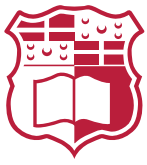
Skateboard Trick Recognition through an AI-based Approach

Kris Saliba

Supervisor: Dr. Joseph Bonello

June 2024

Submitted in partial fulfilment of the requirements
for the degree of Bachelor of Science in Information Technology (Honours)
(Software Development).



L-Università ta' Malta
Faculty of Information &
Communication Technology

Abstract

Acknowledgements

Contents

Abstract i

Acknowledgements ii

Contents v

List of Figures vi

List of Tables vii

List of Abbreviations 1

Glossary of Symbols 1

1 Introduction 1
- 1.1 Motivation . . . 2
- 1.2 Hypothesis . . . 2
- 1.3 Research Questions . . . 2
- 1.4 Aims and Objectives . . . 3
  - 1.4.1 Aims . . . 3
  - 1.4.2 Objectives . . . 3
- 1.5 Structure . . . 3

2 Background 4
- 2.1 Skateboard Tricks . . . 4
- 2.2 Machine Learning . . . 4
- 2.3 Activity Recognition . . . 5
- 2.4 Neural Networks . . . 5
  - 2.4.1 Artificial Neural Networks . . . 5
  - 2.4.2 Convolutional Neural Networks . . . 6
  - 2.4.3 Recurrent Neural Networks . . . 7

3 Literature Review 8
- 3.1 Activity Recognition . . . 8

3.1.1 Challenges in this field . . . 9
3.1.2 Preprocessing techniques . . . 9
3.1.3 Activity Recognition Techniques . . . 11
3.2 Advancements in Skateboard Trick Classification . . . 14
3.2.1 Accelerometer-based approaches . . . 14
3.2.2 Computer Vision-based Approaches . . . 15

4 Methodology **17**
4.1 Class Establishment . . . 17
4.2 Data Preparation . . . 17
4.2.1 Dataset . . . 17
4.2.2 Labelling techniques . . . 17
4.3 Preprocessing . . . 18
4.3.1 Frame Extraction . . . 18
4.3.2 Data Augmentation . . . 19
4.3.3 Normalisation . . . 19
4.3.4 Feature Extraction with Transfer Learning . . . 19
4.3.5 Dimensionality Reduction . . . 20
4.4 Adapted Models . . . 20
4.5 Evaluation Methods . . . 21

5 Implementation **24**
5.1 Development Environment . . . 24
5.2 Frame Extraction using Optical Flow . . . 24
5.3 Dataset Split . . . 25
5.3.1 Data Augmentation . . . 25
5.4 Feature Extraction and Preprocessing for Training . . . 26
5.4.1 PCA Dimensionality Reduction . . . 26
5.5 Hyperparameter Optimisation . . . 27
5.6 Training History . . . 27
5.7 Callbacks . . . 28
5.7.1 Early Stopping . . . 28
5.7.2 Model Checkpoint . . . 28
5.8 Baseline Model Approach . . . 28

6 Implementation **29**
6.1 Experiments . . . 29

7 Sample A **30**

8 Sample B **31**

List of Figures

Figure 1.1 Number of skateboarding participants in the United States from 2010 to 2021 in millions. Reproduced from [**skatestatistics**], data sourced from Outdoor Foundation [**outdoorfounation**]. 1

Figure 2.1 Schematic Representation of an Artificial Neural Network. Reproduced from López et al. (2022) [**FundamentalsOfArtificialNeuralNetworksAndDeepLearning**]

Figure 3.1 CNN-LSTM Architecture. Reproduced from Donahue et al. (2015) [**LongTermRecurrentConvolutionalNetworksForVisualRecognitionAndDescription**] 13

Figure 4.1 Folder-based labelling. 18
Figure 4.2 Text-based labelling 18
Figure 4.3 Visualisation of optical flow: (a) Optical flow representation (b) Individual frames extracted. 18
Figure 4.4 Artefact Architecture. 23

Figure 5.1 Comparison of frame extraction between optical flow method and uniform sampling 25
Figure 5.2 Comparison of Augmented sequence against original 26
Figure 5.3 Model loss graphs for (a) Adam and (b) SGD optimisers. 28

List of Tables

Table 4.1 Characteristics of Transfer Learning Models VGG16 and ResNet50 . . . 20

1 Introduction

Skateboarding dates back to the late 1940s or early 1950s, evolving from a leisure activity for surfers on flat land to a worldwide phenomenon [**SkateboardingEncyclobeadia**]. Its inclusion as an official sport in the 2020 Tokyo Olympic Games [**SkatebaordingOlimpics**] further solidified its popularisation. As Figure 1.1 illustrates, this event caused a spike in the number U.S. skateboarding participants between the years 2020 and 2021.

This dynamic sport encompasses various disciplines and riding styles, each offering unique challenges for skateboarders to explore. Two of the most prominent styles are “vert” and “street.” Vert skateboarding revolves around riding on specialised obstacles, namely, half-pipes and ramps, emphasising aerial manoeuvres. Street skateboarding transpires in urban environments, utilising various obstacles that can be found outdoors, including stairs, rails, ledges, gaps or flat ground for skaters to showcase their creativity [**skateStyles**].

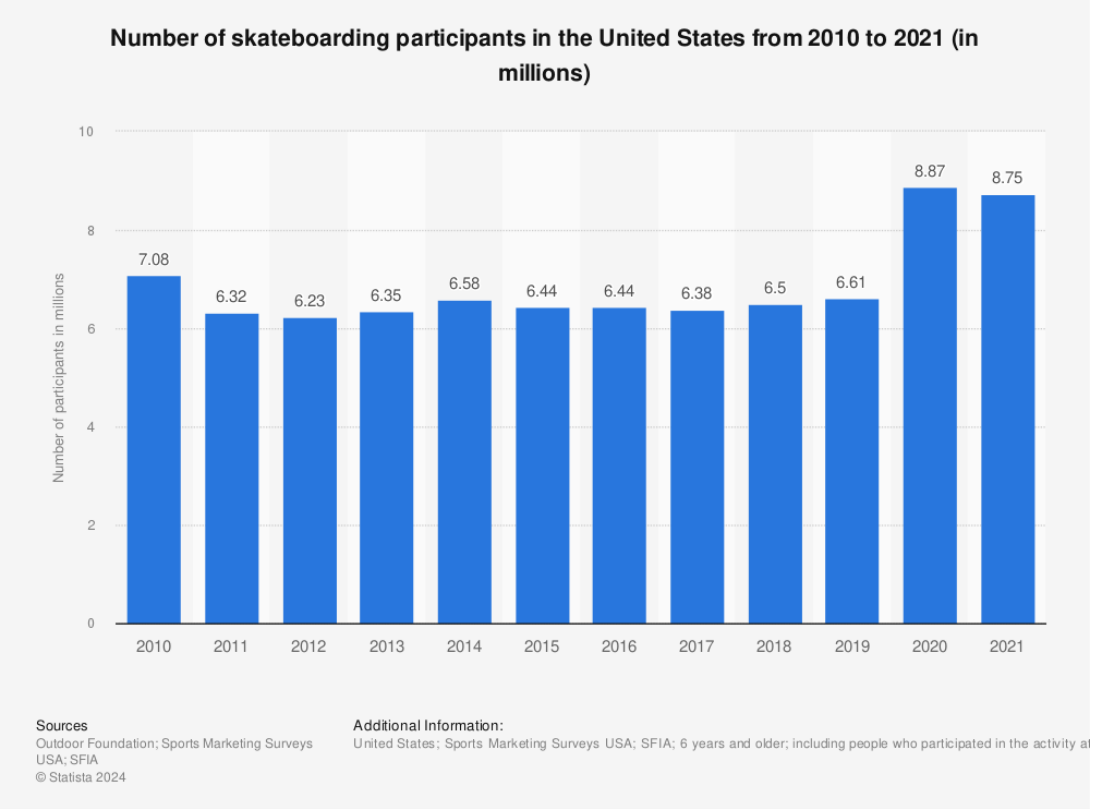


Figure 1.1 Number of skateboarding participants in the United States from 2010 to 2021 in millions. Reproduced from [**skatestatistics**], data sourced from Outdoor Foundation [**outdoorfounation**].

