

**Evaluating the ConvNeXt-ViViT Hybrid Architecture in Predicting  
Vehicular Crashes from Video Data**

Submitted April 2024, in partial fulfilment of  
the conditions for the award of the degree **Computer Science with Year in  
Industry BSc Hons.**

**Justinas Naumenka**  
School of Computer Science  
University of Nottingham

I hereby declare that this dissertation is all my own work, except as indicated in  
the text:

**Signature: J.N.**

**Date 29-04-2024**

I hereby declare that I have all necessary rights and consents to publicly  
distribute this dissertation via the University of Nottingham's e-dissertation  
archive.

## **ABSTRACT**

This dissertation draws inspiration from existing accident anticipation and prevention systems implemented in vehicles. The research is driven by the aspiration to develop something of real-world utility that could aid others in their research or improve existing technologies. The dissertation aims to develop an advanced predictive model that utilises both ConvNeXt and ViViT architectures to accurately detect vehicular crashes from dashcam footage, advancing road safety research. The methodology involved the integration of the ConvNeXt and ViViT models to capture spatial and temporal features from video data. Data preprocessing included resizing, normalisation and augmentation to manage class imbalances and enhance training. Training incorporated strategies like layer freezing, early stopping and pooling to optimise learning. Adam optimiser and Binary Cross-Entropy Loss were utilised to finetune model adjustments during training. Validation was conducted through a subset of the Dashcam Accident Dataset to monitor performance and adjust parameters dynamically. The proposed ConvNeXt-ViViT hybrid architecture achieves state-of-the-art 72.80% AP and 2.03 sec TTA when maximising AP for the DAD dataset. The architecture demonstrates its ability to capture spatial and temporal dynamics from its individual models and suggests that optimising ViViT temporal feature extraction can lead to improvements in TTA.

## **Keywords**

ConvNeXt, ViViT, vehicular crash prediction, deep learning, video processing, spatial-temporal analysis.

## **ACKNOWLEDGEMENTS**

My sincere thanks to Professor Valerio Giuffrida for his mentorship and expertise. I also express my deepest gratitude to my family for their continuous support and encouragement during my studies.

# TABLE OF CONTENTS

1	Introduction.....	5
1.1	Background and Motivation .....	5
1.2	Aims and Objectives .....	5
1.3	Seventeen United Nations Sustainable Development Goals .....	6
1.4	LSEPI.....	6
1.4.1	Intellectual Property .....	6
1.4.2	Research Ethics & Data Protection.....	7
1.4.3	Broader Ethical and Social Considerations.....	7
1.5	Definitions.....	7
1.5.1	Spatial-Temporal Modelling .....	7
1.5.2	RNN/LSTM .....	7
1.5.3	ConvNeXt .....	8
1.5.4	Transformers .....	8
1.5.5	Visual Transformers .....	8
1.5.6	Video Vision Transformers .....	9
1.5.7	Multi-Task Learning .....	9
1.5.8	Supervised Fine-Tuning.....	9
1.5.9	Hybrid CNN-ViT over End-to-End ViViT.....	10
1.5.10	Alternative Approaches .....	10
2	Related Works .....	11
2.1	Early Developments in Accident Anticipation.....	11
2.2	Integration of Spatial-Temporal Dynamics.....	11
2.3	Enhancements Through Adaptive Learning and Uncertainty Modelling .....	11
2.4	Creation of Diverse Datasets .....	12
2.5	Adoption of Advanced Architectures .....	12
2.6	Summary .....	12
3	Design, Methodology and Implementation.....	13
3.1	Architecture Diagram.....	13
3.2	Dataset.....	14
3.3	Data Selection .....	14
3.4	Model.....	15
3.4.1	Data Pre-Processing .....	15

3.4.2 ConvNeXt .....	16
3.4.3 ViViT.....	17
3.4.4 Layer Freezing .....	17
3.4.5 Pooling .....	18
3.4.6 MLP .....	18
3.4.7 Early Stopping & Validation .....	19
3.4.8 Cluster Training .....	19
3.4.9 Criterion .....	20
3.4.10 Optimiser.....	20
3.5 Fine-Tuning.....	20
3.6 Early Models .....	21
4 Results.....	24
4.1 Detail of implementation .....	24
4.2 Evaluation metrics .....	25
4.3 Evaluation .....	26
5 Discussion .....	27
6 Conclusion .....	29
6.1 Study Limitations.....	29
6.2 Future Work .....	29
6.3 Reflection.....	30
6.3.1 Project Management .....	30
6.3.2 Data Management .....	31
6.3.3 Self-Reflection .....	32
7 Bibliography .....	33

# 1 INTRODUCTION

## 1.1 Background and Motivation

With the advancement of AI and Computer Vision technologies, there has been a progressive development in self-driving cars. Anticipation of road incidents for taking preventative measures is an important aspect. Based on the 10-year average from 2013 to 2022, someone is seriously injured or killed on UK roads every 16 minutes [1]. It is an important task as increasing the anticipation time and anticipating accidents with good precision can prevent injuries and loss of life.

This project is driven by a motivation to bridge the gap between high-performing models in controlled datasets and the unpredictable nature of real-world driving scenarios. By tapping into the strengths of recent deep learning architectures and the diversity of dashcam datasets, the goal is to develop an algorithm that understands, anticipates and aids road safety.

Traditional self-driving approaches use Vision and LiDAR for the perception of their surroundings. Vision is important as it can differentiate between objects of similar definitions, whereas LiDAR is excellent at detecting objects and their distance. However, LiDAR can only detect the presence of an object and is unable to clearly determine its significance (e.g., if an object on the road is a paper bag or a tire [5]). While the Vision approach emulates a more desired human-like perception, this can initially introduce complexities in distinguishing objects. However, in the long run, it offers the advantage of understanding the wider context in which these objects interact, delivering a more nuanced perspective than what LiDAR can provide. This is crucial for anticipating accidents as understanding the context and interactions between objects can lead to more accurate predictions of potential hazards. Furthermore, Vision in the form of dashcam footage, creates a low barrier to entry for recreating state-of-the-art technology, as even manufacturers such as Tesla still rely on eight cameras for their sensor coverage [12]. By training a model on widely available and diverse dashboard camera footage, accident anticipation proof of concept can be recreated as shown by existing examples in Table I and their respective performance in Table III.

I will design a model using a hybrid ConvNeXt [21] model which implements spatial feature extraction operations yet uses CNN approach and outperforms many transformer-based alternatives. Combined with the Video Vision Transformer (ViViT) [22] architecture, for handling long-term temporal dependencies well. A proposed optimal solution is the integration of these architectures into a hybrid ConvNeXt-ViViT model, focusing on capturing both short-term and long-term temporal dependencies in dashcam footage.

## 1.2 Aims and Objectives

Design, implement, and evaluate a hybrid accident anticipation model for dashcam footage using cutting edge deep learning architectures such as the Video Vision Transformer and ConvNeXt.

## Objectives:

- Review the current state-of-the-art models in accident anticipation, particularly focusing on those trained on the DAD and CCD.
- Implement a spatial feature extraction model using ConvNeXt and a temporal analysis model using ViViT, focusing on a functional hybrid integration for vehicle crash anticipation.
- Train and fine-tune the hybrid model on the DAD dataset, benchmarking against existing models for AP maximisation.
- Produce an analysis of the hybrid model's performance, showcasing its strengths, flaws and areas for future research and improvements in the field of accident anticipation.

## 1.3 Seventeen United Nations Sustainable Development Goals

**Goal 3** *“Ensure healthy lives and promote well-being for all at all ages”.*

Target 3.6 states “by 2020, halve the number of global deaths and injuries from road traffic accidents” [24]. While this goal is past the due date, the project directly contributes to the cause by providing a tool and research that could potentially prevent road accidents, thereby reducing injuries and saving lives.

**Goal 9** *“Build resilient infrastructure, promote inclusive and sustainable industrialization and foster innovation”.* [25]

This target is addressed by improving road safety therefore contributing to maintaining infrastructure quality from reduced accidents and improving the quality of transport. Technological research in AI also works towards furthering innovation which is a key aspect of this goal.

**Goal 11** *“Make cities and human settlements inclusive, safe, resilient and sustainable”.*

Target 11.2 states “By 2030, provide access to safe, affordable, accessible and sustainable transport systems for all, improving road safety...” [26]. By reducing traffic accidents, the project contributes to making cities and human settlements inclusive, safe, resilient and sustainable. Integrating AI technologies into cities can increase public safety and living standards.

## 1.4 LSEPI

### 1.4.1 Intellectual Property

The work aims to create intellectual property in the form of a functional, tested and evaluated ConvNeXt-ViViT hybrid model architecture for predicting vehicle crashes from video data. The work is based on existing and credited models as well as publicly sourced and referenced datasets. The research is not sponsored or funded by anyone and therefore as per “Provision and Processing of Intellectual Property Rights for Students and Graduates at The University of Nottingham” the undergraduate or postgraduate student on a teaching course will own their own IP.

### **1.4.2 Research Ethics & Data Protection**

The work does not involve human participants or data subjects. No data is collected or personal data is used. As covered in the research ethics, the data is publicly sourced and freely available on the internet.

### **1.4.3 Broader Ethical and Social Considerations**

The overall aim of this work is to provide research on the performance difference between the proposed hybrid model architecture and previous alternatives for vehicle crash prediction and anticipation. The ethical implications, should the work provide state-of-the-art results, would imply that the work shows potential for adaptation towards real-world application for preventing vehicle accidents. With the hope that such model gets adapted for real-world self-driving applications, it would potentially save lives and decrease injuries. As per any model, it can be trained towards whatever data it receives, therefore other potential camera vision implementations such as criminal card counting or military improvised explosive device detection estimation could be implemented. However, this research is exclusively focused on dashcam traffic data benchmarks and aims to only propose performance analysis to the data.

## **1.5 Definitions**

### **1.5.1 Spatial-Temporal Modelling**

Spatial-Temporal modelling analyses both spatial (space-related) and temporal (time-related) aspects of video data and is important in understanding and predicting vehicular crashes from dashcam footage [9]. Spatial analysis focuses on understanding the visual components of each frame, such as vehicle positions, road layout and pedestrian movement. Temporal analysis tracks these elements over time to identify patterns, behaviours and potential hazards that evolve as the situation changes. A typical hybrid approach, combining Convolutional neural Networks (CNNs) such as ConvNeXt and Transformers such as ViViTs, uses the strengths of both models for effective spatial-temporal modelling. CNNs first process the video frames for feature extraction. These features are then fed into a ViViT model, which analyses the sequence of frames as a whole.

