/abs/1604.02808.

Arian Shajari, Houshyar Asadi, Sebastien Glaser, Adetokunbo Arogbonlo, Shady Mohamed, Lars Kooijman, Ahmad Abu Alqumsan, and Saeid Nahavandi. Detection of driving distractions and their impacts. *Journal of Advanced Transportation*, 2023:1–17, 09 2023. doi: 10.1155/2023/2118553.

Zhiqiang Shen, Zechun Liu, Jie Qin, Lei Huang, Kwang-Ting Cheng, and Marios Savvides. S2-bnn: Bridging the gap between self-supervised real and 1-bit neural networks via guided distribution calibration. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 2165–2174, 2021.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Joonwoo Son and Myoungouk Park. The effects of distraction type and difficulty on older drivers' performance and behaviour: visual vs. cognitive. *International journal of automotive technology*, 22:97–108, 2021.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1): 1929–1958, 2014.

Yanmin Sun, Mohamed S Kamel, Andrew KC Wong, and Yang Wang. Cost-sensitive boosting for classification of imbalanced data. *Pattern recognition*, 40(12):3358–3378, 2007.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818–2826, 2016.

Aik Choon Tan, David Gilbert, and Yves Deville. Multi-class protein fold classification using a new ensemble machine learning approach. *Genome Informatics*, 14:206–217, 2003.

Yuchun Tang and Yan-Qing Zhang. Granular svm with repetitive undersampling for highly imbalanced protein homology prediction. In *2006 IEEE International Conference on Granular Computing*, pp. 457–460. IEEE, 2006.

IVAN Tomek. Two modifications of cnn. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-6 (11):769–772, 1976. doi: 10.1109/TSMC.1976.4309452.

Ashish Vaswani, Noam M. Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Neural Information Processing Systems*, 2017. URL https://api.semanticscholar.org/CorpusID:13756489.

Fernando Vilariño, Panagiota Spyridonos, Jordi Vitrià, and Petia Radeva. Experiments with svm and stratified sampling with an imbalanced problem: Detection of intestinal contractions. In *International Conference on Pattern Recognition and Image Analysis*, pp. 783–791. Springer, 2005.

Francis Walugembe, Francis Levira, Balasubramanian Ganesh, and Dickson Wilson Lwetoijera. A retrospective study on the epidemiology and trends of road traffic accidents, fatalities and injuries in three municipalities of dar es salaam region, tanzania between 2014-2018. *Pan Afr. Med. J.*, 36:24, May 2020.

Benjamin X Wang and Nathalie Japkowicz. Imbalanced data set learning with synthetic samples. In *Proc. IRIS machine learning workshop*, volume 19, pp. 435, 2004.

Benjamin X Wang and Nathalie Japkowicz. Boosting support vector machines for imbalanced data sets. In *International Symposium on Methodologies for Intelligent Systems*, pp. 38–47. Springer, 2008.

Jiyang Wang, Weiheng Chai, Archana Venkatachalapathy, Kai Liang Tan, Arya Haghighat, Senem Velipasalar, Yaw Adu-Gyamfi, and Anuj Sharma. A survey on driver behavior analysis from in-vehicle cameras. *IEEE Transactions on Intelligent Transportation Systems*, 23(8):10186–10209, 2022. doi: 10.1109/TITS.2021.3126231.

Shoujin Wang, Wei Liu, Jia Wu, Longbing Cao, Qinxue Meng, and Paul J Kennedy. Training deep neural networks on imbalanced data sets. In *2016 international joint conference on neural networks (IJCNN)*, pp. 4368–4374. IEEE, 2016.

Wikipedia contributors. Cosine similarity, 2024a. URL `https://en.wikipedia.org/wiki/Cosine_similarity`. Accessed: 2024-05-15.

Wikipedia contributors. Euclidean distance, 2024b. URL `https://en.wikipedia.org/wiki/Euclidean_distance`. Accessed: 2024-05-15.

World Health Organization. Global status report on road safety 2023. `https://www.who.int/publications/i/item/9789240086517`, December 2023. Accessed: 17-04-2024.

Gang Wu and Edward Y Chang. Class-boundary alignment for imbalanced dataset learning. In *ICML 2003 workshop on learning from imbalanced data sets II, Washington, DC*, pp. 49–56, 2003.

