2019

Multi-Sensory Deep Learning Architectures for Slam Dunk Scene Classification

Paul Minogue

Technological University Dublin

Multi-Sensory Deep Learning Architectures for Slam Dunk Scene Classification

OLLSCOIL TEICNEOLAÍOCHTA
BHAILE ÁTHA CLIATH
DUBLIN
TECHNOLOGICAL
UNIVERSITY DUBLIN

Paul Minogue

A dissertation submitted in partial fulfilment of the requirements of
Technological University Dublin for the degree of
M.Sc. in Computing (Data Analytics)

16 June 2019

'Statisticians, like artists,
have the bad habit of falling in love
with their models.'

GEORGE BOX

Declaration

I certify that this dissertation which I now submit for examination for the award of MSc in Computing (Data Analytics), is entirely my own work and has not been taken from the work of others save and to the extent that such work has been cited and acknowledged within the text of my work.

This dissertation was prepared according to the regulations for postgraduate study of the Technological University Dublin and has not been submitted in whole or part for an award in any other Institute or University.

The work reported on in this dissertation conforms to the principles and requirements of the University's guidelines for ethics in research.

Signed:

Date:

Abstract

Basketball teams at all levels of the game invest a considerable amount of time and effort into collecting, segmenting, and analysing footage from their upcoming opponents previous games. This analysis helps teams identify and exploit the potential weaknesses of their opponents and is commonly cited as one of the key elements required to achieve success in the modern game.

The growing importance of this type of analysis has prompted research into the application of computer vision and audio classification techniques to help teams classify scoring sequences and key events using game footage. However, this research tends to focus on classifying scenes based on information from a single sensory source (visual or audio), and fails to analyse the wealth of multi-sensory information available within the footage.

This dissertation aims to demonstrate that by analysing the full range of audio and visual features contained in broadcast game footage through a multi-sensory deep learning architecture one can create a more effective key scene classification system when compared to a single sense model.

Additionally, this dissertation explores the performance impact of training the audio component of a multi-sensory architecture using different representations of the audio features.

Keywords: Deep learning, Artificial intelligence, Image recognition, Video classification, Audio processing, Basketball

Acknowledgements

I would like to begin by thanking my project supervisor Brian Leahy for his guidance, advice, and availability throughout the whole process. Without his efforts, this dissertation would not be what it is today.

I would also like to express my gratitude to everyone on the lecturing staff at Technological University Dublin who I have had the pleasure of studying under during my six years at the university. Without the knowledge passed on from all of you, none of this would be possible.

Finally, I would like to express my sincere thanks to my family who have supported throughout the entire process, and to all of my friends whose insistence on getting me out of the house prevented me from losing my mind.

Contents

Declaration **I**

Abstract **II**

Acknowledgments **III**

Contents **IV**

List of figures **VII**

List of tables **XI**

Table of acronyms I **XV**

Table of acronyms II **XVI**

1 Introduction **1**

1.1 Background 1

1.2 Research problem 2

1.3 Challenges & nuances 3

1.4 Research objectives 4

1.5 Research hypothesis 4

1.6 Research methodologies 5

1.7 Resources 5

1.8 Scope & limitations 6

1.9 Dissertation outline 7

2 An introduction to the game of basketball 8

2.1 History of the sport and it's organisations 8

2.2 Rules of basketball 10

3 Review of existing literature 14

3.1 Human perception based on multi-sensory information 14

3.2 Image classification 15

3.3 Video classification 20

3.3.1 Handling the temporal component of video 20

3.3.2 Frame sub-sampling 21

3.3.3 Hybrid networks 22

3.4 Audio classification 23

3.5 Multi-sensory deep learning networks 25

3.6 Class imbalance 27

3.7 Basketball analytics 30

4 Data collection & preparation 32

4.1 The Dataset 32

4.2 Video segmentation 34

4.3 Audio Spectrograms 42

5 Experiment design and methodology 48

5.1 Computer Vision Model 48

5.2 Multi-Sensory Model 55

6 Results, evaluation and discussion 60

6.1 Results 60

6.1.1 Computer Vision Model 60

6.1.2 Multi-Sensory Model 68

6.2 Model comparison and evaluation 82

6.2.1 Computer vision vs. Multi-sensory models 83

6.2.2 Viridis vs. Hot colour maps 85

6.2.3 MFCC vs. Filter banks features 87

6.3 Discussion 89

7 Conclusion 92

7.1 Problem definition & research overview 92

7.2 Experimentation, evaluation & results 93

7.3 Contributions and impact 94

7.4 Future work & recommendations 95

7.5 Closing thoughts 97

References 99

A A summary of games analysed 107

List of figures

2.1 Image of James Naismith holding an early version of a basketball and a peach basket (source:https://springfield.edu/where-basketball-was-invented -the-birthplace-of-basketball) 9