### **1.5.2 RNN/LSTM**

Recurrent Neural Networks (RNNs) play an important role in the analysis of sequential and temporal data, a key aspect of predicting vehicular crashes from video data. Unlike CNNs, which excel in spatial feature extraction, RNNs are adept at interpreting temporal dynamics and sequential patterns within data [14]. In crash prediction models, RNNs are often employed to process the temporal aspects of video data following initial spatial analysis by CNNs. However traditional RNNs face challenges such as difficulty in handling long-term dependencies due to issues like vanishing gradients. To address this, variants such as Long Short-Term Memory (LSTM) networks or Gated Recurrent Units (GRUs) are often utilised, offering improved performance in capturing long-range temporal relationships [14]. Prior to transformers, variations on RNN and LSTM were commonly used in vehicle crash anticipation architectures [3,9].

### 1.5.3 ConvNeXt

ConvNeXt is a modern convolutional neural network architecture that adapts traditional CNN designs in purple to compete against Vision Transformers in orange as shown in Figure 1.1 [21]. Each bubble’s area is proportional to FLOPs of a variant in a model family [21]. However, it keeps the core approach of CNNs for processing images but makes several modernisations, outperforming some hierarchical vision transformers across multiple benchmarks. ConvNeXt achieves this by fine-tuning the convolutional layers, improving their efficiency and effectiveness in spatial feature extraction [21]. This results in a pure CNN model that not only matches but, in some cases, surpasses the performance of Vision Transformers, meanwhile maintaining the simplicity and efficiency of CNNs. The research shows that many of ConvNeXt’s design choices while not new, have not been widely explored in the past, resulting in excellent performance [21].

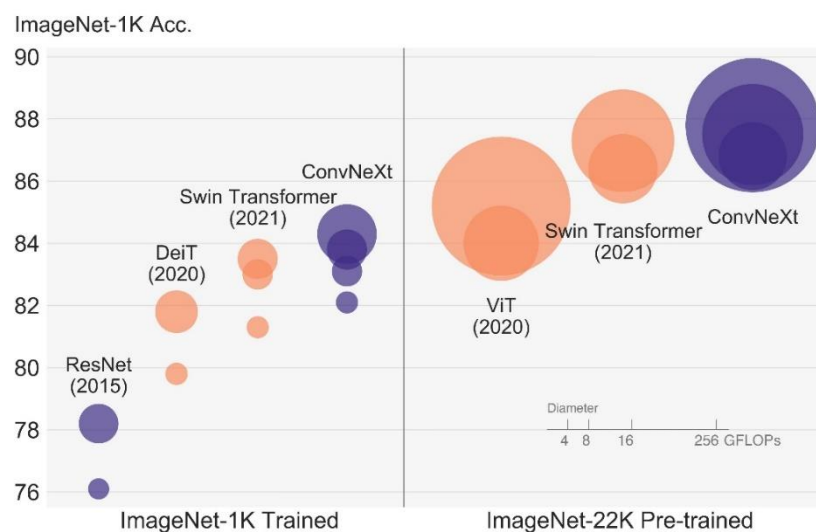


Figure 1.1: ConvNeXt performance compared to alternative models. Source: A ConvNet for the 2020s [21]

### 1.5.4 Transformers

Transformers do not process data in a sequential manner like RNNs and CNNs, but instead use self-attention mechanisms to weight the importance of different parts of input data, regardless of their position [11]. The key strength of transformers is their ability to handle long-range dependencies with greater efficiency compared to traditional RNNs. This makes them well suited for analysing long sequences of data where critical events may be separated by significant time gaps.

### 1.5.5 Visual Transformers

Visual Transformers (ViTs) are a leading development in the field of computer vision, as they adapt the Transformer architecture, traditionally used in natural language processing, for image and video analysis. ViTs handle visual data by dividing images or video frames into a sequence and applying the self-attention mechanism to understand the global context of the scene [13]. ViTs offer a great approach for processing visual data in the context of vehicular crash prediction as the global perspective is beneficial in traffic scenarios. For example, where understanding the broader context of the environment such as the positions and



movements of multiple vehicles and pedestrians is important. Their ability in capturing long-range dependencies within visual data also makes them a powerful tool for analysing temporal video sequences [13].

### 1.5.6 Video Vision Transformers

A Video Vision Transformer (ViViT) is a model specifically designed for video processing. It offers the understanding of both spatial and temporal dynamics in video data. ViViT applies ViT architecture shown in Figure 1.2, to sequences of image frames, enabling the capture of complex temporal relationships within video data [22]. This makes it suitable for tasks like vehicular crash prediction, where understanding small changes over time is crucial.

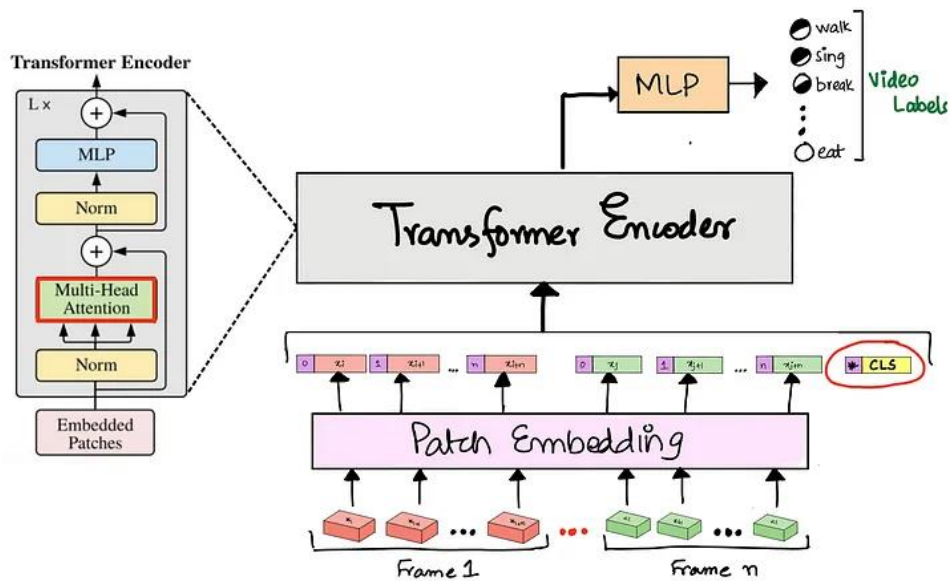


Figure 1.2: ViViT model architecture. Source: A ViViT Video Vision Transformer [23]

### 1.5.7 Multi-Task Learning

Multi-task learning in machine learning means training the model not only on the primary task of predicting vehicular crashes from video data but also on secondary tasks that are relevant e.g., time until impact, weather conditions, time of the day [15]. By training the model to recognise these additional elements the model can develop a deeper understanding of various components in the video, leading to more accurate and reliable predictions.

### 1.5.8 Supervised Fine-Tuning

A combination of pre-trained models will be used for both spatial and temporal aspects of a hybrid CNN-ViViT approach. While a self-supervised approach could be used on abundant non-labelled data, that is car crash footage sourced from YouTube etc. It would be computationally expensive to train making it infeasible and removing the possibility of using a variety of CNNs for feature extraction as a point of comparison. After a model has been pre-trained in a supervised or a self-supervised manner, it often undergoes a fine-tuning phase. During fine-tuning, the model is going to be trained on a smaller, labelled dataset specific to the task at hand [16]. The goal is for the labelled data to adjust the model in a supervised manner for a specific task. Since labelled data can be hard and expensive to

obtain, using a pre-trained model with fine-tuning allows for the maximum use of available labelled data [16].

### **1.5.9 Hybrid CNN-ViT over End-to-End ViViT**

An end-to-end Video Vision Transformer model is an approach where the ViViT is responsible for both spatial and temporal analysis, therefore the entire process of analysing video data is done solely by the ViViT. The key difference between an end-to-end ViViT model and a hybrid CNN-ViViT approach is how they handle the spatial aspects of the video. The end-to-end approach treats spatial analysis as part of its overall sequence processing, applying the self-attention mechanism to spatial features. Whereas a hybrid approach uses CNNs for detailed spatial feature extraction and passes the data to the ViViT for temporal analysis. As the end-to-end model processes data in a unified manner, it ends up requiring the model to process a wide range of tasks potentially leading to decreased accuracy. The targeted and interchangeable nature of CNN feature extraction means that several pre-trained CNN models could be finetuned and used for comparison.

### **1.5.10 Alternative Approaches**

The availability and quality of pre-trained models play a crucial role in model selection, especially in complex tasks like video-based crash prediction. ViViTs have gained a lot of popularity leading to a wide range of high quality pre-trained models. However, the majority of such video pre-trained models have been trained on limited human action-based datasets [4], limiting them in developing a robust understanding of various visual features, which is especially important in tasks like complex traffic analysis. Successor models such as the Swin Transformer V2 [17] might offer architectural improvements, however the lack of available, high quality pre-trained models specifically for video classification limits the usability of the model. Pre-training such models requires significant computational resources and expertise, making ViViTs a more realistic current approach.

Additionally, newer architectures such as the RWKV [18] and Retentive networks [19], while promising, do not seem to have an adaptation for processing video data. In contrast ViViTs have been extensively applied and documented in context of video analysis. Since no mainstream video processing application or pre-trained model exists for either of the promising models, it falls outside the scope of this dissertation.

## **2 RELATED WORKS**

The development of autonomous driving is closely lined to advancements in road safety technology. A crucial aspect of these systems is their ability to prevent accidents by anticipating them before they occur. This literature review examines how dashcam videos, combined with artificial intelligence and machine learning have contributed to this field. As the demand for safer roads increases, it is essential to understand how technologies for predicting accidents have evolved. This understanding will help build ongoing research and guide future directions in accident anticipation.

### **2.1 Early Developments in Accident Anticipation**

The initial focus in accident anticipation was using standard computer vision techniques to analyse dashcam footage for immediate hazard identification. Foundational work by (Chan et al., 2017) is an important reference point as they developed a model that applies Dynamic-Spatial-Attention (DSA) mechanisms within a Recurrent Neural Network (RNN) to process dashcam video data [3]. That is the DSA dynamically allocates attention to various objects detected in the video frames. This enables the model to focus on the most relevant parts of each frame that might contribute to a potential accident [3]. Their model's ability to dynamically shift focus across different frames for accident prediction set a new standard for subsequent research [8, 9, 10]. However, the proposed early architecture often failed to cope with the unpredictable and varied nature of real-world driving conditions [3]. This limitation highlighted the need for models that could learn from complex and diverse data to predict uncommon and varied, accident scenarios reliably.

### **2.2 Integration of Spatial-Temporal Dynamics**

Recognising the limitations of models focused only on temporal features, researchers began to explore the integration of spatial data. A major breakthrough was made by (Karim et al., 2022) who developed a Dynamic Spatial-Temporal Attention Network. This model represents an evolution from earlier architectures by using both spatial and temporal data from dashcam videos. It focuses on the dynamics of space and time to better understand the pre-conditions leading up to accidents [8]. By recognising and analysing subtle cues within the traffic environment over time, the model improves its predictions. This shift towards hybrid models handling complex spatial-temporal relationships produced state-of-the-art results and marked a significant advance in the field.