Gang Wu and Edward Y Chang. Aligning boundary in kernel space for learning imbalanced dataset. In *Fourth IEEE International Conference on Data Mining (ICDM'04)*, pp. 265–272. IEEE, 2004.

Gang Wu and Edward Y Chang. Kba: Kernel boundary alignment considering imbalanced data distribution. *IEEE Transactions on knowledge and data engineering*, 17(6):786–795, 2005.

Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V Le. Self-training with noisy student improves imagenet classification. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10687–10698, 2020.

Qiantong Xu, Tatiana Likhomanenko, Jacob Kahn, Awni Hannun, Gabriel Synnaeve, and Ronan Collobert. Iterative pseudo-labeling for speech recognition. *arXiv preprint arXiv:2005.09267*, 2020.

I Zeki Yalniz, Hervé Jégou, Kan Chen, Manohar Paluri, and Dhruv Mahajan. Billion-scale semi-supervised learning for image classification. *arXiv preprint arXiv:1905.00546*, 2019.

Kristie Young, Michael Regan, Mike Hammer, et al. Driver distraction: A review of the literature. *Distracted driving*, 2007:379–405, 2007.

Weichen Yu, Hongyuan Yu, Yan Huang, and Liang Wang. Generalized inter-class loss for gait recognition. In *Proceedings of the 30th ACM International Conference on Multimedia*, pp. 141–150, 2022.

Xiao-Peng Yu and Xiao-Gao Yu. Novel text classification based on k-nearest neighbor. In *2007 International Conference on Machine Learning and Cybernetics*, volume 6, pp. 3425–3430. IEEE, 2007.

Jinhui Yuan, Jianmin Li, and Bo Zhang. Learning concepts from large scale imbalanced data sets using support cluster machines. In *Proceedings of the 14th ACM international conference on Multimedia*, pp. 441–450, 2006.

Bianca Zadrozny, John Langford, and Naoki Abe. Cost-sensitive learning by cost-proportionate example weighting. In *Third IEEE international conference on data mining*, pp. 435–442. IEEE, 2003.

Ben Zhang. Apply and compare different classical image classification method: Detect distracted driver. *Dept. Comput. Sci., Stanford Univ., Stanford, CA, USA, Project Rep*, 229, 2016.

Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part III 14*, pp. 649–666. Springer, 2016.

Richard Zhang, Phillip Isola, and Alexei A Efros. Split-brain autoencoders: Unsupervised learning by cross-channel prediction. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1058–1067, 2017.

Yingzhi Zhang, Taiguo Li, Chao Li, and Xinghong Zhou. A novel driver distraction behavior detection method based on self-supervised learning with masked image modeling. *IEEE Internet of Things Journal*, 2023.

Yulu Zhang, Liguo Shuai, Yali Ren, and Huilin Chen. Image classification with category centers in class imbalance situation. In *2018 33rd youth academic annual conference of Chinese association of automation (YAC)*, pp. 359–363. IEEE, 2018.

Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2921–2929, 2016.

Jinghao Zhou, Chen Wei, Huiyu Wang, Wei Shen, Cihang Xie, Alan Yuille, and Tao Kong. ibot: Image bert pre-training with online tokenizer. *arXiv preprint arXiv:2111.07832*, 2021.

Appendix A

Table 6.1: Hyperparameter Search Experiments conducted using Cosine Annealing Scheduler. Two configurations were chosen for the maximum number of iteration parameter of the CosineAnnealing Scheduler. Total 16 experiments were conducted as shown in the table, which resulted in overfitting.