2.2 Image of basketball half court which illustrates the position of the arc and the line. 12

2.3 An example of a jump shot (left), layup (middle), and dunk (right). . . 13

3.1 Illustration of the perceived number of visual flashes plotted against the number of beeps emitted from the audio source (Shams, Kamitani, & Shimojo, 2000) 16

3.2 A simple example of a 3x3 convolution window passed over an input space with a stride of length 2 (source: https://adeshpande3.github.io/ A-Beginner%27s-Guide-To-Understanding-Convolutional-Neural-Networks -Part-2/) 16

3.3 Illustration of a sample CNN architecture (Spanhol, Oliveira, Petitjean, & Heutte, 2016) 17

3.4 Visual representation of the InceptionV3 architecture developed by (Szegedy, Vanhoucke, Ioffe, Shlens, & Wojna, 2016) 18

3.5 Visual representation of VGG19 architecture (Zheng, Yang, & Merkulov, 2018) 19

3.6 An illustration of a simple LSTM cell (Fan, Lu, Li, & Liu, 2016). In this example x_t represents the input at time step t, c_t represents the information stored in the hidden state, i_t represents input gate, o_t controls the output of the cells hidden state, h_t which is the activation function applied to c_t, and f_t represents the forget gate (Yue-Hei Ng et al., 2015) 21

3.7 A simple overview of a hybrid network model to classify soccer scenes as illustrated by Baccouche, Mamalet, Wolf, Garcia, and Baskurt (2010). 22

3.8 Visual flow diagram of MFCC and filter bank extraction process from .wav file input (Ballan, Bazzica, Bertini, Del Bimbo, & Serra, 2009) . . 24

3.9 Viridis spectrograms representing an example of 4 different types of snore sounds (Amiriparian et al., 2017): velum (left), oropharyngeal lateral walls (centre left), tongue base (centre right), & epiglottis (right) 25

3.10 Illustration of a sample multi-sensory (audio-visual) network (Tzirakis, Trigeorgis, Nicolaou, Schuller, & Zafeiriou, 2017) 26

3.11 Illustration of a sample multi-sensory (audio-visual) network (Tzirakis et al., 2017). 29

3.12 Shot map created by Goldsberry (2012). In this map the blue grids represent the 1,284 different scoring areas and the red dots illustrate shooting density within these areas. 31

4.1 Illustration of number of slam dunks per game analysed in this study. . 35

4.2 An example of 26 frames extracted from a slam dunk scene in the dataset. 37

4.3 Illustration of train, test, and validation split on dataset. 38

4.4 An example of each of the 5 image augmentations performed to the minority class data in this study: Original image (top left), flipped (top right), blurred (middle left), blurred & flipped (middle right), unsharpened (bottom left), blurred & unsharpened (bottom right). 41

4.5 Illustration .wav files for slam dunk (left) and non-slam dunk (right) scenes 42

4.6 Examples of normalised filter banks (left & middle left) and MFCC (right and middle right) spectrograms for slam dunk (top) and non-slam dunk (bottom) scenes. 47
4.7 Flow chart of process required to create spectrograms. 47

5.1 Illustration of a simple neural network node describing the role of weights within the network (source: https://skymind.ai/wiki/neural-network). . 50
5.2 Illustration of how a frame extracted from a video segment gets converted into normalised 229x229 RGB arrays. 50
5.3 Flow diagram of the video clip feature extraction process. 51
5.4 Comparison of RELU and leaky RELU activation functions. 52
5.5 High level visual summary of the computer vision model architecture. . 53
5.6 Illustration of the audio steam which makes up our multi-sensory architecture. 56
5.7 High level summary of the end to end multi-sensory model architecture. 57
5.8 Illustration of the χ^2 with 1 degree of freedom and significance level of $\alpha = 0.05$ (critical region shaded in blue). 59

6.1 Illustration of training and validation loss over training for the model described in experiment CV-1. 61
6.2 Illustration of training and validation loss over training for the model described in experiment CV-2. 64
6.3 Illustration of training and validation loss over training for the model described in experiment CV-3. 66
6.4 Illustration of training and validation loss over model training for the model described in experiment MS-1. 70
6.5 Illustration of training and validation loss over model training for the model described in experiment MS-2. 72
6.6 Illustration of training and validation loss over model training for the model described in experiment MS-3. 74