1.1 Motivation

The recent popularity of skateboarding in video content on platforms such as YouTube and Instagram has increased demand for the use of on-screen trick identification. This enables inexperienced viewers to identify tricks performed by skaters in videos through a digital overlay displaying the performed trick. However, existing approaches require manual addition of the performed trick names in videos, which may be time consuming. Automating this process could save time and offer real time benefits for live streamed content. For instance, automated trick recognition could play a vital role in skateboarding competitions, providing insightful footage to viewers, enhancing the understanding and appreciation of the sport. An Artificial Intelligence (AI) model capable of accurately classifying skateboard tricks could significantly improve viewer experience in these scenarios.

Despite the popularity in computer vision, particularly in video activity recognition, limited research exists in the domain of skateboard trick classification. While some initial studies such as, Shapiee et al. (2020) [**skatePaper1**] and Hanxiao Chen (2023) [**SkateboardAIPaper**] explored this domain, there remains a gap in research, driving the need for further growth in this area.

This task poses significant challenges, due to factors such as diverse camera angles, lighting conditions and complex skateboard movements. However, this research aims to overcome these challenges and create a robust and accurate tool that can reliably classify skateboard tricks from video.

1.2 Hypothesis

The hypothesis for this research asserts that employing a combination of deep learning strategies and preprocessing techniques can improve the accuracy and robustness in classifying skateboard tricks.

1.3 Research Questions

- How effectively can combining deep learning techniques and algorithms accurately classify skateboard tricks from video data?
- What is the impact of different video pre-processing techniques on classification accuracy?

1.4 Aims and Objectives

1.4.1 Aims

- To classify skateboard tricks performed in videos into their respective classes.
- To compare the performance of Deep Learning architectures in the context of skateboard trick classification.

1.4.2 Objectives

- To Implement a set of Deep learning architectures and evaluate them using appropriate metrics.
- Employ suitable frame processing techniques

1.5 Structure

This study is structured systematically to explore the potential of deep learning in skateboard trick recognition. Chapter 2 begins by establishing the necessary background knowledge required for this study, while Chapter 3 provides an overview of the past research conducted in this domain.

Next, Chapter 4 outlines the approach behind the artefact, with implementation specifics provided in Chapter 5. Chapter 6, discusses the outcomes and evaluates them against other studies and finally, Chapter 7 presents the final findings acknowledges any limitations that this research encountered.

2 Background

2.1 Skateboard Tricks

Skateboard tricks can be described as dynamic manoeuvres that involve complex orchestration of the skateboard and the skateboarder's body. Skateboard tricks can be categorised into various types, including rotations (flips, rails), grinds (on ledges or rails) and manuals (balancing on two wheels). Successful execution of these tricks relies heavily on precise foot placement, which determines the boards velocity and direction, enabling a skateboarder to manipulate the board to replicate specific tricks.

- **Ollie:** One of the first tricks beginners learn. Where the skateboarder pops the tail of the board while sliding their foot across the board, causing the board to level out in the air, used to jump over obstacles.
- **Kickflip:** A trick where the skateboarder flips the board under their feet while jumping, making it spin 360°around the x-axis.
- **Pop-Shuvit:** A trick where the skateboarder scoops the board with their back foot causing a 180°rotation around the y-axis.

Skateboarders continually innovate and come up with new trick combinations, contributing to the dynamic nature of the sport.

2.2 Machine Learning

Machine Learning (ML) can be defined as a field of study that explores algorithms and statistical models employed by computer systems to execute tasks without the need to be explicitly programmed. It is particularly applicable in situations where the information we seek from a dataset is not interpretable, and as the volume of available datasets continues to surge so does the demand for machine learning [**ML_Algorithms**].

Morris (2019) [**UnderstandingLSTM**] characterises ML as the advancement of algorithms that progressively enhance their performance through practice, suggesting that the more training the learning algorithm undergoes, the better it becomes at executing tasks. Numerous critical factors shape a model's performance within this phase, as exmplified by Budach et al. (2022) [**TheEffectsofDataQualityonMachineLearningPerformance**]. Such factors include dataset quality and diversity, data preprocessing, the selection of a suitable model architecture, training time and the fine-tuning of hyper-parameters.

There exist three main categories for ML models [**ML_Algorithms**]:

- **Supervised:** This is a machine learning concept that centres around the development of algorithms to make predictions or classifications using labelled data. The model is trained on a dataset comprising of example input-output pairs.
- **Unsupervised:** This ML concept concentrates on discovering relationships within data when there are no predefined ”correct” answers or labelled examples to guide the learning process. These algorithms are left to autonomously explore and divulge structures in the data.
- **Reinforcement:** This type of learning consists of an agent that interacts with the environment and learns from the continuous feedback it receives in the form of rewards or punishment.

detection as the following equation where an image is denoted as $\mathcal{I}$, and $O(I)$ represents the collection of object descriptions for objects within the image.

In the above equation, each description encompasses two parts, $Y_i^* \in \mathcal{Y}$ characterises the category or type of an object, and $Z_{N^*i}^* \in \mathcal{Z}$ represents information about its location, size or shape within the image. $\mathcal{Z}$ represents the different ways to describe an object, this is typically done by specifying the object's centre $(x_c, y_c) \in \mathcal{R}^2$ or as a bounding box $(x_{min}, y_{min}, x_{max}, y_{max}) \in \mathcal{R}^4$.

2.3 Activity Recognition

Activity recognition is the process of identifying and categorizing human activities from video sequences. Human activity involves a wide range of motions and interactions with objects, varying from simple isolated actions like dancing to more complex activities that engage multiple body parts and external objects. Similar to object detection, the human ability to perceive these behaviours is a trivial task; yet, it is a challenging problem for computers due to the sequential nature and the resemblance of visual content in such activities [**ActionRecognitionDeepBi-DirectionalLSTM**],[**AReviewOfHumanActivityRecognitionMethods**].

2.4 Neural Networks

2.4.1 Artificial Neural Networks

Artificial Neural Networks (ANNs) are a class of machine learning models that are inspired by the interconnected systems of neurons found in the nervous system of living organisms. They consist of connected nodes capable of learning from their environment and adapting to complex patterns in data

[**FundamentalsOfNeuralNetworks**]. Figure 2.1 depicts a schematic representation of an ANN. The diagram is organised into three fundamental layers: the Input Layer, the Hidden Layer(s) and the Output Layer [**FundamentalsOfArtificialNeuralNetworksAndDeepLearning**].

- **Input Layer:** This is the set of neurons that serve as the initial entry point for external data or features. Each input neuron in this layer corresponds to a specific feature or variable used in the Neural Network model.
- **Hidden Layer(s):** This is the set of neurons that are situated between the Input and Output Layers where the network captures complete non-linear behaviours of data and feature transformations.
- **Output Layer:** This is the set of neurons that provide the final predictions produced by the neural network. Depending on how the ANN is configured, the final output can be continuous, binary, ordinal, or count.