### **2.3 Enhancements Through Adaptive Learning and Uncertainty Modelling**

Recent advances in accident anticipation have seen the introduction of adaptive learning and uncertainty modelling. (Kataoka et al., 2018) significantly improved the predictive capability of their model by incorporating an Adaptive Loss for Early Anticipation (AdaLEA). This method adjusts the learning focus based on how close the frames are to an accident, which helps the model prioritise critical moments in video data [10]. This adaptive evaluating not only refines the learning process but also tailors the model's response to the complexity of real-world driving data, substantially improving its predictive accuracy.

Furthermore, the introduction of uncertainty modelling by (Bao et al., 2020) addresses a critical gap in earlier models. It combines graph convolutional networks (GCNs), recurrent networks (RNNs) and Bayesian neural networks to improve the prediction of traffic accidents from dashcam videos [9]. By accounting for the inherent uncertainty in traffic scenarios, their model not only better predicts potential accidents but also quantifies the reliability of these predictions [9]. This advancement is important for developing robust anticipation models that can operate reliably in unpredictable environments, providing a framework that further enhances the safety and reliability of crash anticipation.

## **2.4 Creation of Diverse Datasets**

As part of ongoing research, datasets such as the Car Crash Dataset (CCD) and the Dashcam Accident Dataset (DAD) were created [9,3]. Containing a wide range of dash cam video footage in different environments, CCD assists research in the field of traffic accident analysis [9]. Another especially important aspect of existing research is the creation and use of the DAD dataset. This dataset consists of 678 diverse dashcam accident videos [3], providing an even more effective way of evaluating models, due to its challenging and complex real-world driving scenarios.

## **2.5 Adoption of Advanced Architectures**

The field of accident anticipation has recently begun adopting more sophisticated neural network architectures, particularly transformers, which have shown great promise in other areas like natural language processing. The study by (Hajri and Fradi, 2022) represents an alternative technological approach, utilising DenseNet and Vision Transformers (ViTs) to analyse dashcam footage [20]. This approach takes advantage of the transformer's ability to handle complex data patterns. The implementation of transformers in accident anticipation demonstrates the potential of these advanced models to transform the field, providing deeper insights and quicker runtimes from the parallelisable attention-based transformer architecture.

## **2.6 Summary**

This review has tracked the evolution of accident anticipation from basic to sophisticated models incorporating spatial-temporal analysis, adaptive learning and advanced deep learning architectures. The ongoing adoption and improvement of machine learning reflects the increasing capabilities of AI in tackling complex real-world data and improving road safety.

Future research could focus on experimenting with alternative and improved models as proposed in this research. Furthermore, developing real-time accident anticipation capabilities that utilise streaming video data could provide quicker or more accurate predictions. Additionally, integrating multi-model data sources such as LiDAR or radar could further improve the performance and robustness of said models. As AI technology continues to advance, the integration of these sophisticated models into everyday vehicles will likely become more common, helping to prevent accidents and save lives.

### 3 DESIGN, METHODOLOGY AND IMPLEMENTATION

This section outlines the approach taken to develop a hybrid predictive model using ConvNeXt and ViViT architectures for prediction of vehicular crashes from video data.

The process begins with data preprocessing, which includes video frame extraction, colour space conversion, normalisation and data augmentation techniques such as horizontal flipping and equal class distribution to address class imbalance.

The model architecture merges ConvNeXt and ViViT models by removing their classifiers and training an external MLP head to concatenate and process the extracted features that remain. Each dataset input has ConvNeXt processing individual frames for spatial details, while ViViT analyses frame sequences for temporal dynamics as shown in Figure 3.1. This is enhanced by attention pooling in ViViT and adaptive pooling in ConvNeXt and further fine-tuning such as layer freezing, dropout, early stopping and learning rate reduction on plateau.

#### 3.1 Architecture Diagram

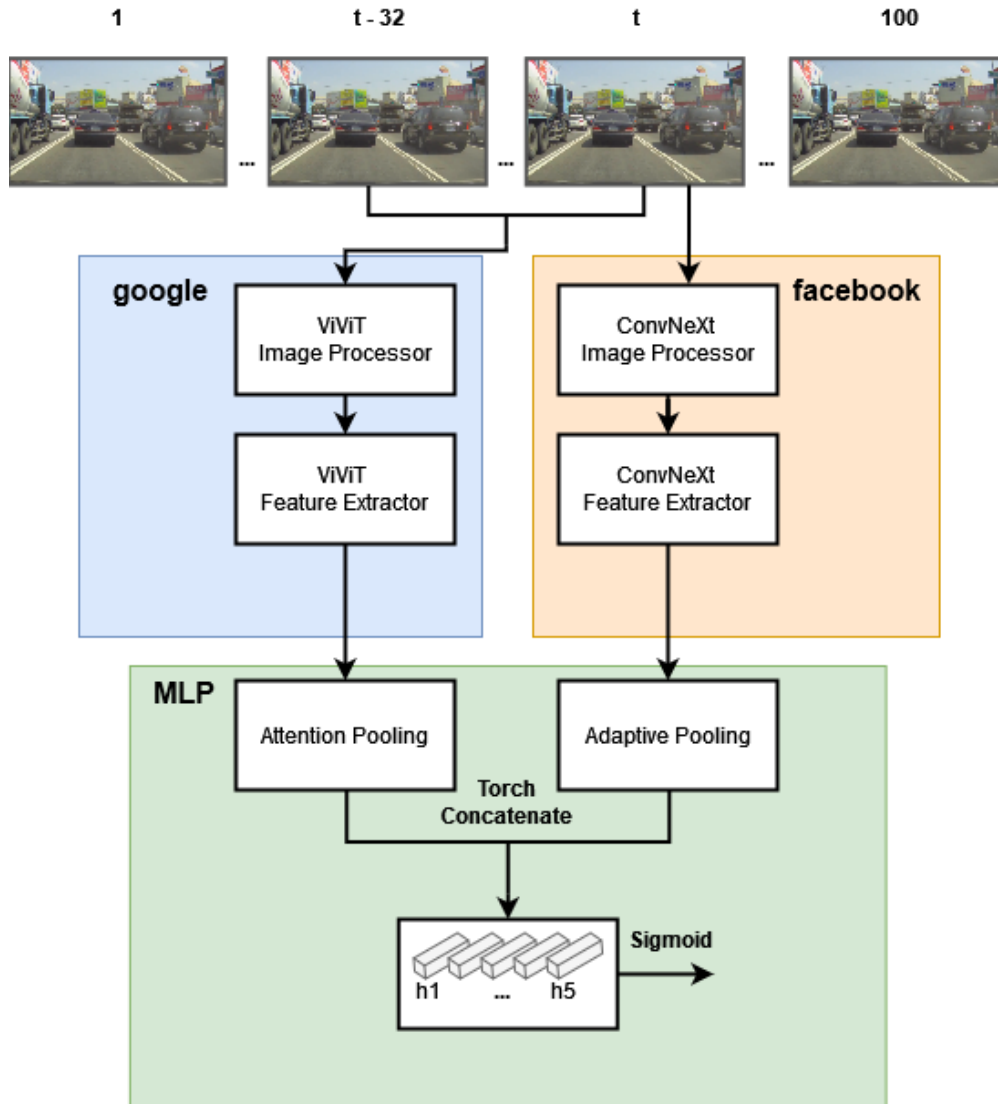


Figure 3.1: Proposed hybrid ConvNeXt-ViViT architecture.

### 3.2 Dataset

As the roads get busier and unpredictable events become more common, drawing insights from datasets like the Dashcam Accident Dataset (DAD) [6], which offer a plethora of real-world, challenging scenarios, becomes crucial. This dataset contains a complicated road scene, crowded streets and diverse accidents where 42.6% of the time motorbike hits car, 19.7% car hits car, 15.6% motorbike hits motorbike and 20% other type. [6]. Each video lasts 5 seconds and consists of 100 frames with the frame rate at 20 frames per second, with 1130 normal driving and 620 crash accident videos. Other existing datasets such as the CarCrashDataset (CCD) [7], have already achieved superior performance. The videos are based in mostly generic driving environments with variations mostly focusing on weather conditions such as Normal, Snowy or Rainy, as well as Day and Night [7]. With the world moving towards urbanization, driving environments are becoming more complex and with 7.2% of severe or fatal accidents by car road users and 4.4% by motorcycles [1], it is important to select a representative dataset. It's not just about identifying a potential collision but understanding the activities leading up to it. Therefore, DAD consists of a wide set of videos captured in the busy streets of Taiwan and is overall more challenging than other datasets like CCD.

### 3.3 Data Selection

The dataset distribution is carefully structured to address class imbalance, which improves the model's accuracy. This balance is crucial for preventing bias towards any particular class, which is often a challenge in datasets where one class outnumbers another.

The dataset is divided into training, validation and testing sets, each with an equal number of positive and negative examples listed in Table I. This even distribution ensures that the model is trained, validated and tested on uniform data samples, removing any preference that could occur from unequal data representation. Such a strategy is vital for developing a model that performs well across various scenarios and reflects performance in real-world settings.

Initially, the dataset included a large number of negative examples, which also consisted of irrelevant videos that did not include driving contexts. To refine the dataset and enhance the quality of training, irrelevant videos were identified and removed manually.

A relatively small validation set was chosen due to the long model runtime and limited hardware availability.

Table 1

DAD DATASET VIDEO DISTRIBUTION

Type	Positive examples	Negative examples	Total
Training set	400	400	800
Validation set	55	55	110
Testing set	165	165	330
Total	620	620	1240
Original Dataset	620	1130	1730

### 3.4 Model

#### 3.4.1 Data Pre-Processing

Pre-processing is a crucial step aimed at normalising the input data and preparing it for effective feature extraction. The dataset comprises dashcam videos, which are annotated with temporal information indicating whether an accident occurs. Each frame within these videos is tagged either as an accident or non-accident frame, creating a binary classification task. Loading the video involves:

- **Frame Extraction:** Sequential frames are extracted from the video as the input for analysis.
- **Colour Space Transformation:** Frames are converted from BGR to RGB colour format, aligning with the input requirements of the neural networks.
- **Resizing and Normalisation:** Frames should be resized to a standard dimension and normalised to match the neural network input, this is done using the pretrained image processors of the corresponding Hugging Face models [28,29].

There are 20 data instances to be trained per video. Initial 10 data points are sampled near the end of the video and horizontal flipping is implemented to artificially expand the dataset (resulting in the mentioned 20 data instances) introducing variability in the visual data presented. This ensures that the model is not simply memorising specific frame orientations, but learns to recognise patterns that are invariant to said transformations. It is separated into two inputs for the individual models:

**ViViT Image Processing:** 20 data instances of 32 frames (based on input requirements) are fed into the ViViT image processing pre-trained “google/vivit-b-16x2-kinetics400” [29]. The data instances are sampled from the 59<sup>th</sup> frame as shown in Figure 3.2 and then flipped respectively.