6.7 Illustration of training and validation loss over model training for the model described in experiment MS-4. 77

6.8 Illustration of updated architecture required to incorporate SVM component. 80

List of tables

1.1 Sample of average number of shots taken from each range for 4 NBA teams in the 2018/2019 season to illustrate shooting tenancies (source: stats.nba.com) 1

4.1 Full set of features described in dataset created by Ramanathan et al. (2016) 33

4.2 Summary of the dataset based on number of games analysed, segments within each game, and frames within each segment. 36

4.3 Description of features contained in the frames information dataframe created for this study. 39

5.1 Example of a contingency table in which *a* represents the number of instances Model 1 and Model 2 prediction correctly, *b* indicates the number of instances that model 1 predicted correctly which model 2 predicted incorrectly, etc. 58

6.1 Confusion matrix produced when the model developed as part of experiment CV-1 was applied to the training set. 62

6.2 Summary of model metrics from experiment CV-1 which includes precision and recall metrics for each class, MCP, MCR, TPR, TNR, and support. 62

6.3 Confusion matrix produced when the model developed as part of experiment CV-2 was applied to the training set. 63

6.4 Summary of model metrics from experiment CV-2. 63

6.5 Confusion matrix produced when the model developed as part of experiment CV-3 was applied to the training set. 65
6.6 Summary of model metrics from experiment CV-3. 65
6.7 Summary of confusion matrices produced when models described in the other computer vision experiments section were applied to the testing set. 67
6.8 Confusion matrix obtained when testing set is applied to audio stream of the multi-sensory network MS-1. 68
6.9 Confusion matrix produced when the model developed as part of experiment MS-1 was applied to the training set. 69
6.10 Summary of model metrics from experiment MS-1. 70
6.11 Confusion matrix obtained when testing set is applied to audio stream of the multi-sensory network MS-2. 71
6.12 Summary of model metrics from experiment MS-2. 72
6.13 Confusion matrix obtained when testing set is applied to audio stream of the multi-sensory network MS-3. 73
6.14 Confusion matrix produced when the model developed as part of experiment MS-3 was applied to the training set. 74
6.15 Summary of model metrics from experiment MS-3. 75
6.16 Confusion matrix produced when the model developed as part of experiment MS-4 was applied to the training set. 76
6.17 Summary of model metrics from experiment MS-4. 76
6.18 Summary of hyper-parameters used in SVM experimentation. 80
6.19 Summary of confusion matrices produced when models described in the other multi-sensory experiments section were applied to the testing set. 82
6.20 Performance comparison between models developed in experiments CV-1 and MS-2. 83
6.21 Contingency table used to calculate McNemar's test statistic when comparing models CV-1 and MS-2. 84

6.22 Performance comparison between models developed in experiments CV-3 and MS-4. 84

6.23 Contingency table used to calculate McNemar's test statistic when comparing models CV-3 and MS-4. 85

6.24 Comparison between MFCC viridis/hot multi sensory models (top) and Filter bank viridis/hot multi-sensory models (bottom). 86

6.25 Contingency tables used to calculate statistical significance for MFCC (top) and filter bank (bottom) spectrogram comparisons. 87

6.26 Comparison between MFCC/filter banks models using viridis spectrograms(top) and hot spectrograms (bottom). 88

6.27 Contingency tables used to calculate statistical significance for viridis (top) and hot (bottom) audio feature comparisons. 89

A.1 Information about each game analysed as part of this dissertation. Games can be viewed by prefixing the YouTube ID with http://youtube .com/watch?v= 107

Listings

4.1 Python/SQL code used to join slam dunk success labels from the full dataset to the subset of data used in this study. 35

4.2 Python function to extract single band audio from video clip at a sample rate of 44.1kH and bit rate of 160kbps. 43

4.3 Python function to create spectrogram of input audio using the viridis colour pallet and normaised MFCC features. 46

Table of acronyms I

Acronym	Term
NCAA	National Collegiate Athletic Association
BAA	Basketball Association of America
NBA	National Basketball Association
ABA	American Basketball Association
FIBA	Fédération Internationale de Basketball
CNN	Convolutional neural networks
ISVLRC	ImageNet Large Scale Visual Recognition Challenge
RELU	Rectified linear unit
RNN	Recurrent neural network
LSTM	Long short term memory network
MFCC	Mel-frequency cepstrum coefficients
DBN	Deep belief network
SMOTE	Synthetic minority oversampling techniques
GAN	Generative adversarial networks
SVM	Support vector machine

Table of acronyms II

Acronym	**Term**
GPS	Global positioning systems
MKV	Matroska multimedia container
PNG	Portable network graphics
ECG	Electrocardiogram
FFT	Fast Fourier transform
WAV	Waveform audio file
TN	True negative
TP	True positive
FN	False negative
FP	False positive
MCP	Mean class precision
MCR	Mean class recall
TRP	True positive rate
TNR	True negative rate

Chapter 1

Introduction

1.1 Background

Analysing footage from an upcoming opponents previous games in order to identify the teams shooting tendencies (see table 1.1) and defensive vulnerabilities has been an integral part game planning in the sport of basketball (and indeed many other sports) since the advent of televised sports. This form of analysis is employed by basketball teams from middle school to NBA level across the United States, and often times the most successful teams attribute their success to the work performed in the ”film room” preparing for their opponents.