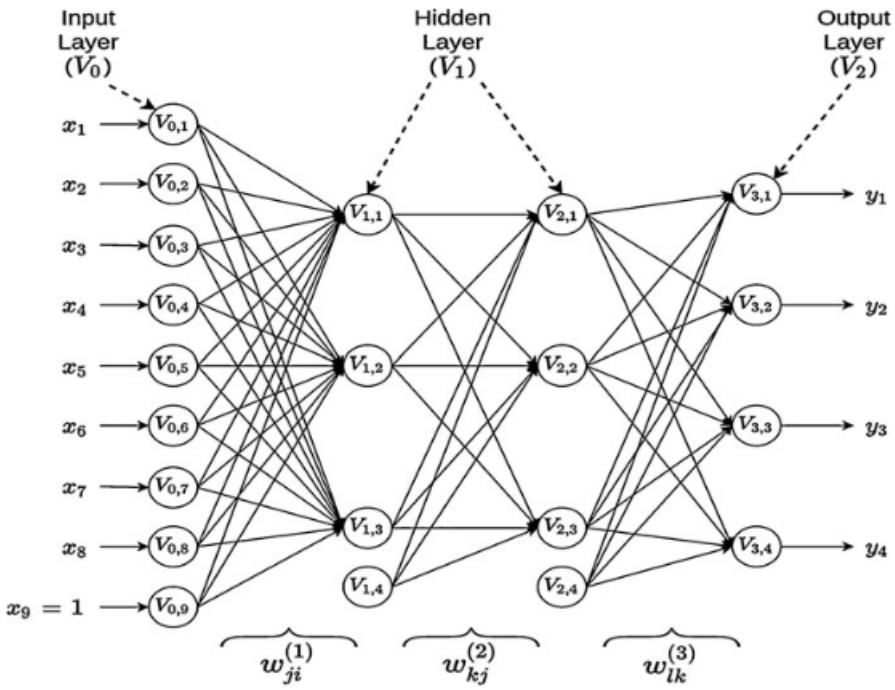


Figure 2.1 Schematic Representation of an Artificial Neural Network. Reproduced from López et al. (2022) [**FundamentalsOfArtificialNeuralNetworksAndDeepLearning**]

2.4.2 Convolutional Neural Networks

Convolutional Neural Networks (CNNs), as described by Gu et al. (2018) [**RecentAadvancesInConvolutionalNeuralNetworks**] are a category of Deep learning architectures with roots in the biological visual perception mechanisms of living

organisms. These networks have gained widespread attention for their incredible performance in various fields such as visual recognition, speech recognition and natural language processing.

CNNs, incorporate multiple layers and are capable of extracting effective representations

– continue explanation, explain how feature maps are created through convolutions

2.4.3 Recurrent Neural Networks

Recurrent Neural Networks (RNNs) are a subset of Neural networks that are designed for sequential data processing. RNNs are capable of modelling dynamic relationships in sequential data by feeding signals from past time steps back into the network. However, they are limited by their inability to access long-term data, limited to approximately ten sequential time steps. This constraint arises from the challenge of dealing with vanishing or exploding gradients in the backpropagation procedure while processing extended sequences, as discussed in prior works [**UnderstandingLSTM**],[**Long-termConvolutionalNeuralNetworksforVisualRecogntion**].

To address the limitation that RNNs encounter in capturing long-term dependencies, Long Short-Term Memory Recurrent Neural Networks (LSTM-RNNs) were introduced [**learningPriceseTimingWithLSTM**]. LSTM networks unlike RNNs are capable of capturing dependencies across more than 1,000 time steps depending on the network's complexity and are more biologically credible [**UnderstandingLSTM**].

3 Literature Review

3.1 Activity Recognition

Building on the foundational understanding of activity recognition defined in the background, it's important to acknowledge the impact it has on various sectors. The recent advancements in recognising human actions in videos have have not only revolutionalised sectors such as healthcare and security as demonstrated in the studies [**3DhumanActionDetectionForHealthCareSystems**], [**HumanActivityRecognitionSecurityAndMonitoring**], but also hold extensive potential in the realm of sports.

The research conducted by K. Host and M. Ivašić-Kos in [**HARinSportsComputerVision**] along with Wu et al. in [**AsurveyonvideoactionrecognitioninsportsDatasetsmethodsandapplications**] further elaborate on this potential, such as its numerous applications in categorising complex sports actions, injury prevention methods and refinement of game strategies through video analysis. These studies also propose various methodological advancements, including the utilisation of Deep Learning models to enhance accuracy, the exploration of multimodal data sources for sports activities and an emphasis on real-time analysis capabilities. These insights provide valuable techniques that can be leveraged for the development of activity recognition models in the field of sports.

The study by Beddiar et al. [**VisionBasedHARASurvey**], identifies two main streams of Human-computer interaction (HCI) technologies: Contact-based and Vision-based systems. The authors categorise contact-based HCI as those technologies that require physical user interaction through mediums such as accelerometers, wearable sensors and multi-touch interfaces. Alternatively, the authors describe vision-based methods as the simplification of HCI due to more natural human communication, eliminating the need for physical contact or equipment. These methods use image and video data to recognise human activities offering an advantage in terms of societal acceptance and usability.

While both contact and vision-based systems have their merits, this study will specifically focus on vision-based techniques and their application in the development of a skateboard trick classifier. These methods, which utilise image and video data to recognise human activities are selected due to their societal acceptability and their applicability in sports broadcasts.

3.1.1 Challenges in this field

The domain of Activity recognition comes with many challenges as depicted by Zhang et al. (2017) [**ReviewOfHumanActivityRecognitionUsingVisionBasedMethod**]. The researchers suggest that while certain scenarios utilise static cameras such as surveillance systems, most situations that benefit from Activity Recognition adopt dynamic recording devices such as mobile phones and sports event broadcasts. These dynamic devices introduce a significant level of complexity as a result of their tricky dynamic backgrounds present in video footage. Zhang et al. also point out specific difficulties posed by long-distance and low-quality videos, often encountered in environments like crowded public spaces and sports events. The camera distance, results in smaller subjects, making detailed analysis of human movements more challenging while lower-quality videos further complicate the task for Human Activity Recognition (HAR) systems.

The challenges highlighted by Zhang et al. [**ReviewOfHumanActivityRecognitionUsingVisionBasedMethod**] in the domain of activity recognition are directly relevant to the development of a skateboard trick classifier. In scenarios like televised skateboarding events, skaters may appear relatively small to accommodate the entire skatepark, This factor along with the complexity of the background as a result of dynamic recording poses a challenge for trick recognition in live broadcasts. Furthermore, if a skateboard trick classifier is intended for personal development, then it may encounter videos of lower quality further complicating the task of trick recognition. Addressing these challenges is crucial for the development of a skateboard trick classifier that performs well in real-world conditions.

3.1.2 Preprocessing techniques

Preprocessing is a crucial step in the development of ML models, especially in the field of activity recognition. It involves the application of various techniques to enhance raw data before feeding it to Machine Learning algorithms. —continue or leave out

As a result of the limited research on the emergence of a skateboard trick classifier, there is a significant lack of open-source datasets featuring skateboard tricks. This lack of data presents an opportunity to employ data augmentation techniques, especially valuable in studies with limited data. Methods such as flipping, rotating, scaling and colour manipulation not only artificially enhance the size of the original dataset but also lower the likelihood of overfitting as highlighted in prior works, [**DataAugmentationCanImproveRobustness**], [**AnOverviewOfOverfittingAndItsSolutions**]. In the realm of Skateboard trick classifiers, Shapiee et al (2020) [**skatePaper1**] effectively employed data augmentation

techniques to expand their dataset, demonstrating the application of these methods in improving model performance for trick classification.