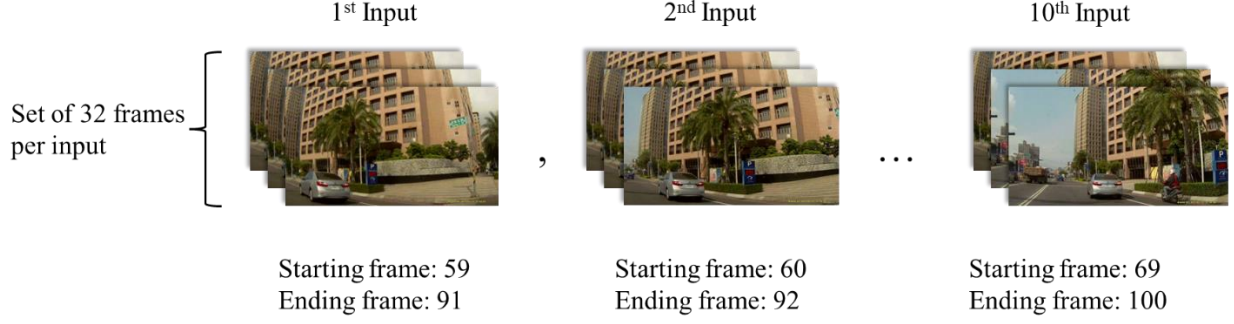


Figure 3.2: Video frames for ViViT input.

ConvNeXt Processing: 20 data instances of 1 frame are fed into the ConvNeXt image processing pre-trained for “facebook/convnext-tiny-224” [28]. The frames in Figure 3.3 are taken as the last frame from the same OpenCV import as ViViT Figure 3.2 to reduce data load time.



Figure 3.3: Video frames for ConvNeXt input.

All of the data inputs will have the same label per video based on the video label. As shown in Figures 3.2 and 3.3, the frames 1-58 are not used when training. This is to prevent a class imbalance since per dataset specification only the last 10 frames contain a crash, which would result in 58 “negative” inputs and 10 “positive” inputs for positively labelled videos. Furthermore, the videos are not labelled uniformly, that is the accident might occur sooner than the last 10 frames. This means that training with frames throughout the entire video would likely be mislabelled and would require additional processing or manual relabelling for the entire dataset.

The PyTorch dataset and data loader classes are used to batch, shuffle and load videos, this is further encased with a PyTorch distributed sampler class to allow parallel processing on multiple GPUs.

### 3.4.2 ConvNeXt

Once the frames are prepared, they are passed through the Hugging Face “facebook/convnext-tiny-224” pretrained ConvNeXt model [28] from the “transformers” python library. This model processes each frame to extract and classify detailed spatial features. The ConvNeXt architecture utilises a series of convolutional layer, each designed to detect different aspects of the input images. From simple edges and textures in the initial layers to more complex patters in the deeper layers, ConvNeXt condenses the frames into a



set of feature maps. These maps represent the crucial visual cues that are necessary for the later parts of the model to assess the likelihood of the crash.

### **3.4.3 ViViT**

In a similar way to ConvNeXt, the batched and processed sets of frames are passed through the Hugging Face “google/vivit-b-16x2-kinetics400” pretrained ViViT model [29] from the “transformers” python library. The output from ViViT consists of a set of classified feature vectors for each set of frames. Unlike ConvNeXt, ViViT processes batches of frames collectively, which allows it to extract and interpret the temporal relationship between subsequent frames. ViViT segments given frames into clips, a further series of consecutive frames and treats each clip as a data point. This segmentation helps the model focus on short and relevant bursts of activity, simplifying the complex problem of video understanding into more manageable parts. Each clip is then passed through a series of transformer blocks that are designed to model interactions not just within a single frame but across the sequence of frames. These interactions help the model learn the contextual relationships necessary to identify patterns indicative of a crash. The use of self-attention mechanisms within the transformer blocks allows the model to weigh the importance of different parts of the video [22], focusing on areas with significant movement or changes that might suggest a crash. By dynamically adjusting its attention, ViViT can more accurately predict crashes by ignoring irrelevant information.

### **3.4.4 Layer Freezing**

Selective layer freezing is implemented as a strategic approach to model training during the fine-tuning of ViViT. This technique is important in stabilising the learning process to improve model performance and accuracy. By freezing early layers, the model avoids catastrophic forgetting [27] that can occur when a pre-trained network is fine-tuned. This is crucial as the target task of crash prediction is different from the generic video classification the model was originally trained for.

Furthermore, freezing layers reduces the number of parameters that need to be updated during training, significantly speeding up the training process. This is especially important given the computational complexity of models like ViViT. One of the hardware limitations for this project was the limited 16gb of ram for each of the 2 available cluster GPUs. By selectively freezing early layers up to the start of the 4<sup>th</sup> ViViT encoder layer the model was not only able to retain pre-trained generalised features, but it also allowed for training on a single GPU within memory constraints reducing complexity and training runtime.

### 3.4.5 Pooling

Pooling techniques reduce the dimensions of the feature maps generated by the respective models. This not only speeds up the processing but also decreases the computational load. Referring to the previously mentioned GPU memory limitation, the end reduction in input dimensionality went from 2446848 MLP input parameters to 1536, that is the sum of 768 output features for each ConvNeXt and ViViT respectively after pooling.

$$[32, 768, 7, 7] \text{ ConvNeXt} + [1, 3137, 768] \text{ ViViT} = \text{MLP input parameters}$$

Condensing the dimensions:

$$37632 + 2409216 = 2446848 \text{ MLP input parameters}$$

#### 3.4.5.1 ViViT Attention Pooling

Attention pooling [31] is used to weigh and then condense the temporal features extracted by the ViViT model across different frames. It involves a learned query mechanism that scores each frame's relevance, allowing the model to focus on the most important parts of the frame sequence.

#### 3.4.5.2 ConvNeXt Adaptive Pooling

Adaptive pooling [32] adjusts the spatial dimensions of the feature map to the specified target size of 1x768 for the MLP head. Like ViViT pooling, by reducing the volume of data, the models can better highlight important features.

The resulting pooled feature provides a summarised representation of the frame/frames, emphasising features that are more relevant to car crash prediction. The reduced overfitting from learning noise and minute details displayed significant improvements in model performance for the given complex task.

### 3.4.6 MLP

Both ConvNeXt and ViViT pre-trained models come with existing classifiers for generic tasks. As the proposed model architecture looks to join both pre-trained models, a further MLP head classifier is created. The main purpose of the proposed MLP classifier is to remove the existing general classifiers that come with the models, concatenate the extracted features from both models and output a classification.

When a forward pass is called, each model is called to process the input, the last hidden layer without classification is taken and passed through their respective pooling. The pooled features are then reshaped and concatenated into a single dimension of 1536 parameters using PyTorch and passed into the first MLP layer.

The input size to the first linear layer of the MLP is twice the size of the individual pooled feature vectors of size totalling 1536 and is proceeded by several hidden layers each of size 1024, 512, 256 and 1 respectively. Each layer is defined by a transformation “nn.Linear” [33] followed by a ReLU activation function “nn.ReLU” [34]. The ReLU activation helps introduce non-linearity into the processing, which helps with complex patterns within the

data. Dropout layers “nn.Dropout” [35] with a probability of 0.1 are added twice at the start of the network to reduce the risk of overfitting. Dropout works by randomly setting a fraction of input units to 0 at each update during training, reinforcing robustness towards unseen data.

The final layer in the MLP is a linear layer that reduces the dimensionality of the output from the last hidden layer to a single value. This value represents the model’s confidence in the presence of a crash. A sigmoid activation function “nn.Sigmoid” is applied to this output to restrain the result between 0 and 1 for interpretation as a probability.

### **3.4.7 Early Stopping & Validation**

Early stopping is a form of regularisation used to avoid overfitting during the training of a model. Model’s performance is monitored on a validation set and the training process stops as soon as the performance starts to deteriorate or stops improving significantly.

At the end of each epoch the loss is evaluated on a validation set using Scikit learn metrics “average\_precision\_score” [36]. After several runs the delta was left at 0, that is all improvements were applied and model weights saved, otherwise the best existing model would be loaded.

The “EarlyStopping” class includes a patience parameter, which is the number of epochs to continue training after the validation loss has stopped improving. This value allows some leeway for the model to overcome potential plateaus in the loss landscape which is a common problem.

Alongside monitoring, the early stopping mechanism involves saving the model at the state where it has achieved the best results (checkpointing). If the training concludes due to early stopping, the model is reverted to this checkpoint which is considered the best performing model for the crash classification.

With checkpointing, training can be interrupted after each epoch and the model then loaded once again to resume training. In such a case, the early stop instance gets called to determine an initial best score, this only occurs if an existing model is detected and no best score exists. However, storing and loading the best score value for training externally is recommended due to saved runtime and was not implemented purely for validation testing purposes.

### **3.4.8 Cluster Training**

PyTorch distributed sampler is used during training to partition the dataset across multiple GPUs, this ensures that each GPU handles a subset of data parallelising the computations and data processing. The Compute Unified Device Architecture (CUDA) from NVIDIA is used to enable GPU training. As development was done locally and on the University of Nottingham cluster, environment variables were set up to enable seamless switching between development environments. This was implemented to allow dynamic GPU number when initialising the model, switching between training and testing modes as well as enabling time-to-accident (TTA) training mode designed to process the entire video in testing. Additionally, models, labels and inputs required management across hardware, moved onto their respective GPU when processing. Likewise, model output and labels need to be moved back onto the

CPU for performance metric calculations which are not traditionally handled by the GPU. Lastly good memory management is crucial, the GPU cache is manually emptied after processing each set of video frame inputs as shown in the dips of Figures 3.4, 3.5 and 3.6. This is done in the hopes of freeing up unused resources to avoid memory overflow common in processing large volumes of data repeatedly.

#### **3.4.9 Criterion**

The criterion or loss function, quantifies the error between the model's predictions and the actual data. In this architecture, a Binary Cross-Entropy Loss (BCELoss) is used [37]. BCELoss measures the distance between the model's probability output for each class and the true distribution. For each prediction, it calculates the loss as a function of the difference between the predicted probability of a crash occurring and the actual label of the video. The formula for BCELoss involves the logarithm of the predicted probabilities [38], which penalises incorrect predictions far from the true label. By penalising predictions that are confident and wrong, BCELoss helps to rapidly correct the model's trajectory during training.

#### **3.4.10 Optimiser**

Adaptive Moment Estimation (Adam) was chosen for updating the model's parameters in response to the output from the criterion [39]. A learning rate of 0.000001 was chosen as appropriate after testing. The Adam optimiser is known for its effectiveness in handling sparse gradients and its adaptive learning rate capabilities which make it suitable for crash prediction. Adam calculates adaptive learning rates for each parameter by estimating the first and second moments of the gradients [40]. This feature helps in navigating along the most effective paths in the loss landscape, leading to a faster and more stable convergence.