Team	**<5 Ft**	**5-9 Ft**	**10-14 Ft**	**15-19 Ft**	**20-24 Ft**	**25-29 Ft**
Atlanta	35.4	7.4	6.7	3.9	15.3	21.7
Houston	28.6	6.8	3.9	2.0	16.9	28.2
New Orleans	35.4	7.4	6.7	3.9	15.3	21.7
Philadelphia	31.1	8.7	7.9	7.5	11.5	21.0

Table 1.1: Sample of average number of shots taken from each range for 4 NBA teams in the 2018/2019 season to illustrate shooting tenancies (source: stats.nba.com)

Despite it’s importance in the modern game, collecting and annotating footage is still very much a manual process for teams which involves a human with sufficient domain knowledge watching each game, identifying when an event occurs, before noting the start and end time for the event. This is a very time consuming task for teams and eats into valuable time which could be spent drawing insights from the footage.

This would appear to be an ideal use case for a computer vision model to analyse and classify scenes from game footage. While this is certainly a valid and exciting application, this type of model is only capable of analysing the visual component of the videos and throws away the abundance of important information contained within the audio such as commentator and crowd reactions.

This idea is easy for us to relate to as more than likely we can all recall a time in which we were watching a sporting event, and perhaps due to occlusion[1], or a bad camera angle we were not able to see clearly if a team has scored, or an event has occurred. Instead, our brain relied on the audio information from the broadcast to form a more coherent perception of what was taking place on screen.

As a result of examples such as this, we propose that modern sports video segmentation systems should not rely on information from a single sensory source. Rather these systems should behave similarly to our brains and combine information from multiple sensory sources in order to identify scenes of interest more effectively.

1.2 Research problem

In recent years there has been a considerable amount of research into the application of machine learning/artificial intelligence for key event classification in sports such as soccer, tennis, and basketball. While this research is valid, and has employed many creative and interesting techniques to tackle this problem, for the most part these studies focused on models which analyse information from a single sensory source (video *or* audio).

The primary research problem associated with this dissertation is to determine if a multi-sensory (video & audio) architecture can yield a more effective key event [2] clas-

[1] When the view of a person or object of interest becomes obstructed or obscured by other objects.

[2] This research problem will focus solely on the area of slam dunk classification within basketball broadcast footage.

sification system when compared to a similar single sense (video) model. A secondary research problem within this dissertation aims to asses the impact of different audio feature representations on the overall model performance.

1.3 Challenges & nuances

Sports video segmentation and key event classification is an interesting implementation of computer vision and audio classification. While in many ways it can be considered an extension of traditional AI driven surveillance systems (D'Orazio & Leo, 2010), there are a number of nuances associated with the data contained within broadcast sports footage which make this an intriguing, and challenging application.

Perhaps the most common challenge posed by this application is the relatively low quality audio/visual footage presented by sports broadcasts (Niu, Gao, & Tian, 2012) when compared to that which could be collected within a more controlled environment. In addition to this, the high rates of occlusion which occur between players wearing the same clothing is a phenomena common to sports video which is not prevalent in traditional surveillance systems. Finally, dataset imbalance is a challenge within key event classification as typically scenes of interest occur much less often than they do not.

Despite these challenges, there are some characteristics of sports broadcast footage which enable the use of computer vision and audio classification in sports. One enabler is the wealth of information available from how crowds react to certain sequences within a basketball, or soccer match. However, what often gets overlooked is the fact that sports can be considered a semi-controlled environment whereby the boundaries and rules of the sport limit the movements/actions which players can perform and significantly reduces the set of actions our models need to understand (Kristan, Perš, Perše, & Kovačič, 2009).

The challenges and nuances described above suggest that the application of computer vision and audio analysis in sports scene classification is an incredibly exciting, if challenging research problem.

1.4 Research objectives

The primary objective of this dissertation is to demonstrate that by analysing the wealth of audio and visual features available within sports broadcast footage simultaneously through a multi-sensory deep learning architecture, one can yield more effective key scene classification results when compared to a single sense model.