After establishing the role of data augmentation in addressing the lack of data available, another important step in computer vision is data normalisation. This technique is essential for standardising the range and distribution of pixel values in an image and has shown an increase in model performance, [**RobustnessInMLNormalisation**], [**RealWorldMicroGraphDataQualityNORMALIZATIONCITE**]. Min-max normalisation is particularly prevalent in normalising pixel intensities to a 0-1 scale as follows:

$$\text{Normalized Value} = \frac{\text{Pixel Value} - \text{Min Value}}{\text{Max Value} - \text{Min Value}} \tag{3.1}$$

While less common in computer vision due to its normality assumption, Z-score normalisation is defined as:

$$\text{Z-score} = \frac{x - \mu}{\sigma} \tag{3.2}$$

Here, x represents the individual pixel value, μ is the mean of all pixel values, and σ is their standard deviation.

The research by Pei et al. (2023) [**RobustnessInMLNormalisation**], explored the impact of normalisation on classification accuracy. They demonstrated that, for 8-bit images, min-max normalisation outperformed Z-score, significantly enhancing classification accuracy. On the other hand, the study by de Raad et al. [**EffectOnPreProcessingOnCNNForMedicalImageSegmentation**] suggests that the impact of normalisation on model performance varies depending on certain dataset characteristics. These contrasting conclusions presented by both studies suggest that while normalisation is an important preprocessing step, its application should be carefully adapted to the characteristics of the dataset.

Having optimised the distribution of pixel values through the normalisation process, the next crucial step is feature extraction. Xudong Jiang [**FeatureExtractionForImageRecognitionAndComputerVision**], describes this technique as the process of capturing the core attributes of an object through the elimination of redundancies, resulting in a set of numerical features ideal for classification. CNNs excel at this due to their capabilities in detecting complex patterns and enhancing features. While they are primarily used in image recognition, as an initial step before deploying classification algorithms, their effectiveness in feature extraction is demonstrated by studies like Manjunath Jogin et al. (2018) [**FeatureExtractionUsingCNNandDeepLearning**] where they achieved 86% accuracy using CNNs for feature extraction alongside several classifiers. Beyond this primary function, CNN's ability to extract complex features from images makes them very effective when coupled with other models for more complex tasks such as activity

recognition. In the context of skateboard trick classification, CNNs can efficiently extract detailed spatial features like board rotations, foot positioning and limb movements. Such features can then be passed through a sequence analysis model like an LSTM, proven to be efficient at understanding temporal information.

Following the discussion on CNNs for feature extraction, it is important to highlight the role of transfer learning in enhancing ML models, especially in fields with limited data like skateboard trick classification. The concept of transfer learning as detailed by the studies [**ASurveyOfTransferLearning**] and [**ApplicationAndAnalysisOfTransferLearningSurvey**] involves using a pre-trained ML model to leverage its experience for a new, but related task. The study by Sargano et al. (2017) [**HARUsingTransferLearningWithDeepRepresentations**], employed transfer learning with pre-trained deep CNNs like AlexNet [**AlexNetCite**] and GoogleNet [**GoogleNetCite**], for human activity recognition. Notably, they utilised these models for feature extraction, followed by an SVM classifier for final classification. Their approach showcases the resource-efficient and time-saving nature of transfer learning, evident in their impressive accuracies of 98.15% and 91.47%. Moreover, research on transfer learning is not unique within the field of skateboard trick classification. Two other studies [**skatePaper1**], [**SkateboardAIPaper**] have utilised this approach and achieved high accuracies, however, a more in-depth analysis of these papers can be found in the 'Advancements in Skateboard Trick Classification' section. These studies strongly suggest the exploration of various pre-trained models for feature extraction in the development of a skateboard trick classifier.

3.1.3 Activity Recognition Techniques

The accurate classification of skateboard tricks poses a unique challenge for computer vision due to the sport's dynamic and complex nature, characterised by rapid movements, potential occlusions and camera angles. This section explores various computer vision techniques and architectures employed in previous literature, exploring their potential impact on this specific task.

Deep learning (DL) techniques have become increasingly popular over traditional ML methods, for their ability to learn feature representations automatically from raw data, significantly improving performance [**AReviewOnComputerVisionBasedMethodsForHARRecognition**]. One particularly effective DL architecture is the CNN-LSTM. This approach leverages the CNNs strength in extracting spatial features from individual frames and LSTMs ability to capture temporal information across frames, leading to a deeper understanding of video content.

The study by Orozco et al. (2020)

[**HARRecognitionInVideosUsingARobustCNNLSTMApproach**] adopted this approach to test its effectiveness against three activity recognition datasets: KTH, UCF-11, HMDB-51, particularly focusing on how the number of LSTM units impacted performance. The authors employed transfer learning, utilising the VGG16 model [**VGG**] for feature extraction from videos, followed by an LSTM network for classification. Their findings show that 360 LSTM units achieved an accuracy of 93.86% on the KTH dataset, while 320 units led to an accuracy of 91.93%. However, the performance on the HMDB-51 dataset dropped, with 400 LSTM units resulting in a lower accuracy of 47.36%. These findings show the potential of the CNN-LSTM approach, particularly for simpler datasets, highlighting the need for further investigation and optimisation for more complex datasets like HMDB-51.

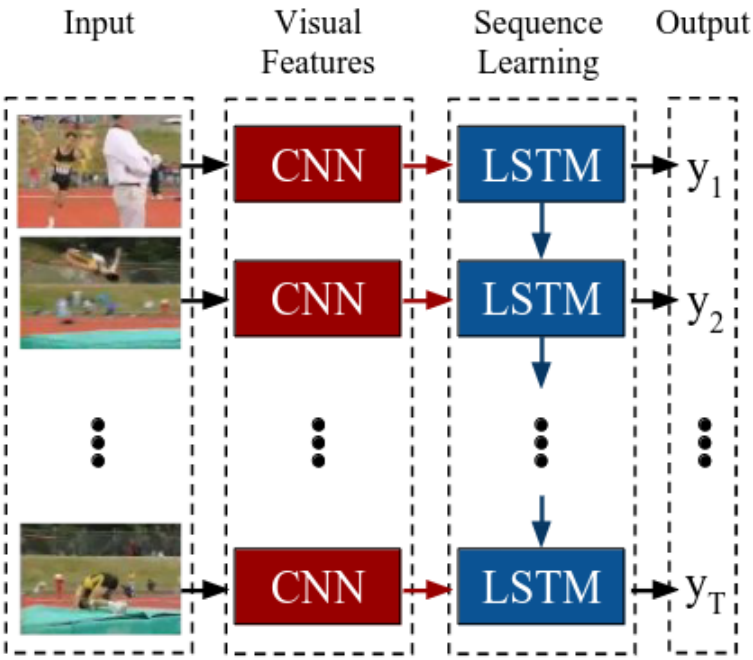


Figure 3.1 CNN-LSTM Architecture. Reproduced from Donahue et al. (2015) [**LongTermRecurrentConvolutionalNetworksForVisualRecognitionAndDescription**]

Building upon the foundational CNN-LSTM architecture, a recent study by Saoudi et al. (2023) [**Advancinghumanactionrecognition:ahybridapproachusingattention-basedLSTMand3DCNN**] enhances this model by incorporating three key advancements:

1. **3D Convolutional Neural Networks (3D CNNs):** Unlike regular CNNs, data is processed as 3D volumes, enabling them to capture both spatial and temporal information by performing convolution operations on all three dimensions: width, height and time. The authors opted to use the I3D model, leveraging transfer learning to bypass the resources required to train one from scratch. The I3D model was selected for its proven efficiency and its ability to be fine-tuned for activity recognition tasks.

2. **Bi-directional Long-Short-Term Memory (BiLSTM) network:** The authors chose to use a variant of LSTM called Bi-directional Long Short-Term Memory (BiLSTM). Whereas LSTMs only process data in one direction, BiLSTMs can analyse data in both directions, allowing them to understand relationships between preceding and subsequent actions.