Additionally, the training process uses the "ReduceLROnPlateau" scheduler from PyTorch [41] which adjusts the learning rate based on the validation performance. This scheduler monitors validation loss and if no improvement is seen for a patience number of epochs, set at 3 epochs, it will reduce the learning rate by a factor of 0.1. This is useful for fine-tuning the model when it is close to a local minimum in the loss landscape as it allows the model to take smaller steps when necessary. This smaller stepping helps find a more optimal minimum which might be missed with a larger learning rate without overshooting.

### **3.5 Fine-Tuning**

Fine-tuning a hybrid ConvNeXt-ViViT model is a crucial step in adapting the pre-trained neural networks to the specific task of vehicular crash prediction. Both models are defined inside of the MLP class. In this case the models are called in a sequence during the model's forward pass. During training the model's parameters need to be updated, therefore "torch.no\_grad()" is removed to allow for gradient calculations and gets called when running validation and testing. This makes sure that backpropagation updates the weights of models during training. Likewise the models are set between ".eval()" evaluation mode in validation and testing and ".train()" mode during training. This results in lower memory usage and faster performance.

### 3.6 Early Models

Initial development was done using the individual ConvNeXt model as a development benchmark. Given the unfamiliarity with PyTorch and transformer model memory usage that is unusable locally, the majority of model architecture features were initially implemented for the singular ConvNeXt model. This would double as a performance benchmark when testing the average precision of singular models compared to the proposed hybrid architecture.

Data loading was the first initial hurdle as reading entire videos and only using a single frame for ConvNeXt would have to be done several times for a single video to comply with the PyTorch DataLoader [42] “\_\_getitem\_\_” standards necessary for batching. This was overcome with a “load\_frame” custom function which selectively loads specified set or singular frames from a video using opencv. To reduce load times further in the combined implementation, the target frame for ConvNeXt is taken as the last frame from the set of loaded ViViT frames, see Figure 3.1.

Although the frames for a video are limited and only the end sections of videos are used to prevent class imbalance, frame skipping was implemented for the individual ConvNeXt implementation at the interval of 4 frames, as neighbour frames are relatively similar. To combat the reduced training set, horizontal flipping was implemented. In the ConvNeXt implementation horizontal frames were selected as t-2 frames, where t is the target frame. This is done to give more variety to the training data as t-2 frames would not be used given the 4-frame skip interval. Given the sequential nature of ViViT inputs, frame skipping was not used in the combined model, however horizontal flipping remained, flipping every input of a video.

Running the out of the box individual ViViT model is theoretically possible locally without updating its weights, using shared GPU memory on an 8gb 3060ti. This is very time consuming and will not finetune the ViViT model for the specified task. Automatic mixed precision was introduced to reduce memory usage and speed up processing. However, whilst speeding up the training time, the reduced stored accuracy of weights prevented any further improvements for the individual ViViT model after the first epoch. This pattern was also observed when later implemented in the combined model and was respectively removed. Further efforts to reduce memory usage included layer freezing and pooling. With the benefit of reducing catastrophic forgetting [27] layer freezing also significantly reduced the memory usage of the ViViT model when finetuning. To view the model architecture and layers, the model was simply imported and printed as well as iterated through based on its “named\_parameters” function call. Selecting the layers to freeze was done with the binary search methodology, trying to run the model maximising the number of full unfrozen layers whilst staying in the memory limits viewed locally within the task manager. With shared memory the local 3060ti is able to simulate 16gb of memory, which is the same memory as the later used NVIDIA RTX A4000 16gb cluster GPUs. Unfortunately given the time constraints, layer freezing was initially overlooked and later not implemented on the ConvNeXt model and in turn the combined model either. Whilst simple to do, retraining the adjusted combined model would take several days to get multiple results resulting in late

project submission. One of the main memory reducing features was adaptive and attention pooling for ConvNeXt and ViViT respectively. As the output of the ViViT feature extractor is of 2409216 parameters, each one multiplied by the MLP input layer, it presented an issue of capturing all features in the MLP hidden layers whilst maintaining low memory size especially in the initial layers. Initially fearing that condensing the extracted feature layers would result in lost features and decreased accuracy as the added pooling would be done after the noisy image has already been processed, the results were an opposite. Around a 10% average precision improvement was seen by pooling to an input of 768 parameters for individual and hybrid models. An example memory usage for ViViT without and with pooling is demonstrated in Figures 3.4, 3.5 and 3.6 below.

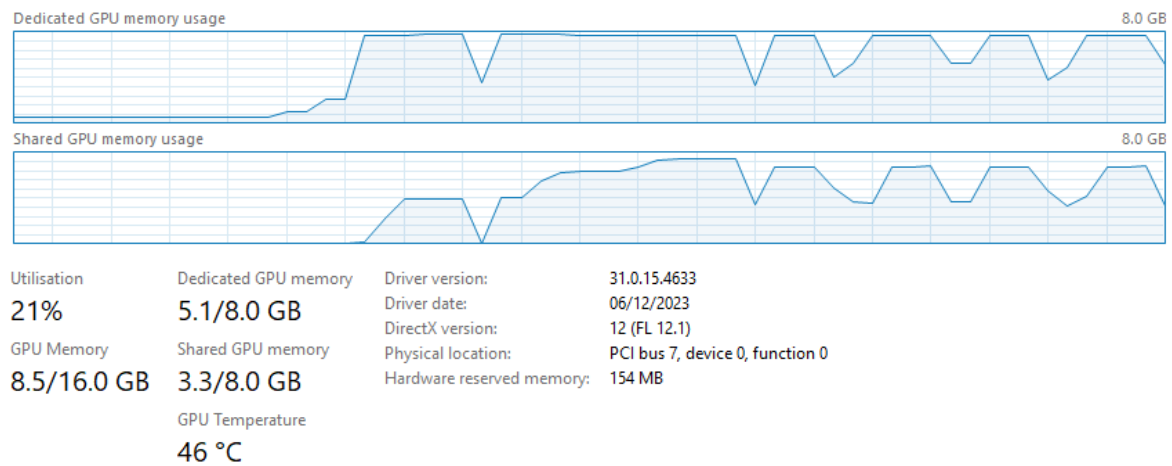


Figure 3.4: ViViT without attention pooling ~15gb memory, hidden layer size of [128,64,32,1].

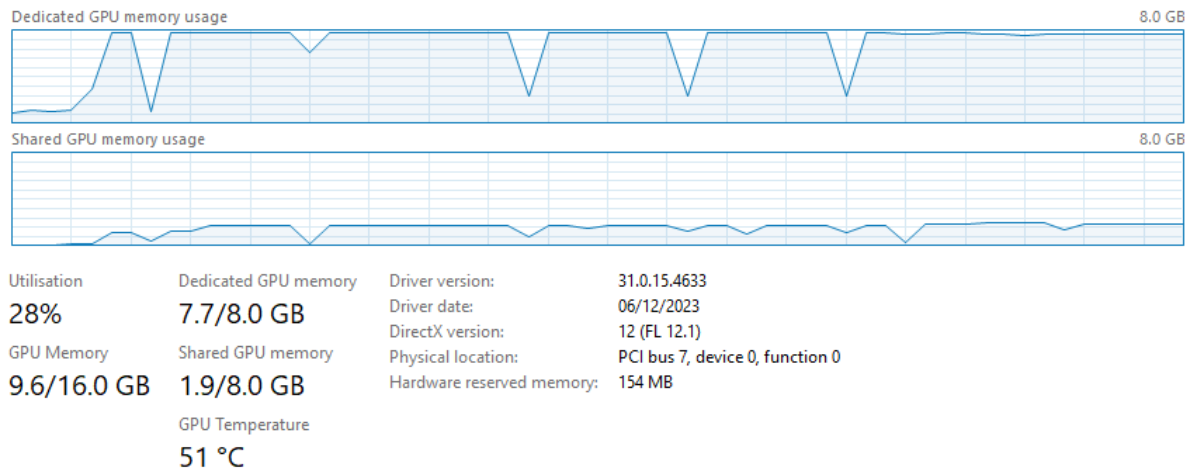


Figure 3.5: ViViT with attention pooling ~9.6gb memory, hidden layer size of [128,64,32,1].

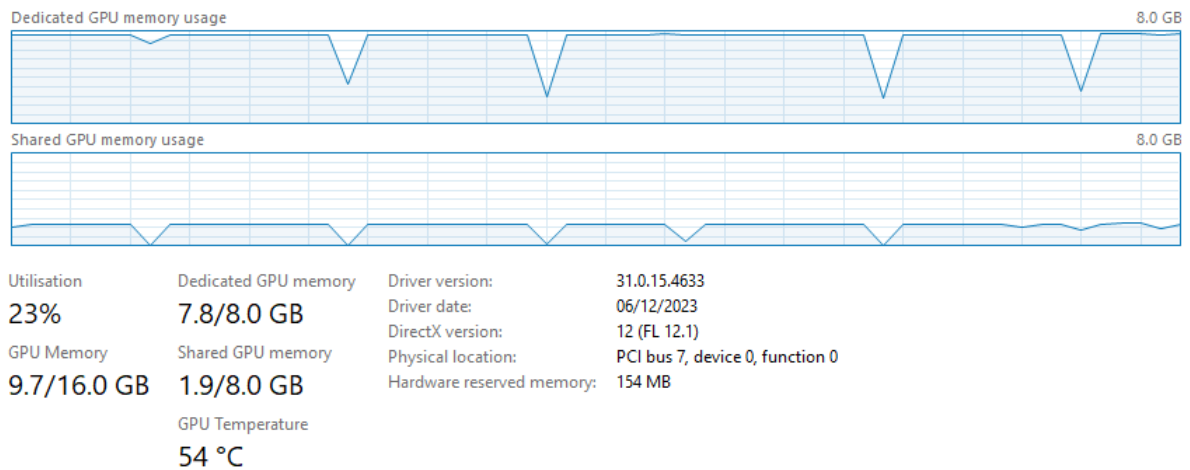


Figure 3.6: ViViT with attention pooling ~9.7gb memory, hidden layer size of [768,512,128,1], no noticeable memory difference.

Joining both models introduced a few restrictions, most important of which was the main frame  $t$ , must be the last frame of ViViT model. This is so that the model can be applied to real-world scenarios simulating the processing of a live dashcam feed without relying future frames. In the architecture this was achieved by feeding exactly the last frame of the data to ConvNeXt. Processing all of the frames in the ViViT temporal sequence using ConvNeXt and processing their sum extracted features alongside ViViT features in the MLP could produce higher accuracy. This however would shift the frame median to the centre of the frame sequence and not the latest “real-time” frame potentially increasing TTA.