Secondary to this, the architectures developed as part of this study will aim to provide a building block for future research into the application of multi-sensory deep neural networks for more advanced scene classification within the sport of basketball, and indeed across different sports.

These objectives will be achieved through completion of the milestones laid out below:

- Performing a comprehensive review of existing literature relating to the sport of basketball, the human sensory system, and the fields of image, audio and video classification.
- Identifying, acquiring, and preparing an appropriate dataset for this study.
- Developing a baseline computer vision model.
- Extending the baseline model to a multi-sensory (audio-visual) model.
- Performing a statistical comparison of the models performance on a testing dataset.
- Evaluating the proposed multi-sensory solution and discussing potential improvements to the architecture for future use.

1.5 Research hypothesis

Of course, in order to test our proposed solution formally in accordance with the scientific method, we must define our research hypothesis in a clear, predictive, and testable statement. The research hypothesis for this dissertation is defined as follows:

H_0: A multi-sensory hybrid network model comprising of CNN and LSTM components which analyses both visual and audio features extracted from broadcast basketball footage simultaneously ***will not*** classify the occurance of slam dunks with significantly greater mean class precision, mean class recall, true positive rate, and true negative rate than a single sense hybrid network model comprised of CNN and LSTM components which analyses only the visual features extracted from broadcast basketball footage.

H_a: A multi-sensory hybrid network model comprising of CNN and LSTM components which analyses both visual and audio features extracted from broadcast basketball footage simultaneously ***will*** classify the occurance of slam dunks with significantly greater mean class precision, mean class recall, true positive rate, and true negative rate than a single sense hybrid network model comprised of CNN and LSTM components which analyses only the visual features extracted from broadcast basketball footage.

1.6 Research methodologies

This study will follow a deductive approach, beginning with the research hypothesis defined above in section 1.5. From here, we will use quantitative methods to analyse the audio and visual features contained within our data before finally gathering empirical evidence (mean class precision/recall, true positive rate and true negative rate metrics) to test the feasibility of our proposed solution.

1.7 Resources

In order to achieve the milestones and objectives discussed in section 1.4, we utilised the following technical resources:

- Desktop PC: Intel Core i5-7500 CPU @ 3.4GHz, 8GB RAM, NVIDIA GeForce

GTX 1050 Ti graphics card, running Windows 10

- Python 3.6
- TensorFlow-GPU v1.10
- LaTeX/MiKTeX 2.9 & TeXstudio 2.12.6
- Cloud backup storage

1.8 Scope & limitations

The scope of this dissertation will be restricted to the classification of slam dunk scenes from broadcast basketball game footage. These scenes will be classified using a multi-sensory deep learning architecture trained using audio and visual features extracted from the footage of 10 basketball games played between the years 1992 and 2009. The models developed as part of this study will not attempt to identify or classify any other events which occur during these games other than a slam dunk.

Perhaps the biggest limitation for this study is the quality of the footage contained within our dataset. As mentioned above, some of the footage for the 10 games analysed in this dissertation was recorded as far back as 27 years ago. This is obviously a limitation as we understand that the recording quality at that time was far inferior to what is capable with modern recording equipment. It is fair to assume that the relatively poor recording quality could hinder any models trained on this data, however this is something we will attempt to overcome by sub-sampling an appropriate number of frames from our video footage and fine tuning our model parameters.

Another limitation of this study, which is alluded to in section 1.3 is the heavy amount of class imbalance contained within the dataset. Since this study is focused on the classification of slam dunk scenes, which is the least common type of shot in the game of basketball, we are expecting to experience a considerable amount of imbalance within the dataset. However, we will look to employ techniques such as oversampling and SMOTE to overcome this imbalance.

It is also a fair to question whether or not footage from just 10 games is enough data to train a deep learning architecture on. Our reason for choosing these games in particular revolve around the class imbalance discussed above, as well as the timeline available to conduct this study. These reasons are discussed in more detail in chapter 4.1

Finally, given that the analysis will be performed on the desktop machine described above (section 1.7) we will be limited with regard to how long we can train our models for. Unlike a server environment, it will not be feasible to train a model over a number of days, or weeks. We will look to overcome this limitation by utilising pre-trained CNN architectures where possible.

1.9 Dissertation outline

Chapter two of this dissertation will provide a brief introduction to the game of basketball. This chapter will provide a history of the game and it's organisations before explaining the basic rules and terminology of the sport to provide readers with the baseline domain knowledge required to follow this study.

Chapter three will review some of the existing literature related to this study. This review will touch on areas such as image, audio and video classification, as well as methods for dealing with class imbalance. We will discuss the various methods and techniques proposed in the literature and how they pertain to our problem.