3. **Attention Mechanisms:** Saoudi et al. incorporated attention mechanisms after their BiLSTM, enabling the model to prioritise particular parts of the input data. This technique allowed the model to learn which temporal features were most

applicable to the task, providing a more detailed representation of the input and ultimately, improved performance.

With the integration of these advancements, Saoudi et al. achieved model accuracies of 97.98% and 96.83% on the HMDB51 and UFC101 datasets respectively, demonstrating the potential of 3D CNNs, BiLSTMSs and attention mechanisms for activity recognition tasks.

3.2 Advancements in Skateboard Trick Classification

In the emergent field of skateboard trick classification, leveraging activity recognition techniques from a video have led to two primary methodologies among researchers. The first technique involves utilising signals obtained from skateboard-mounted accelerometers or signals artificially generated based on the findings of prior studies. These signals are then fed into a study-dependent model for classification, as outlined by Abdullah et al. (2021) [**skateboardClassificationTransferLearningPipelinesAccelermetry**] and Corrêa et al. (2017) [**skateboardTrickClassifierUsingAccelerometryAndML**]. The second approach employs computer vision techniques, leveraging video footage of skateboard tricks to train and refine models for accurate trick identification, as depicted by the studies Shapiee et al. (2020) [**skatePaper1**] and Hanciao Chen (2023) [**SkateboardAIPaper**].

3.2.1 Accelerometer-based approaches

The study by Abdullah et al. (2021) [**skateboardClassificationTransferLearningPipelinesAccelermetry**], makes use of a custom dataset comprising six skateboard tricks most commonly executed in competitive events. Amateur skateboarders performed each trick five times on a modified skateboard equipped with an Inertial Measurement Unit (IMU) to record the signals produced. The researchers capture six signals for each trick, including linear accelerations along the x, y, and z axes (aX, aY, aZ) and angular accelerations along the same axes (gX, gY, gZ). They then opt for the unique approach of concatenating all six signals onto a single image corresponding to one trick, employing two input image transformations: raw data (RAW) and Continuous Wavelet Transform (CWT).

With the application of six transfer learning models on this data, Abdullah et al. [**skateboardClassificationTransferLearningPipelinesAccelermetry**] reports exceptionally high accuracies, achieving a 100% test accuracy over multiple models. While these results are remarkable, very high levels of accuracy are rare in ML applications and are typically associated with models that may be overfitting the data.

Recognising the rarity of such high accuracies, this study will consider these findings and efforts will be made to ensure a robust model by employing techniques to avoid overfitting such as early-stopping and the use of a diverse dataset [**AnOverviewOfOverfittingAndItsSolutions**].

The study by Corrêa et al. (2017) [**skateboardTrickClassifierUsingAccelerometryAndML**], obtained their sample data by artificially generating 543 signals based on prior research, utilising tools such as MATLAB 2015 and Signal Processing Toolbox. These signals were then categorised into five distinct classes representing different skateboard tricks, each with various samples ranging from 30 to 50 per class, across three axes (X, Y and Z). This study developed and validated individual Artificial Neural Networks (ANNs) for each axis, as well as the combination of the three: ANN XYZ, displaying the potential of Neural Networks to categorise multidimensional skateboard tricks. The ANNs are all multilayer feed-forward neural networks (MFFNNs), structured into three distinct layers. They feature an input layer with 82 neurons, a hidden layer, comprised of 23 neurons utilising a tan-sigmoid transfer function and an output layer consisting of 5 neurons with a softmax function. Finally, the study achieved high accuracies, with ANNs X, Y and Z achieving 94.8%, 96.7% and 98.7%, respectively, while the combined ANN XYZ achieved an accuracy of 92.8%.

3.2.2 Computer Vision-based Approaches

The paper by Shapiee et al. (2020) [**skatePaper1**] leverages a custom data set comprising videos capturing the execution of five distinct skateboard tricks, each attempted five times. Each video spans two to three seconds, yielding a total of 750 images by extracting 30 frames per video. This study made use of data augmentation techniques to expand their dataset further. Consequently, they introduced an additional 2,250 images, achieving 3,000 images in their data set. On the other hand, Chen (2023) [**SkateboardAIPaper**] compiled a comprehensive data set by collecting videos from multiple platforms, including YouTube, Twitter and Instagram. Furthermore, Chen trained the model using 15 fundamental tricks commonly observed in competitive settings. The researcher collected 50 videos per trick, summing up to a total number of 750 videos. Of these, 45 videos per trick were allocated for training, and the remaining 5 were reserved for validation.

The paper by Shapiee et al. [**skatePaper1**] utilises data augmentation techniques with the application of three rotations to the images: horizontal rotation, positive 90°rotation and negative 90°rotation. The researchers experimented on three Transfer learning models: MobileNet, NASNetMobile and NASNetLarge, each evaluated using a k-Nearest Neighbor (k-NN) classifier. As a result, the models

demonstrated impressive classification accuracies, with MobileNet achieving 95%, NASNetMobile 92% and NASNetLarge 90%.

In the student abstract by Hanciao Chen [**SkateboardAIPaper**], extensive experimentation is conducted using diverse models, exploring various combinations of CNN-LSTM and CNN-BiLSTM architectures. The study also incorporated attention mechanisms and explored transfer-based methods for activity recognition. This study further documents and analyses important metrics such as training time, training accuracy and validation accuracy for each model experimented on. Among these, the top three models that stood out in terms of validation accuracy were the ResNet50 with Attention and BiLSTM (84%), ResNet50 with BiLSTM (81%) and ResNet50 with LSTM (80%). Chen's study provides valuable insight into the application of diverse models in activity recognition in skateboarding.

4 Methodology

4.1 Class Establishment

This study pursued a multi-class classification strategy, targeting three fundamental skateboard tricks: ollie, pop shuvit and kickflip. These tricks were selected based on two primary criteria. Firstly, they are often associated with the first tricks learnt by beginners, highlighting their role in foundational skateboarding skills. Secondly, their popularity within the skateboarding community, often performed in competitions emphasises their relevance, making them highly relevant for analysing and evaluating competitive performance.

4.2 Data Preparation

4.2.1 Dataset

This study utilised video recordings of skateboarders performing tricks as its primary data source. To ensure model robustness, the final dataset consisted of videos across diverse environmental conditions and varying skateboarder skill levels.

The initial dataset was sourced from the publicly available ”SkateboardML” repository on GitHub [**lightningdrop2020skateboardml**], comprising 200 video clips corresponding to two common tricks: the ollie and the kickflip. To expand the dataset's diversity, additional data was obtained through direct communication with Hanxiao Chen, the author of the SkateboardAI paper [**SkateboardAIPaper**]. This communication yielded a dataset containing 750 videos covering 15 distinct tricks. Given the wide range of tricks included in this dataset, many were beyond the scope of this study, therefore only a subset of these videos were selected and included in the final dataset.

4.2.2 Labelling techniques

This study investigated two primary labelling techniques: the folder-based and the text-based approach as visualised by Figures 4.1 and 4.2. In the folder-based method, videos were categorised into folders named after their corresponding class label offering a simple organisation method. On the other hand, in the text-based approach, each video's path and corresponding label were listed on a text file, providing more flexibility. Given the limited number of classes and manageable dataset size, this study chose to utilise the folder-based approach, as the extra complexity from the text-based method wasn't necessary for this project.