Only the latter section of the video is processed, where an accident is likely to occur. If the video is labelled positive, then every frame output of the neural network will be trained against the positive label and vice versa for all developed models. This is not only done to prevent a class imbalance, but it also combats the incorrect assumption of labels that an accident occurs in the last 10 frames, that is in the last 0.5 seconds. Figure 3.7 is only one of many examples. The impact of the car happens around  $\frac{1}{4}$  of the way through the video. As the video is 5 seconds long (the player incorrectly shows total time of 4 seconds instead of 5 for all videos), the impact would occur around 1.25 seconds before the end of the video compared to the proposed 0.5 seconds. That is frame ~75 instead of the proposed 90. Even with consideration that the estimated measurements are inaccurate, this is a frequent occurrence and cannot be ignored.



Figure 3.7: Positive training video no. 200, accident occurs before the last 10 frames (0.5 seconds)

Nevertheless, a testing benchmark was created using the entire video for a comparative score against existing models. This is done by taking frames from  $t = 32$  to  $t=100$  in the combined model testing, as ViViT requires an existing buffer of 32 frames to measure temporal dynamics. As the model is not trained on the entirety of the video, the AP and TTA results are expected to be lower than alternative models.

## 4 RESULTS

This section presents the results of experiments conducted to evaluate the performance of the proposed hybrid ConvNeXt-ViViT model in predicting vehicular crashes. Model training was designed to maximise AP on the DAD dataset due to the inconsistencies found in DAD categorising “the moment of accident at the last 10 frames” [6] used for TTA calculations.

### 4.1 Detail of implementation

The study was implemented using PyTorch with training and testing performed using 2 Nvidia RTX A4000 16gb GPUs. An equal number of positive and negative videos were selected from the DAD dataset for training and testing. Selected frames from videos were assigned a positive or negative crash label based on the entire video’s label rather than the suggested “last 10 frames” [6]. For a single video 10 inputs were chosen to train the model in the step of 1 from the end of the video, aiming to capture the proposed accident frames. A single ViViT input consisted of 32 frames with ConvNeXt input using the last frame from the ViViT’s subset as demonstrated in Figure 3.1. The videos were processed with pre-trained respective model’s image processors. Pooling dimensions of 768 were selected due to the matching [..., 768] dimensionality from both models after feature extraction. Similarly [1024, 512, 256, 1] MLP hidden states were chosen to reasonably match and gradually reduce the dimensions for the total sum of 1536 pooled extracted features. Learning rate of 0.000001 was



chosen to train the network and batch size of 1 was selected due to GPU memory limitations. Similarly to the DSTA implementation [8], ReduceLROnPlateau was used as the learning rate scheduler. Adam optimiser was used with 20 epochs and early stop validation was based on AP with a generous patience of 12.

## 4.2 Evaluation metrics

The evaluation of the proposed ConvNeXt-ViViT model for vehicular crash prediction will be implemented using proposed past methods for correctness and earliness [8] for a standardised benchmark. However, due to inconsistencies found in DAD categorising “the moment of accident at the last 10 frames” [6], the model will be trained to maximise AP.

*Correctness:*

**Recall (R)** – Measures the proportion of actual crashes that the model correctly identifies. A higher recall value would indicate that the model is effective in identifying most crash instances, minimising missed detections.

$$R = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}}$$

**Precision (P)** – The ratio of correctly predicted crash instances compared to all instances predicted as crashes. High precision indicates that the model has fewer false positives,

$$P = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}}$$

**Average precision (AP)** – Provides a single metric to evaluate the model’s performance across different levels of recall. It is useful to keep recall and precision balanced to address the imbalance of positive and negative cases and provide a more accurate measure of performance.

$$AP = \sum_{i=1}^n (R_i - R_{i-1})P_i$$

*Earliness:*

**Time-To-Accident (TTA)** – Evaluates the earliness of accident anticipation based on positive predictions. This is calculated by taking the difference in time  $y - t$  between the frame where the accident occurs  $y$  and is estimated to have occurred  $t$ . Averaging the true positive anticipations will provide the average expected anticipation time for the TTA [6].

### 4.3 Evaluation

Please note that the proposed model was trained on a subset of a video and the achieved TTA was measured by applying the trained model to the entire video and measuring the first frame indicating above 0.5 certainty threshold output. An important metric in Table II is the recall, many papers in use results at 80% recall finding “the most attractive combination of precision and TTA” [8]. However, given the class balanced DAD dataset used in training and testing, recall values cannot be directly compared. Nevertheless, training in AP maximisation has produced state-of-the-art results for AP and TTA at 72.80% and 2.03 seconds respectively. Differences between recall as well as precision in Table II and Table III indicate that on average whilst transformer-based models have slightly higher average precision, introducing complexity into the ViViT model will correspondingly reduce recall.

Table II

COMPARISON OF MODELS MAXIMISING AP ON DAD

Year	Model	AP(%)↑	TTA(s)↑
2016	DSA [3]	56.14	1.86
2018	adaLEA [10]	52.30	3.43
2020	GCRNN [9]	72.22	1.33
2021	DSTA [8]	72.34	1.50
2022	DenseNet-ViT [20]	68.32	-
2024	ConvNeXt-ViViT (Ours)	<b>72.80</b>	<b>2.03</b>

Before the ablation studies shown in Table III, a no dropout ViViT model without pooling achieved AP of 59.59% with 57.34% Precision, 51.97% Recall and an average guess of 0.4526 favouring negative predictions. Given the comparative 11.85% increase in top measured ViViT performance after pooling it was then applied to all later ConvNeXt and Hybrid models showing similar improvements in experimentation. Automatic Mixed Precision (AMP) did not result in noticeable AP changes or memory usage, however any further epochs after the initial results did not yield improvements. Therefore, AMP was deemed incompatible with the proposed architecture and subsequently removed.

Table III

MODEL STUDY ON DAD

Experiment	Model	Dropout	AP (%) ↑	Precision (%) ↑	Recall (%) ↑	Average Guess 0: negative 1: positive
1	ConvNeXt	0	60.12	57	69.09	0.5504
2	ConvNeXt	0.2	62.24	57.55	77.77	0.6145
3	ConvNeXt	0.4	63.80	59.05	67.73	0.5536
4	ViViT	0	71.44	56.50	77.06	0.6787
5	ViViT	0.1	69.00	62.03	77.21	0.6217
6	ConvNeXt-ViViT	0.1	<b>72.80</b>	59.76	68.94	0.5742
7	ConvNeXt-ViViT	0	71.62	61.62	73.36	0.5917
8	ConvNeXt-ViViT	0	68.61	57.66	73.30	0.6351

Given the long testing runtimes of proposed models, only the 2 best performing unique models were selected to compare TTA values in Table IV. A reasonably significant 20 second TTA improvement between ViViT and ConvNeXt-ViViT architecture is observed. Initial hypothesis is made that combined hybrid spatial-temporal extracted features lower the significance of ViViT temporal features that capture important TTA dynamics, resulting in lower TTA predictions. However, due to the extremely limited test size, more testing is needed which would guide areas for improvement.

Table IV

MODEL TTA STUDY ON DAD

Experiment	Model	TTA (s) ↑	AP (%) ↑	Precision (%) ↑	Recall (%) ↑	Average Guess 0: negative 1: positive
1	ViViT	2.21	71.44	56.50	77.06	0.6787
2	ConvNeXt-ViViT	2.03	<b>72.80</b>	59.76	68.94	0.5742

## 5 DISCUSSION

The primary objective of this dissertation was to explore the effectiveness of a hybrid ConvNeXt-ViViT model in predicting vehicular crashes by utilising their respective strengths in spatial and temporal processing. The aim was to measure whether the hybrid integration of ConvNeXt and ViViT could outperform state-of-the-art models by respectively capturing spatal-temporal relationships based on their pre-trained backgrounds. The findings revealed that the hybrid model slightly surpassed the current state-of-the-art maximum accuracy with an AP of 72.80% compared to the 72.34% achieved by the DSTA model [8].

The experimental results in Table III show some variability in model performance, likely influenced by the reduced dataset, limited data and long training time required per model iteration. Specifically, the best performing hybrid model achieved a precision of 59.76% and a recall of 68.94%, with dropout at 0.1 enhancing model generalisation. Individual performances of ConvNeXt and ViViT models with varying dropout levels indicate sensitivity of these models to hyperparameter tuning. Particularly transformer-based models whilst having a slight improvement from small dropout rates, showed drastic fall in AP when using standard starting dropout rates of 0.4 and 0.5 when testing. This can indicate a reliance on temporal coherence across frames, which would be more sensitive to disruption by larger dropout values. Sensitivity to parameters can also be observed with learning rates as 0.000001 with “ReduceLROnPlateau” was used to give consistent results whereas 0.0001 resulted in the model overfitting to consistent guesses of 0 or 1 after the sigmoid function with near 50% AP.

A critical finding was the role of pooling in the model’s architecture. Initially anticipated to reduce accuracy due to the reduction in feature granularity, pooling instead improved the model’s AP by approximately 10% from ViViT development testing without pooling of AP at

59.59%. This improvement can likely be attributed to a more focused feature extraction and a gradual utilisation of relevant features from an increase in initial MLP hidden layers.

However, the study faced obstacles due to the unavailability of tools for visualising the effectiveness of video feature extraction and classification, especially for the ViViT model, at the time of implementation. This further hinders the explanation and reasoning behind the model's decisions which can be dangerous in practical applications as varied and nuanced crash scenarios become unpredictable.

Higher hybrid ConvNeXt and ViViT AP demonstrates the utilisation of individual model strengths. Whilst the individual ViViT temporal model reached similar results, not only the backbone model should be taken into consideration, but the overall importance of temporal analysis as well. Furthermore, the significantly larger transformer model is expected to have a more drastic impact on AP than CNN based ConvNeXt, which can lead to smaller noticeable improvements when joining the models. Nevertheless, an attempt to balance model significance was made in the MLP input layers by pooling the features of both models to an individual extracted total of 768.

The model's adaptability to different road and environment conditions from the DAD dataset demonstrates potential for real-world applications. Whilst TTA, an important and underdeveloped aspect of the model and real-time driving has not been fully utilised due to variance in the DAD real positive labels. State-of-the-art AP and existing works such as TTA maximisation by DSTA [8] demonstrate that adapting the proposed architecture towards TTA has been done and has potential to show similar high-performance results.

Practical model applications remain unexplored and outside of the scope for this dissertation. Nevertheless, transformer applications for real-time tasks such as multi-object tracking [30] and speech recognition [2] are actively developed. Furthermore, the parallel processing capability of transformer models improves hardware utilisation over traditional CNNs. That is why the proposed hybrid model shows substantial promise for deployment in vehicle environments and real-time monitoring systems designed to improve road safety.