In chapter four we will discuss the dataset and the various pre-processing required to get the data into a format we could use for developing our models. From here, in chapter five we will discuss our proposed architectures and how they are evaluated.

In chapter six we will discuss the results of our experimentation before formally testing our research hypothesis. Finally, in chapter seven we will conclude on the findings and learnings of this study in addition to advising on areas for further research going forward.

Chapter 2

An introduction to the game of basketball

This chapter provides a brief introduction to the game of basketball. In this chapter we discuss the origins of the sport, its organisations, and it's cultural impact before talking through the basic rules of the game.

This information will provide the reader with a baseline understanding of the sport which will in turn aid them in their understanding of the terms and ideas discussed in this dissertation.

2.1 History of the sport and it's organisations

The sport of basketball was created in 1891 by James Naismith in Springfield, Massachusetts. The game was originally created to provide students of the local college with a sport could play to keep fit during the winter months while popular field sports such as football and lacrosse were in their off seasons. While the sport created in 1891 is quite different to what we see on our TV's today, the basic premise of the game still remains the same, for teams to throw a ball (originally a soccer ball) into a (peach) basket in an effort to accumulate points and win the game.

By the end of the 19th century basketball had spread to a handful of colleges across the United States of America (USA). In 1909 the game had become so popular that

Figure 2.1: Image of James Naismith holding an early version of a basketball and a peach basket (source:https://springfield.edu/where-basketball-was-invented-the-birthplace-of-basketball)

it became an officially regulated sport by the National Collegiate Athletic Association (NCAA), the governing body for amateur collegiate sports in the USA. Thirty years later, in 1939 the inaugural NCAA Men's Basketball Tournament took place. This competition was an 8 team single game elimination style [1] tournament and was won by the University of Oregon Ducks who defeated the Ohio State University Buckeyes in the tournament final by a scoreline of 46 to 33.

The NCAA Men's Basketball Tournament has grown in size and popularity over the years and is viewed by millions of people across North America every year. The most recent NCAA Men's Basketball Tournament Championship Game drew an attendance of over 72,000 spectators who witnessed the Virginia Cavaliers defeat the Texas Tech Red Raiders by a scoreline of 85 to 77 [2].

The increasing popularity of the game through the 1940's paved the way for the establishment of a large scale professional basketball league. The Basketball Association of America (BAA) was established in 1946, however it wasn't until the BAA changed it's name to the National Basketball Association (NBA) in 1949 that the league began to take off. The 1949 NBA season was contested by 17 teams in a tra-

[1] The winner of each game advances to the next round

[2] A review of the game courtesy of the New York Times is available at https://www.nytimes.com/2019/04/08/sports/ncaa-tournament-virginia-texas-tech.html

ditional league followed by playoff format with the championship being won by the Minneapolis Lakers (now known as the Los Angles Lakers).

As the popularity of the sport continued to trend upwards through the 1950's and 1960's it gave rise to other professional leagues such as the American Basketball Association (ABA). The ABA was established in 1967 and consisted of 11 teams. The ABA's ability to attract rising stars and future NBA Hall of Fame players [3] such as Julius Erving and Moses Malone paired with its innovative rules and competitions (such as the introduction of the 3 point line and annual slam dunk competition) made the ABA a legitimate competitor to the long established NBA. After 9 season the ABA was able to successfully force a merger with the NBA to create a larger, and more talent-rich professional league in North America.

The emergence of stars such as Michael Jordan, Shaquille O'Neal, Kobe Bryant and Allen Iverson from the late 1980's all the way through to the early 2000's helped propel the sports popularity to unprecedented levels and solidify it's place in American pop culture. Today, the NBA is viewed by millions of people across North America every week. The sport's popularity globally has led to the establishment of a world governing body for basketball known as FIBA (Fédération Internationale de Basketball) and the sports inclusion in the Summer Olympic Games.

2.2 Rules of basketball

At it's most basic level, basketball is a team sport in which 2 teams of 5 players compete against each other to accumulate points by putting the ball through their opponents basket. The team who manages to accumulate the most points at the end of the game (4x12 minute quarters in the NBA, or 2x20 minute periods in the NCAA) is deemed to be the winner.

While in possession, a player can advance the ball by passing it to another player with their hands, or by bouncing the ball off the ground while moving in an act known

[3] A list of NBA Hall of Fame Inductees can be found at https://www.nba.com/history/hall-of-fame-inductees

as a dribble. Once a player has stopped dribbling the ball they are allowed take a maximum of 2 steps without dribbling before being called for a travelling foul. In addition to this, once a player has stopped dribbling the ball, they are not permitted to dribble again until they release the ball by taking a shot or passing it to another player (referred to as a double dribble). While a player cannot take more than 2 steps after completing their dribble, they may retain control of the ball without moving for as long as the shot clock [4] permits.