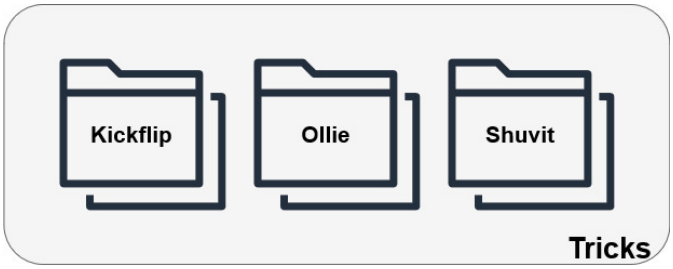


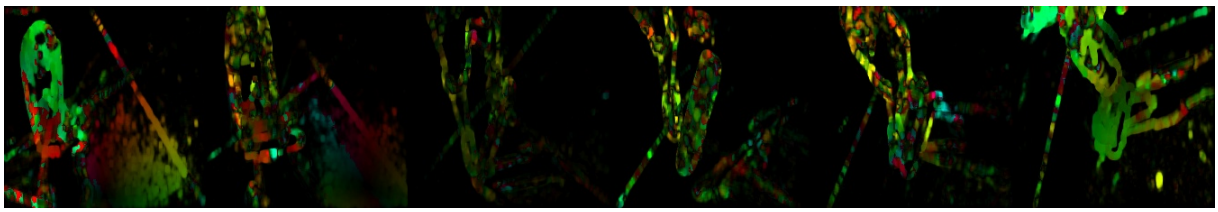
Figure 4.1 Folder-based labelling.

Figure 4.2 Text-based labelling

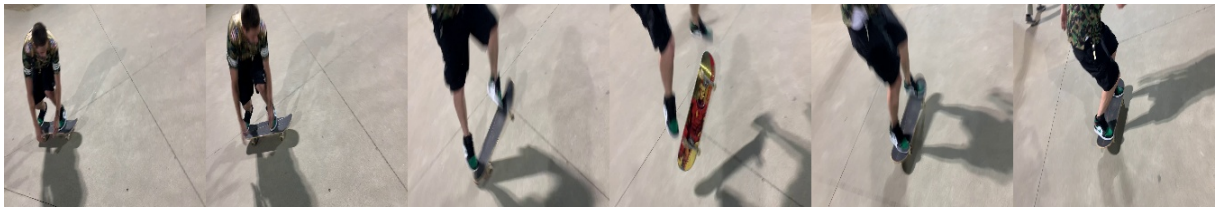
4.3 Preprocessing

4.3.1 Frame Extraction

Given that the nature of this research is an activity recognition task, this study employed frame extraction to convert videos into sequences of frames suitable for processing by ML models. This technique aims to extract a subset of frames that encapsulate the essential dynamics of the activity. Thus, in the context of skateboard trick recognition, the frames should capture the entire sequence, including the wind-up, the trick and the landing.



(a) Optical flow visualisation of a kickflip sequence, highlighting regions with significant motion.



(b) Extracted frames from the sequence based on the greatest weightings from the optical flow representation.

Figure 4.3 Visualisation of optical flow: (a) Optical flow representation (b) Individual frames extracted.

Two primary frame extraction techniques were explored, uniform sampling and optical flow method. The former technique evenly splits a video into its corresponding

frames by using a pre-calculated step size determined by the number of frames of a video. The second technique leveraged optical flow, a method that assigns weights to frames determined by the motion of pixels between them. With this approach, frames with the greatest weightings were chosen for extraction, resulting in capturing frames with the most movement. An example of this technique can be observed in Figure 4.3.

4.3.2 Data Augmentation

Due to the limited number of data used in this research, data augmentation techniques were implemented to increase the dataset's size before model training. These techniques including rotating, flipping and adding noise to images also help reduce overfitting, which was a risk evident due to the over-parametrised nature of the chosen models, meaning that their number of trainable parameters surpass the size of the dataset, making them vulnerable to overfitting.

4.3.3 Normalisation

During the preprocessing stage, normalisation played a crucial role in preparing the video frame data before further processing. In particular this study utilised min-max normalisation to scale the pixel intensities between 0 and 1. Normalising data in this manner ensured that the model's input values data was within a standardised range.

4.3.4 Feature Extraction with Transfer Learning

This research leveraged two well-established CNNs for this task: VGG16 and ResNet50. Both models were pre-trained on the large ImageNet dataset [**imageNetDataset**], leveraging extensive image recognition knowledge that were then fine-tuned to the domain of skateboard trick recognition.

VGG16: VGG16, introduced in 2014 by Simonyan and Zisserman [**VGG**], is a CNN architecture known for its success in image classification and feature extraction for action recognition tasks. This model is relatively simple made up of 16 convolutional and fully connected (FC) with repeated use of 3x3 convolutional filters to preserve spatial information within frames.

Resnet50: Resnet50, introduced in 2016 by He et al. [**Resnet50**], is another powerful CNN architecture that is well known for its advancements in image classification and frame extraction capabilities. Unlike VGG16, this model has a deeper architecture employing 50 convolutional layers allowing it to learn more intricate feature representations of the data. Furthermore, it incorporates residual connections to counteract the vanishing gradient problem that is often caused by many layers.

Table **??** summarises the key characteristics of VGG16 and ResNet50.

Model	VGG16	ResNet50
Published	2014	2016
Layers	16 (Convolutional and FC layers)	50 (Convolutional layers)
Parameters	∼138 million	∼25 million
Input Image Size	224x224	224x224

Table 4.1 Characteristics of Transfer Learning Models VGG16 and ResNet50

The final four fully-connected (FC) layers of these models which are typically used for classification were removed. This was crucial to adapt these models for feature extraction, as the aim was not to classify each frame of the video, but to extract a collection of features that could help sequence models gain a better understanding. These features in the form of vectors not only encapsulate the visual information present in each frame but also more abstract ideas such as low-level information (like edges/shapes) to higher-level features like objects and even actions themselves.

4.3.5 Dimensionality Reduction

The use of pre-trained models used in this study generate high-dimensional feature vectors, while these features capture valuable information, a large number of dimensions can lead to challenges. Firstly, it increases the computational costs of training models. Secondly, it could potentially lead to the ”curse of dimensionality” as explained by Venkat (2018) [**curseofdimensionality**], where models may struggle to learn effectively with high dimensional data. To address these issues, dimensionality reduction is employed after feature extraction to reduce the number of features while still retaining valuable information for classification.

4.4 Adapted Models

This research investigates the performance of four different deep learning architectures for skateboard trick classification. these models leverage pre-trained CNNs for feature extraction followed by sequence modelling techniques to capture the temporal dynamics of skateboard tricks. The architectures investigated are as follows.

1. **VGG16-LSTM:** This architecture leverages the pre-trained VGG16 model to extract features, which are then fed to an LSTM for sequence modelling. LSTMs are powerful for capturing long-term dependencies within sequential data such as skateboarding tricks.

2. **ResNet50-LSTM:** This architecture utilises the deeper ResNet50 pre-trained CNN for feature extraction, followed by an LSTM for sequence modelling. The increased depth of ResNet50 potentially allows it to learn more complex feature representations than VGG16.
3. **VGG16-BiLSTM:** This architecture employs the VGG16 model for feature extraction and a BiLSTM network for sequence modelling. BiLSTMs are known to be able to learn dependencies in both directions of a sequence, which could be useful for capturing subtle details in skateboard tricks.
4. **ResNet50-BiLSTM:** This architecture combines ResNet50 for feature extraction with a BiLSTM network. This combination leverages the potential advantages of deeper feature learning and the bi-directional capabilities of BiLSTMs.

The effectiveness of these models in prior research for video-based action recognition tasks motivated their selection for this study. For instance, Orozco et al. (2020) achieved an outstanding 91.93% accuracy using a VGG-LSTM architecture [**HARRecognitionInVideosUsingARobustCNNLSTMApproach**]. Additionally Chen's work [**SkateboardAIPaper**] explored all four of these models in the context of skateboard trick classification and demonstrated potential for competitive results. While this study also explored attention mechanisms within these models, the main aim is to build upon existing knowledge and potentially achieve better performance.