The proposed architecture is designed with scalability in mind. Alternative pretrained datasets such as ImageNet-21k or other specialised datasets can be used to train ConvNeXt, likewise ViViT can be trained on a more vehicle motion-oriented dataset or with larger parameter size. Additionally, documented steps during development aim to guide further research or implementations. With provided ways to manage memory usage and improve performance for available hardware, as well as how to avoid some potential setup errors.

## 6 CONCLUSION

### 6.1 Study Limitations

The reliability of results is constrained by the limited number of training runs and the size of the dataset used. With only three runs for the finalised combined model and eight runs for all finalised models, the robustness findings might not be fully explored. Extended training would reduce the variability observed in target metrics of AP, precision and recall.

Extensive computational resources required for training the model highlight research hardware limitations. This restricts the potential for hyperparameter tuning, crucial for optimising model performance.

While a diverse dataset was used, outlier conditions such as obstructed views, lighting, lightning or other diverse occurrences in real-world scenarios were not trained nor tested. This limitation could affect practical deployment of the model for real-world applications.

Real-world implementations of crash anticipation software process inputs from several cameras/ alternative sensors aiming for full coverage of surroundings. This not only helps reduce false positives as several inputs can validate predictions, but also differs from the singular dashcam video input the proposed architecture evaluates.

The performance of the hybrid model heavily relies on the pre-trained states of the ConvNeXt and ViViT models. This raises questions for the evaluation, performance and generalisation metrics since fairly generalised pre-trained datasets were used. Specific driving pre-trained models are not widely available and whilst ViViT pre-trained on dashcam footage could lead to overfitting, it might be desired compared to the current finetuning of Kinetics-400 human action recognition videos for vehicular movement.

A lack of tools to visualise the model's decision making, especially for ViViT attention, limits the ability to interpret and refine the model manually. Better visualisation techniques would allow for targeted improvements as well as a more confident assessment of the spatial-temporal analysis delegation between models.

### 6.2 Future Work

Multi-task learning with DAD provided or custom object detection could refine the model towards incident focused video analysis. This would allow for identification and focus of relevant objects within the video, such as vehicles, people or different types of vehicles which could be processed differently potentially improving accuracy as demonstrated by DSTA [8].

Methods to visually represent attention for both models would help demystify some of the decision making and spatial-temporal processing. Additional analysis should be aimed for when explaining the reasoning behind neural network predictions, a common challenge in deep learning research.

Modifying the loss function to include exponential loss coefficients for frames in positive videos further away from the time of accident, encouraging earlier crash predictions. This

approach is also observed in the DSTA [8] implementation and would be especially useful in real-time applications where TTA has a significant effect on accident prevention.

Alternative optimisers could be explored such as AdaLead, which has shown potential for higher AP in related research [10]. This exploration could lead to more efficient training cycles, potentially better performance or TTA growth.

Integration of additional video or sensor inputs such as radar or LiDAR could enhance model robustness and accuracy as well as focus on development for practical use. Sensor fusion would allow the model to not only rely on visual data, but also alternative dimensions of environmental information.

Alternatively, transformer models designed for spatial feature extraction such as ViT alongside ViViT for temporal analysis could increase AP and TTA. The parallel nature of transformers which would now be used for both spatial and temporal feature extraction could also result in improved performance.

Segmenting the testing dataset into subsets based on features or scenarios could provide more insight into the model's performance across various conditions. This segmentation would enable targeted improvements and a more detailed performance analysis.

Assessment of temporal and spatial feature significance on AP and TTA could help indicate areas for improvement. Furthermore, an analysis of impact that individual models have on performance metrics compared to the temporal and spatial feature significance would help measure the performance difference of alternative feature extraction models.

## **6.3 Reflection**

### **6.3.1 Project Management**

Incremental development approach was used, focusing on deliverables discussed with the project supervisor during consistent weekly in person meetings. Meetings would be planned and organised in advance mostly following the structure of briefing the supervisor over previous week's progress, going through any issues that arose, discussing tasks for the following week, as well as asking for feedback or development guidance.

The interim work plan in Figure 6.2 has changed over the development process to Figure 6.1 as first semester exam and coursework extensions delayed model implementation until the start of February and extended dissertation writeup by 10 days. As access to the university cluster was only received on the 12<sup>th</sup> of March, this further slowed down model development and parallel dissertation writeup became more difficult with the constantly evolving model architecture based off new results. This shifted the development methodology towards full proposed model implementation with only necessary progression updates towards the dissertation paper. While I believe this methodology was the right choice, unexpected delays in model training and testing meant that in order to meet project deadlines, the number of results would be very limited. Given the nature of transformer-based architecture, the proposed model evaluation should be extended to at least a month for testing due to runtime.

With the release of exam dates, coursework deadlines and previously mentioned adjustments during the development process, the work plan was updated in Figure 6.1.

Further unexpected hindrances arose in tasks like setting up a conda environment on the university cluster that is compatible with CUDA and Nvidia drivers and works within Pytorch. General bugs in the code or test processing, lead to numerous hours of debugging, however this was expected and greatly minimised with good coding practices and version control using GitLab. That being said, no major issues significantly halting development arose and the overall structure of the work plan has been followed as planned.

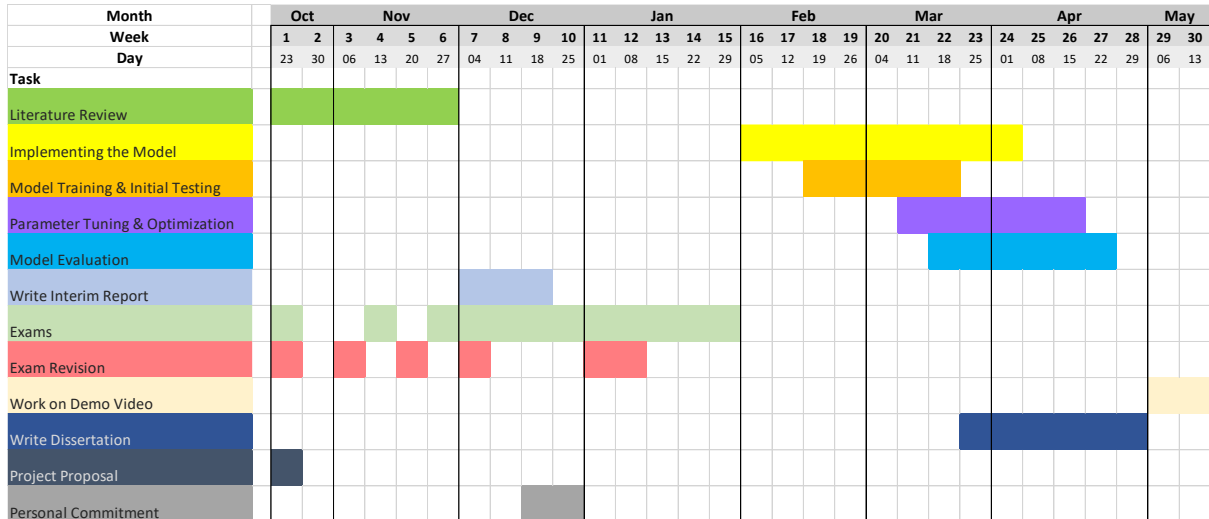


Figure 6.1: Dissertation work plan.

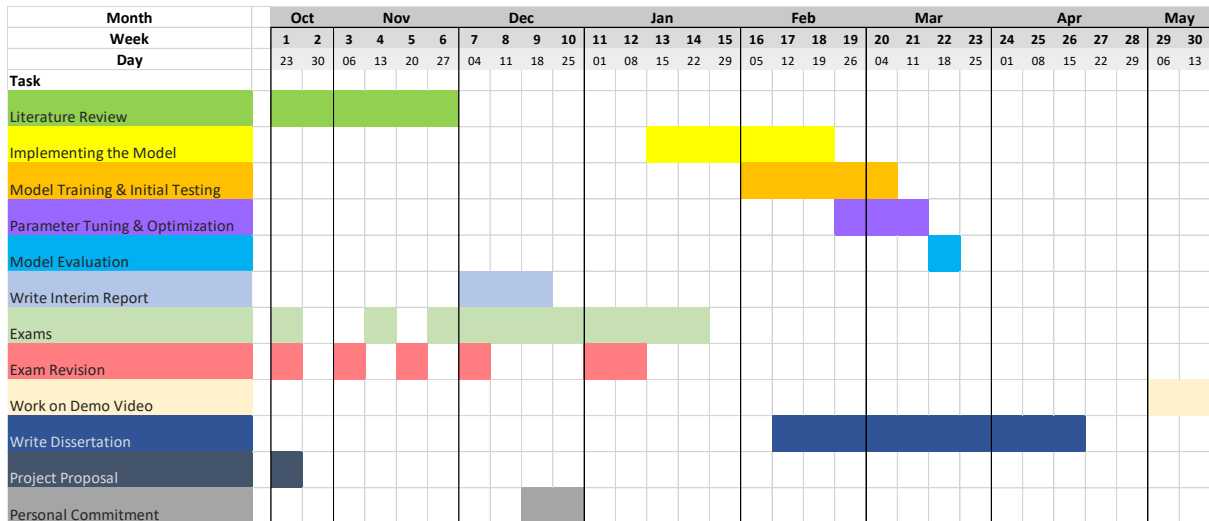


Figure 6.2: Interim work plan.

### 6.3.2 Data Management

Source code, dataset and model weights are saved on the university's GPU cluster. Code and any research information including the paper are uploaded to GitLab for data backup and version management. Notes for meetings, project tasks and any other relevant information is stored on a private personal discord server. This is an easy solution to track, archive and organise the project using simplistic cross platform cloud storage. It is worth noting that no

one else has access to the server and if access was granted by accident, the relevant channels storing the information are private and restricted.


### **6.3.3 Self-Reflection**


I believe that the premise of a hybrid spatial-temporal model has potential outlined by the research. However, due to a lack of transformer accident anticipation research, implementation of the proposed architecture was at times stressful. With more time I would prioritise evaluating the performance of the individual ViViT model against the ConvNeXt-ViViT architecture for a more thorough comparison and optimisation. Adding a different dataset such as the CCD could also be used for accurate implementation of TTA maximisation as a means of comparing the model with existing research.

Due to my very limited past experience with computer vision and machine learning, the research has been overwhelming yet rewarding. I have enjoyed developing a deeper understanding in the field as well as received the chance to apply cutting edge approaches in AI models to produce state-of-the-art results. I am grateful for the opportunity to develop research that has significance and aims to help individuals.