For each standard score (commonly referred to as a bucket) inside the ”arc” (illustrated in figure 2.2), the scoring team receives 2 points. However, if the team manages to score from beyond the arc, the shooting team will receive 3 points. If it is deemed that the a member of the defending team fouled the attacking player (through illegal personal contact) while in the act of shooting the attacking player will be sent to ”the line”, a position on the court 15 feet away from the basket for 2 [5] free throw attempts. In this scenario the fouled player gets a chance to shoot 2 uncontested shots from the line in which each successful shot is worth 1 point.

While many variations exist, basketball shots can be grouped into 3 distinct categories (see figure 2.3 for an example of each shot type):

- Jump shot: This is the most common type of shot in a game of basketball and can be performed anywhere on the court. This shot involves the player in possession jumping into the air and releasing the ball above their head with a high arc towards the basket. NBA Miner [6] calculated that jump shots were successful 38.56% of the time during the 2018/19 NBA season.

- Layup: This shot involves the player in possession leaping towards the basket and laying/bouncing the ball off the backboard before letting it fall into the basket for a score. Due to the nature of this shot it may only be performed at

[4] A 24 second countdown timer indicating how long the team in possession has before they must take a shot.

[5] A player may receive 3 free throws if the shot attempt was taken from beyond the arc, or 1 free throw attempt if they score despite being fouled.

[6] A collection of detailed NBA stats from every game as far back as 1996.

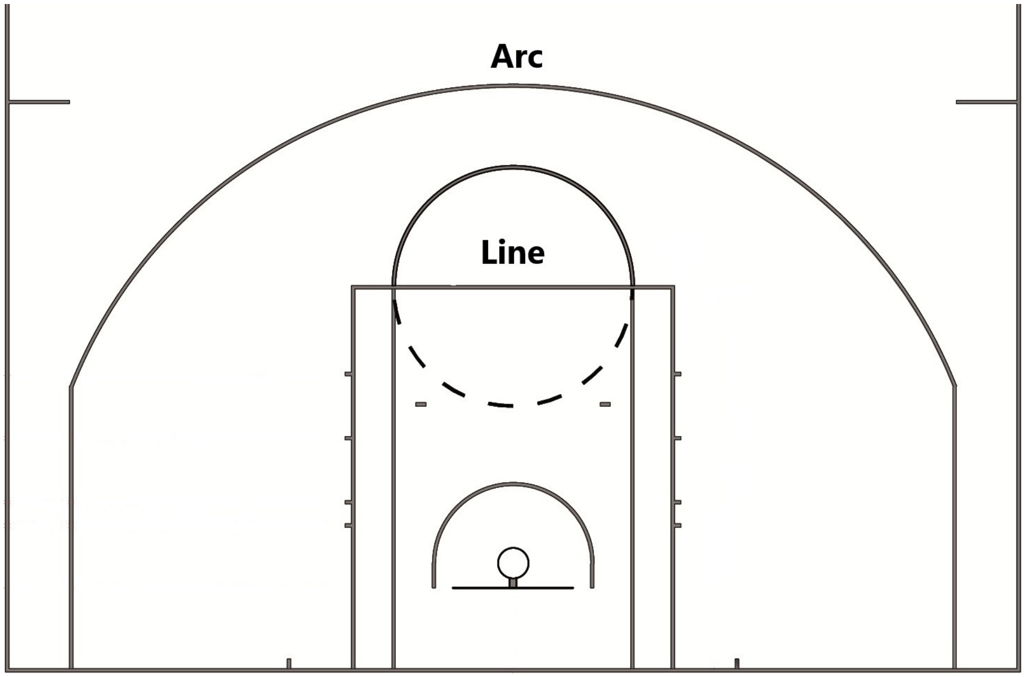

Figure 2.2: Image of basketball half court which illustrates the position of the arc and the line.

a close proximity to the basket. As a result of this the shot success rate on a layup is greater than that of the jump shot, with layups being successful 55.5% of the time throughout the 2018/19 season.

- Slam dunk: This involves the player in possession leaping into the air towards the basket and placing the ball into the basket. While this is a high percentage shot with a success rate of 89.82%, the shot requires significantly more space to execute than the jump shot or layup and as a result is much rarer occurrence in game situations. NBA Miner's data suggests that slam dunks made up just 5.44% of total shots in the NBA in 2018/19. As a result of its rarity, and the emphatic nature of the shot, the slam dunk is the most popular shot type from a fans perspective. The occurrence of a slam dunk typically draws a loud response from the crowd and is the subject of many highlight packages on YouTube.