4.5 Evaluation Methods

To thoroughly evaluate the performance of the proposed models, this study employed various evaluation metrics, including accuracy, precision, recall, F1-score and confusion matrix. These metrics provided valuable insights into the model's ability to correctly classify different tricks.

The above mentioned metrics depend on the following definitions, described in the context of the "kickflip" class.

- **True Positive (TP)**: The model correctly identified a video as containing a kickflip.
- **True Negative (TN)**: The model incorrectly identified a video as containing a kickflip.
- **False Positive (FP)**: The model correctly identified a video as not containing a kickflip.
- **False Negative (FN)**: The model incorrectly identified a video as not containing a kickflip.

Accuracy: Measures the proportion of correctly classified instances, over the total number of predictions made by the model. This metric provides a generic indicator of the model's reliability, however this specific metric can be sensitive in scenarios with class imbalance [**classificationAssessmentMethodstharwat2020**].

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.1}$$

Preci-
sion: Measures the proportion of predicted positive instances that are truly real positives [**Evaluation:fromprecisionrecallandF-measuretoROCinformednessmarkednessandcorrelation**]. In the context of kickflip detection, it represents the proportion of true kickflips against all predicted kickflips.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{4.2}$$

Recall: Measures the proportion of positive instances that are truly predicted positive [**Evaluation:fromprecisionrecallandF-measuretoROCinformednessmarkednessandcorrelation**].In the context of kickflip detection, it represents the proportion of true kickflips that are identified correctly.

$$\text{Recall} = \frac{TP}{TP + FN} \tag{4.3}$$

F1-score: This metric provides a balanced measure of performance by combining both precision and recall. It is calculated using the harmonic mean on both values and outputs a value ranging from zero to one, with values closer to one, indicating better performance.

$$\text{F1-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4.4}$$

Confusion Matrix:

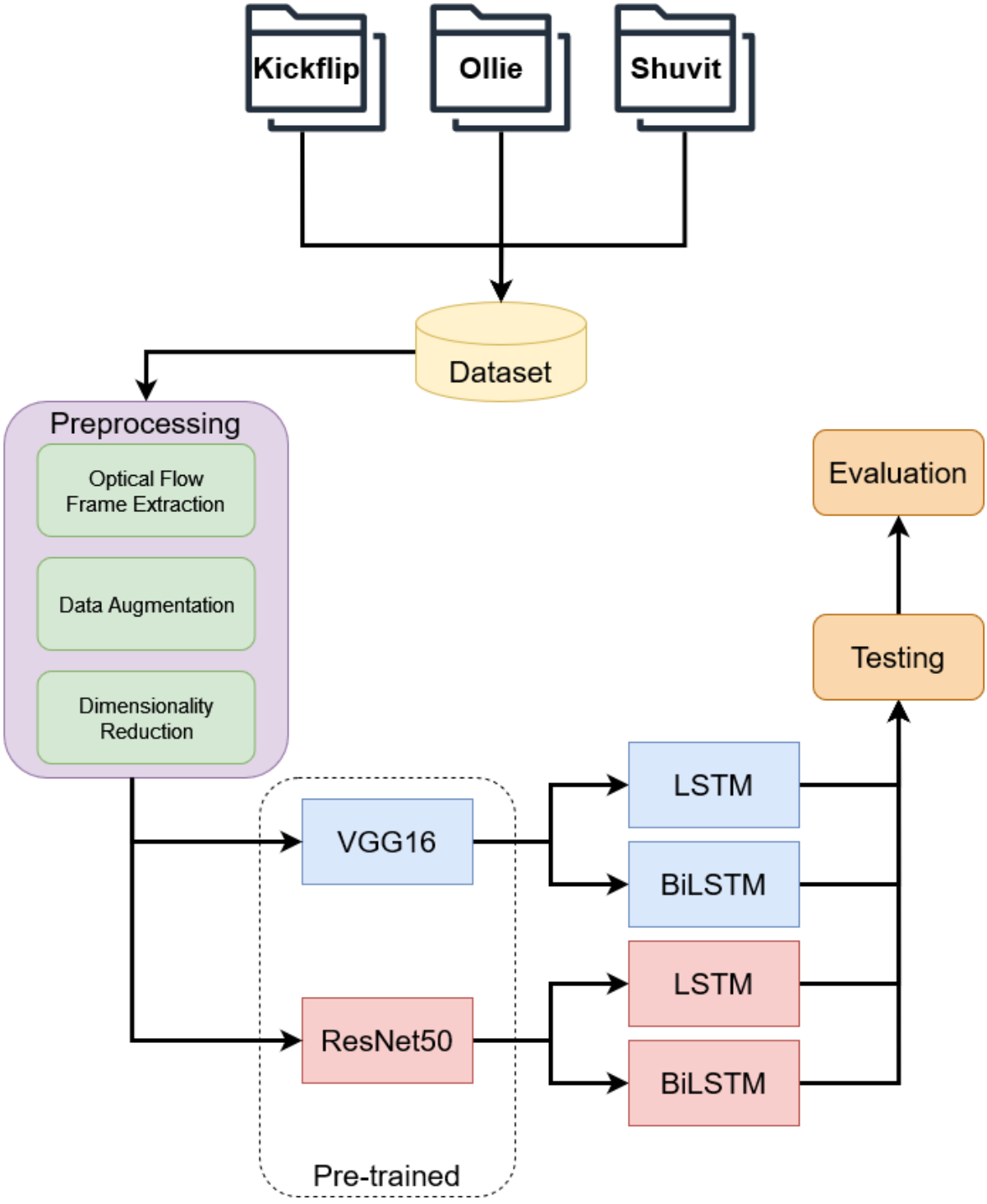


Figure 4.4 Artefact Architecture.

5 Implementation

5.1 Development Environment

Python: Python's large database of libraries, along with its wide use in Machine Learning, made it the ideal choice for developing this artefact. Furthermore, its large and active community made tackling problems and troubleshooting issues simpler. Prior experience using Python also contributed to this decision.

Tensorflow: Tensorflow is an open-source library, powerful in numerical computation and Machine Learning applications. It provides a rich toolset that allows for efficient development, training and deployment of Machine learning models. Notably, Tensorflow comes with the Keras API, further simplifying the creation development by offering wide ranger of tools such as access pre-trained models

OpenCv (cv2): This open source library aided

Matplotlib: This research made use of Matplotlib for its effective plotting and data visualisation functions, heavily used during the evaluation stage to properly visualise results or any other insights gained throughout.

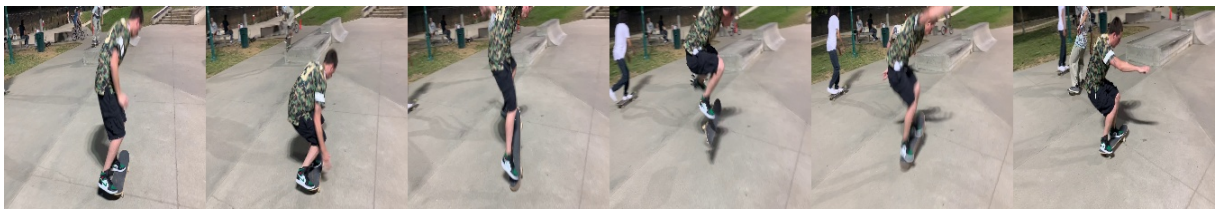
5.2 Frame Extraction using Optical Flow

The exploration of two frame extraction techniques aimed to find the most effective way to capture the crucial motions in a video through a series of frames. The optical flow method was hypothesised to be a superior technique for its ability to capture frames with the most significant motion, unlike uniform sampling which often missed crucial parts of the skateboard trick, capturing more frames before or after the trick execution. An example of this behaviour can be observed in Figure 5.1. Furthermore, experiments showed an increase in accuracy when using the optical flow method over uniform sampling, further supporting this hypothesis.