## 7 BIBLIOGRAPHY

- [1] BRAKE (2022). *UK road death and casualty statistics*. [online] Brake. Available at: <https://www.brake.org.uk/get-involved/take-action/mybrake/knowledge-centre/uk-road-safety> [Accessed 18 Oct. 2023].
- [2] Gulati, A., Qin, J., Chiu, C.-C., Parmar, N., Zhang, Y., Yu, J., Han, W., Wang, S., Zhang, Z., Wu, Y. and Pang, R. (2020). Conformer: Convolution-augmented Transformer for Speech Recognition. *arXiv:2005.08100 [cs, eess]*. [online] Available at: <https://arxiv.org/abs/2005.08100>.
- [3] Chan, F.-H., Chen, Y.-T., Xiang, Y. and Sun, M. (2017). Anticipating Accidents in Dashcam Videos. *Computer Vision – ACCV 2016*, [online] pp.136–153. doi:[https://doi.org/10.1007/978-3-319-54190-7\\_9](https://doi.org/10.1007/978-3-319-54190-7_9).
- [4] paperswithcode.com. (n.d.). *Papers with Code - Kinetics 400 Dataset*. [online] Available at: <https://paperswithcode.com/dataset/kinetics-400-1> [Accessed 5 Oct. 2023].
- [5] AutoPilot Review (2019). *Elon Musk on Cameras vs LiDAR for Self Driving and Autonomous Cars*. YouTube. Available at: <https://www.youtube.com/watch?v=HM23sjhtk4Q> [Accessed 3 Oct. 2023].
- [6] Github.io. (2016). Available at: <https://aliensunmin.github.io/project/dashcam/> [Accessed 29 Sep. 2023].
- [7] Bao, W. (2023).  *CarCrashDataset*. [online] GitHub. Available at: <https://github.com/Cogito2012/CarCrashDataset> [Accessed 27 Sep. 2023].
- [8] Muhammad Monjurul Karim, Li, Y., Qin, R. and Yin, Z. (2022). A Dynamic Spatial-Temporal Attention Network for Early Anticipation of Traffic Accidents. *IEEE Transactions on Intelligent Transportation Systems*, 23(7), pp.9590–9600. doi:<https://doi.org/10.1109/tits.2022.3155613>.
- [9] Bao, W., Yu, Q. and Kong, Y. (2020). Uncertainty-based Traffic Accident Anticipation with Spatio-Temporal Relational Learning. *Proceedings of the 28th ACM International Conference on Multimedia*, [online] pp.2682–2690. doi:<https://doi.org/10.1145/3394171.3413827>.
- [10] Suzuki, T., Kataoka, H., Aoki, Y. and Yutaka Satoh (2018). Anticipating Traffic Accidents with Adaptive Loss and Large-Scale Incident DB. doi:<https://doi.org/10.1109/cvpr.2018.00371>.
- [11] Vaswani, A., Noam Shazeer, Parmar, N., Jakob Uszkoreit, Jones, L., Gomez, A.N., Kaiser, Ł. and Illia Polosukhin (2017). Attention is All you Need. *arXiv (Cornell University)*, 30, pp.5998–6008.
- [12] Tesla (2021). *Autopilot*. [online] Tesla.com. Available at: <https://www.tesla.com/autopilot> [Accessed 16 Oct. 2023].

- [13] Alexey Dosovitskiy, Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Matthias Minderer, Georg Heigold, Gelly, S., Jakob Uszkoreit and Houlsby, N. (2020). An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. *arXiv (Cornell University)*.
- [14] Sherstinsky, A. (2020). Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) network. *Physica D: Nonlinear Phenomena*, 404, p.132306. doi:<https://doi.org/10.1016/j.physd.2019.132306>.
- [15] Zhang, Y. and Yang, Q. (2017). An overview of multi-task learning. *National Science Review*, 5(1), pp.30–43. doi:<https://doi.org/10.1093/nsr/nwx105>.
- [16] Martinez, J.J. (2023). *Supervised Fine-tuning: customizing LLMs*. [online] MantisNLP. Available at: <https://medium.com/mantisnlp/supervised-fine-tuning-customizing-llms-a2c1edbf22c3> [Accessed 22 Nov. 2023].
- [17] Liu, Z., Hu, H., Lin, Y., Yao, Z., Xie, Z., Wei, Y., Ning, J., Cao, Y., Zhang, Z., Dong, L., Wei, F. and Guo, B. (2021). Swin Transformer V2: Scaling Up Capacity and Resolution. *arxiv.org*. [online] doi:<https://doi.org/10.48550/arXiv.2111.09883>.
- [18] Peng, B., Alcaide, E., Anthony, Q., Albalak, A., Arcadinho, S., Cao, H., Cheng, X., Chung, M., Grella, M., GV, Kranthi Kiran, He, X., Hou, H., Kazienko, P., Kocon, J., Kong, J., Koptyra, B., Lau, H., Sri, K., Mom, F. and Saito, A. (2023). RWKV: Reinventing RNNs for the Transformer Era. doi:<https://doi.org/10.48550/arxiv.2305.13048>.
- [19] Sun, Y., Dong, L., Huang, S., Ma, S., Xia, Y., Xue, J., Wang, J. and Wei, F. (2023). Retentive Network: A Successor to Transformer for Large Language Models. *arXiv (Cornell University)*. doi:<https://doi.org/10.48550/arxiv.2307.08621>.
- [20] Feten Hajri and Hajer Fradi (2022). Vision Transformers for Road Accident Detection from Dashboard Cameras. doi:<https://doi.org/10.1109/avss56176.2022.9959545>.
- [21] Liu, Z., Mao, H., Wu, C.-Y., Feichtenhofer, C., Darrell, T. and Xie, S. (2022). A ConvNet for the 2020s. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. doi:<https://doi.org/10.1109/cvpr52688.2022.01167>.
- [22] Arnab, A., Dehghani, M., Heigold, G., Sun, C., Lučić, M. and Schmid, C. (2021). ViViT: A Video Vision Transformer. [online] IEEE Xplore. doi:<https://doi.org/10.1109/ICCV48922.2021.00676>.
- [23] Ijaz, M. (2022). ViViT  Video Vision Transformer. [online] AIGuys. Available at: <https://medium.com/aiguys/vivit-video-vision-transformer-648a5fff68a4> [Accessed 21 Dec. 2023].
- [24] United Nations (2023). *Goal 3 | Department of Economic and Social Affairs*. [online] [sdgs.un.org](https://sdgs.un.org). Available at: [https://sdgs.un.org/goals/goal3#targets\\_and\\_indicators](https://sdgs.un.org/goals/goal3#targets_and_indicators) [Accessed 24 Nov. 2023].

- [25] United Nations (2023). *Goal 9 / Department of Economic and Social Affairs*. [online] sdgs.un.org. Available at: [https://sdgs.un.org/goals/goal9#targets\\_and\\_indicators](https://sdgs.un.org/goals/goal9#targets_and_indicators) [Accessed 24 Nov. 2023].
- [26] United Nations (2023). *Goal 11 / Department of Economic and Social Affairs*. [online] sdgs.un.org. Available at: [https://sdgs.un.org/goals/goal11#targets\\_and\\_indicators](https://sdgs.un.org/goals/goal11#targets_and_indicators) [Accessed 24 Nov. 2023].
- [27] Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A.A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., Hassabis, D., Clopath, C., Kumaran, D. and Hadsell, R. (2017). Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, [online] 114(13), pp.3521–3526. doi:<https://doi.org/10.1073/pnas.1611835114>.
- [28] huggingface.co. (2023). *facebook/convnext-tiny-224 · Hugging Face*. [online] Available at: <https://huggingface.co/facebook/convnext-tiny-224> [Accessed 3rd Jan. 2024].
- [29] huggingface.co. (n.d.). *google/vivit-b-16x2-kinetics400 · Hugging Face*. [online] Available at: <https://huggingface.co/google/vivit-b-16x2-kinetics400> [Accessed 3rd Jan. 2024].
- [30] Meinhardt, T., Kirillov, A., Leal-Taixé, L. and Feichtenhofer, C. (n.d.). *TrackFormer: Multi-Object Tracking with Transformers*. [online] Available at: <https://arxiv.org/pdf/2101.02702.pdf> [Accessed 3 Nov. 2023].
- [31] Chen, F., Datta, G., Kundu, S. and Beerel, P. (n.d.). *Self-Attentive Pooling for Efficient Deep Learning*. [online] Available at: <https://arxiv.org/pdf/2209.07659.pdf>.
- [32] pytorch.org. (n.d.). *AdaptiveAvgPool2d — PyTorch 1.13 documentation*. [online] Available at: <https://pytorch.org/docs/stable/generated/torch.nn.AdaptiveAvgPool2d.html> [Accessed 17 Feb. 2024].
- [33] pytorch.org. (n.d.). *Linear — PyTorch 1.8.0 documentation*. [online] Available at: <https://pytorch.org/docs/stable/generated/torch.nn.Linear.html>.
- [34] pytorch.org. (n.d.). *ReLU — PyTorch 1.8.0 documentation*. [online] Available at: <https://pytorch.org/docs/stable/generated/torch.nn.ReLU.html>.
- [35] pytorch.org. (n.d.). *Dropout — PyTorch 1.8.1 documentation*. [online] Available at: <https://pytorch.org/docs/stable/generated/torch.nn.Dropout.html>.
- [36] scikit-learn.org. (n.d.). *sklearn.metrics.average\_precision\_score — scikit-learn 0.24.1 documentation*. [online] Available at: [https://scikit-learn.org/stable/modules/generated/sklearn.metrics.average\\_precision\\_score.html](https://scikit-learn.org/stable/modules/generated/sklearn.metrics.average_precision_score.html).
- [37] pytorch.org. (n.d.). *BCELoss — PyTorch 1.7.0 documentation*. [online] Available at: <https://pytorch.org/docs/stable/generated/torch.nn.BCELoss.html> [Accessed 21 Jan. 2024].
- [38] pytorch.org. (n.d.). *BCELoss — PyTorch 1.7.0 documentation*. [online] Available at: <https://pytorch.org/docs/stable/generated/torch.nn.BCELoss.html> [Accessed 21 Jan. 2024].

- [39] pytorch.org. (n.d.). Adam — PyTorch 1.11.0 documentation. [online] Available at: <https://pytorch.org/docs/stable/generated/torch.optim.Adam.html> [Accessed 21 Feb. 2024].
- [40] Kingma, D. and Lei Ba, J. (2017). *ADAM: A METHOD FOR STOCHASTIC OPTIMIZATION*. [online] Available at: <https://arxiv.org/pdf/1412.6980.pdf>.
- [41] pytorch.org. (n.d.). *ReduceLROnPlateau* — *PyTorch 1.9.0 documentation*. [online] Available at: [https://pytorch.org/docs/stable/generated/torch.optim.lr\\_scheduler.ReduceLROnPlateau.html](https://pytorch.org/docs/stable/generated/torch.optim.lr_scheduler.ReduceLROnPlateau.html). [Accessed 27 Jan. 2024].
- [42] pytorch.org. (n.d.). *Datasets & DataLoaders* — *PyTorch Tutorials 1.11.0+cu102 documentation*. [online] Available at: [https://pytorch.org/tutorials/beginner/basics/data\\_tutorial.html](https://pytorch.org/tutorials/beginner/basics/data_tutorial.html) [Accessed 20 Jan. 2024].