Experiment Details				Balanced Accuracy		
Exp No	**Optimizer-Scheduler**	**Initial LR**	**T Max**	**Train**	**Validation**	**Gap**
1	Adam - CosineAnnealing	0.01	100	98.12%	75.97%	22.15%
2	Adam - CosineAnnealing	0.03	100	98.38%	73.45%	24.93%
3	Adam - CosineAnnealing	0.003	100	97.73%	79.88%	17.84%
4	Adam - CosineAnnealing	0.06	100	98.56%	72.52%	26.04%
5	SGD - CosineAnnealing	0.01	100	96.62%	80.36%	16.26%
6	SGD - CosineAnnealing	0.03	100	97.32%	79.50%	17.82%
7	SGD - CosineAnnealing	0.003	100	95.21%	81.08%	14.13%
8	SGD - CosineAnnealing	0.06	100	97.73%	78.99%	18.74%
9	Adam - CosineAnnealing	0.01	10	97.78%	77.25%	20.53%
10	Adam - CosineAnnealing	0.03	10	97.95%	68.79%	29.16%
11	Adam - CosineAnnealing	0.003	10	97.66%	76.45%	21.21%
12	Adam - CosineAnnealing	0.06	10	94.25%	67.92%	26.33%
13	SGD - CosineAnnealing	0.01	10	96.54%	80.54%	16.00%
14	SGD - CosineAnnealing	0.03	10	97.33%	78.94%	18.38%
15	SGD - CosineAnnealing	0.003	10	95.21%	81.77%	13.43%
16	SGD - CosineAnnealing	0.06	10	97.53%	75.76%	21.77%

Table 6.2: Hyperparameter Search Experiments conducted using LinearDecay Scheduler. The results shows that only experiment number 22 is showing better results as compare to other experiments. However, there are signs of overfitting in each experiemnt. Experiment 22 showed stable training as compare to other experiments and is overfitting less when compared to all other experiments.

Experiment Details		**Learning Rate**		**Balanced Accuracy**		
Exp No	**Optimizer-Scheduler**	**Initial-End LR**	**Slope**	**Train**	**Validation**	**Gap**
17	SGD - LinearDecay	0.02 - 0.01	-0.0001	97.36%	79.62%	17.74%
18	SGD - LinearDecay	0.03 - 0.02	-0.0001	97.68%	79.88%	17.80%
19	SGD - LinearDecay	0.04 - 0.03	-0.0001	97.82%	78.13%	19.69%
20	SGD - LinearDecay	0.05 - 0.04	-0.0001	97.87%	77.09%	20.78%
21	SGD - LinearDecay	0.07 - 0.06	-0.0001	97.94%	73.03%	24.91%
22	SGD - LinearDecay	0.0004 - 0.0002	-0.000002	92.05%	83.87%	8.18%
23	SGD - LinearDecay	0.0005 - 0.0001	-0.000004	92.00%	83.83%	8.17%
24	Adam - LinearDecay	0.0004 - 0.0002	-0.000002	96.66%	80.86%	15.80%
25	SGD - LinearDecay	0.00045 - 0.00015	-0.0000025	92.62%	83.45%	9.18%

Table 6.3: Hyperparameter Search Experiments conducted using StepDecay Scheduler. Four experiments were conducted using step decay learning rate scheduler, each of which showed overfitting and unstability in training.

Experiment Details			**Balanced Accuracy**		
Exp No	**Optimizer-Scheduler**	**Initial LR**	**Train**	**Validation**	**Gap**
26	SGD - StepDecay/20	0.01	95.68%	81.36%	14.32%
27	SGD - StepDecay/20	0.03	96.81%	80.24%	16.57%
28	SGD - StepDecay/20	0.003	93.94%	81.72%	12.21%
29	SGD - StepDecay/20	0.06	97.24%	79.77%	17.48%

Table 6.4: Hyperparameter Search Experiments conducted using ExponentialDecay Scheduler. This table contains the four experiments conducted using exponential decay learning rate scheduler each of which is showing large gap between train and validation balance accuracy indicating overfitting.

Experiment Details			**Balanced Accuracy**		
Exp No	**Optimizer-Scheduler**	**Initial LR**	**Train**	**Validation**	**Gap**
30	SGD - ExponentialDecay	0.01	96.32%	80.33%	15.99%
31	SGD - ExponentialDecay	0.03	97.39%	78.90%	18.49%
32	SGD - ExponentialDecay	0.003	94.95%	81.08%	13.87%
33	SGD - ExponentialDecay	0.06	97.75%	78.51%	19.24%

Table 6.5: Hyperparameter Search Experiments conducted using ConstantLR Scheduler. The table shows two experiments each of which is showing more than 10% gap between train and validation balanced accuracy indicating overfitting.

Experiment Details			**Balanced Accuracy**		
Exp No	**Optimizer-Scheduler**	**Initial LR**	**Train**	**Validation**	**Gap**
34	SGD - ConstantLR	0.001	94.76%	81.07%	13.70%
35	SGD - ConstantLR	0.003	96.03%	81.39%	14.64%