This study employed Farneback's algorithm [**farneback2003two**] using the OpenCV (cv2) library, to estimate the optical flow magnitudes on each frame. The selection process involved reading pairs of consecutive frames from a video, resized for faster processing and converted to greyscale to minimise noise caused by colour variations. The cv2 function `cv2.calcOpticalFlowFarneback()` performed dense optical flow estimation between these frames, using parameters such as pyramid scale, levels, winsize, iterations that can be found in Appendix

The function `cv2.cartToPolar()` converted the Cartesian flow vectors returned by the optical flow function into polar coordinates, discarding the directional information, retaining only the magnitudes. Finally, the code computed the average

magnitude and appended it to a list. This process iterated through all frames to select those with the highest average to be extracted from the video.



(a) Extracted frames using optical flow method.

(b) Extracted frames using uniform sampling.

Figure 5.1 Comparison of frame extraction between optical flow method and uniform sampling

5.3 Dataset Split

This study opted for splitting the original dataset into 70% 20% and 10% for training, validation and testing sets respectively. The training set was used by the models to learn the patterns and releationships from the input data. The validation set enabled is a portion of the dataset used to measure the performance after every epoch. Finally, the model was evaluated using the testing set.

5.3.1 Data Augmentation

This research explored a number of augmentation techniques using the `ImageDataGenerator` library provided by Tensorflow. This process involved defining a number of parameters that influenced the augmentation patterns applied to each frame. In order to expand the number of training samples, this study created multiple copies of the original dataset, each augmented with its own set of unique parameters controlling image shifts, brightness, scaling and other transformations. Figure 5.2 shows an example of an augmented duplication of a trick sequence against its original, split into six frames for illustrative purposes. Experiments revealed that certain augmentation techniques such as image flipping and large rotational shifts disrupted the visualisation of the trick, negatively impacting model performance, thus these were removed from the augmentation parameters.

(a) Augmented frames.

(b) Original frames.

Figure 5.2 Comparison of Augmented sequence against original

5.4 Feature Extraction and Preprocessing for Training

After successfully extracting the most significant frames from each video using optical flow, the next step involved preparing the data for further processing. The main algorithm consisted of an iterative process responsible for resizing and normalising all frames before being fed to the pre-trained CNNs for feature extraction. The `cv2.resize()` function resized all frames to 224x224 to satisfy the VGG and ResNet input image requirements, before performing min-max normalisation on each frame by diving the pixel intensities values by 255.

Once resized and normalised, The pre-trained CNNs extracted features from each frame. With the final classification layers removed, the extracted features of a singular image using VGG returned a shape of $(1 \times 7 \times 7 \times 512)$, while ResNet50, resulted in a shape of $(1 \times 7 \times 7 \times 2048)$. In both cases, the outputs were flattened using the Tensorflow `flatten()` function to transform the extracted features into a one-dimensional vector suitable to be fed into the dense layers of the network. Thus, VGG generates a final feature shape of (20×25088) for an entire video, while ResNet50 generates (20×100352).

The NumPy function `np.save()` saved these extracted features along with their respective labels for training, validation, and test sets. This allowed for easy retrieval of the pre-processed data, eliminating the need to re-extract features with every new session.

5.4.1 PCA Dimensionality Reduction

This research, utilised the Principle Component Analysis (PCA) dimensionality reduction technique which was crucial in refining the feature sets produced by the

pre-trained models. PCA a statistical technique that utilises orthogonal transformation to convert observations of variables that may be correlated into a set of unrelated variables called principle components. These principle components are reorganised in a way that the first few components contain most of the important information [**dimensionality_reduction**].

5.5 Hyperparameter Optimisation

The architectures described in this study represent the best versions of themselves after a number of iterations and optimisations. To speed up the time consuming task of hyperparameter tuning, this research leveraged the power of Optuna, a hyperparameter optimisation framework [**optunaFramework**]. Optuna leverages a Bayesian optimisation algorithm that runs through a number of iterations, observing the performance of past configurations and strategically selecting promising parameter combinations such as learning rate, dropout rate and number of nodes per layer. This iterative approach enables Optuna to navigate the search space and converge on optimal hyperparameters for a specific model and dataset. This library was integrated within a custom-built development environment specifically designed to identify the optimal hyperparameter configurations for multiple model architectures and their corresponding input data.

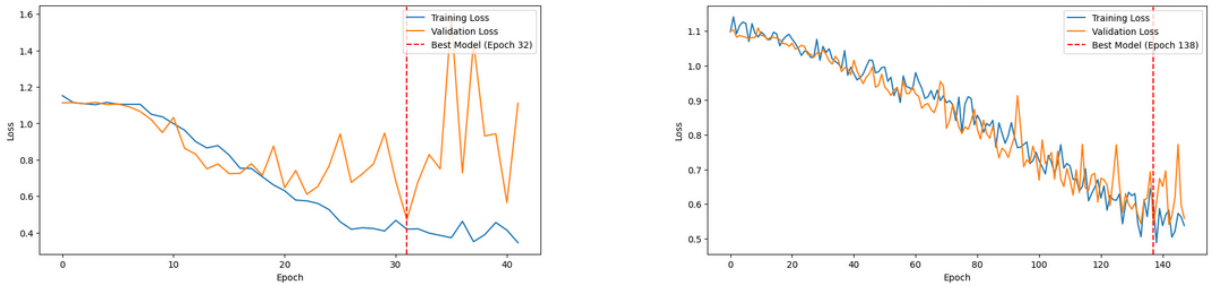
5.6 Training History

Figure shows the training and validation loss curves for the models that achieved the best validation accuracy within each architecture. Additionally, the vertical dashed lines indicate the epoch at which the early stopping callback restored the models weights, preventing overfitting.

It is important to note that the models employed categorical-crossentropy loss function during training. This function measures the discrepancies between the predicted probabilities and the actual one-hot encoded labels. Furthermore, the output layers utilised softmax activation functions to align them with the one-hot encoded labels.

This study compared the results of Stochastic Gradient Descent (SGD) and Adam [**adam_paper**] as optimisation algorithms during training. The results suggest that SGD may be more effective at reducing overfitting compared to Adam. This improvement is illustrated in Figure 5.3, using the VGG-BiLSTM architecture. While, Adam initially shows promise with good generalisation, it quickly encounters a pitfall, where the validation loss begins to climb, indicating overfitting. In contrast, the

SGD-trained model exhibits a more stable validation loss, likely due to its algorithmic stability and its capability to achieve small generalisation errors as explained in the study by Hardt et al. (2016) [**SGD_stability_paper**]. The results observed with the VGG-BiLSTM aligns with the insights discussed by Hardt et al. suggesting that models trained using SGD are less vulnerable to overfitting.



(a) Training and Validation Loss using Adam

(b) Training and Validation Loss using SGD

Figure 5.3 Model loss graphs for (a) Adam and (b) SGD optimisers.

5.7 Callbacks

5.7.1 Early Stopping

This research employed an early stopping callback to reduce overfitting and save computational resources. This method monitored the validation loss at every epoch and halted training if improvement stopped after a predetermined patience value. This study investigated a patience of 10 epochs, based on the observation that the models were unlikely to improve after 10 epochs, with no validation loss advancements.

5.7.2 Model Checkpoint

This study incorporated model check pointing in the training process to save intermediate models after every epoch. This implementation was configured to monitor the validation loss and only save the model when it showed an improvement, allowing a seamless resumption of training in case of interruption.

5.8 Baseline Model Approach

This research, made use of a baseline model approach during development of the artefact. The establishment of a baseline model allowed comparisons to be made with each iteration of the models. The first baseline model

6 Implementation

6.1 Experiments

7 Sample A

8 Sample B