Finally, although basketball is (incorrectly) considered by many to be a non-contact sport, players are permitted to engage in a certain amount of physical contact with members of the opposing team. An example of this is the act of setting a screen. A screen is where a member of the attacking team attempts to use their body to block the path of a player from the opposing team. In this case the player setting the screen may physically block the opponents path providing they are not moving, and do not lean towards the opposing player when doing so. This technique is commonly employed at the highest level of the game to clear space for a player is possession to perform one of the shot types mentioned above.

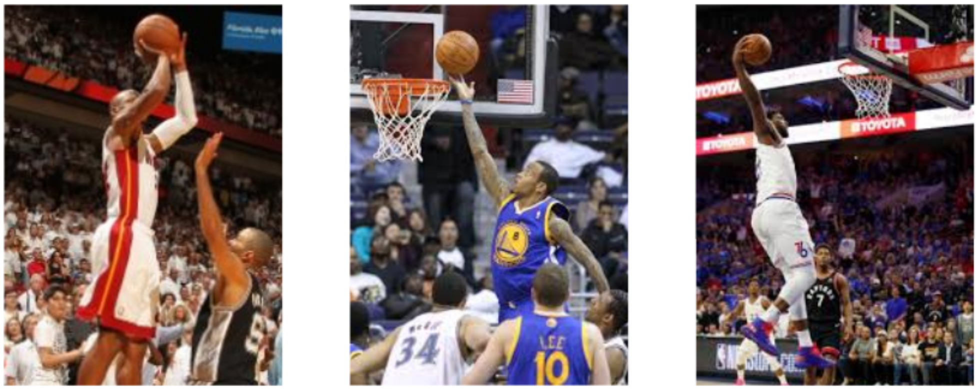

Figure 2.3: An example of a jump shot (left), layup (middle), and dunk (right).

Chapter 3

Review of existing literature

This chapter summarises and compares previous studies into areas related to the research objectives of this dissertation. It will discuss the methods used in these studies and how useful they could be in the context of this dissertation.

3.1 Human perception based on multi-sensory information

As humans, we process information from multiple sensory sources to form a robust perception of the world around us and in turn, make decisions on how to act based on this information. In the past, perception had been thought of as a modular function in which each of our different sensory systems acted independently to each other (Shimojo & Shams, 2001). However, intuitively this idea seems flawed when we consider the simple real world example of sitting on a stationary train looking out the window at a neighbouring train. In this example, if one of the trains starts to move our visual system often struggles to identify which train is in motion. However, once our brain combines the visual information with information from the vestibular system [1], it can quickly identify which train is in motion (Ernst & Bulthoff, 2004). Examples such as this have not only inspired this study, but have also inspired an enormous amount

[1] A biological system which provides our brains with information about motion

of research into the relationships between different sensory systems and how these systems work together to enable our brains to understand the world around us.

Studies such as Newell, Bulthoff, and Ernst (2003) designed a simple set experiments to analyse the relationship between visual and haptic (touch) sensory systems to demonstrate empirically that humans perform better at object recognition tasks when combining information from both sensory systems when compared to performance using a single sensory system (either visual **or** haptic). Similarly Campos, Butler, and Bulthoff (2012) demonstrated that humans perform better at judging the distance travelled when combining both visual and vestibular (walking down a corridor while wearing a VR headset) information when compared to visual information alone.

Conversely Shams et al. (2000) illustrated how the human brain combines audio and visual information together when forming a perception of the world by cleverly demonstrating how audio information can alter the perception of an unambiguous visual stimulus. To illustrate this, each subject was asked to count the number of flashes produced from a single light source (a white disk) while also being fed audio information in the form of beeps from an audio source. The results (illustrated in figure 3.1) showed that the subjects perception of the number of flashes outputted from the light source was correlated to the number of beeps coming from the audio source.

3.2 Image classification

Convolutional neural networks (CNNs) are commonly employed in computer vision tasks such as image recognition due to their aptitude for efficiently reducing a complex image input to a compressed feature map (Krizhevsky, Sutskever, & Hinton, 2017). CNNs were originally based on the mathematical idea of convolution which is defined as an integral that expresses the overlap as one function passes over another (Weisstein, 2003). However, in the field of computer science and deep learning this idea of convolution can be thought of as iteratively analysing small portions of an input space (e.g. 3x3 pixels across an input image) through what is known as a convolution

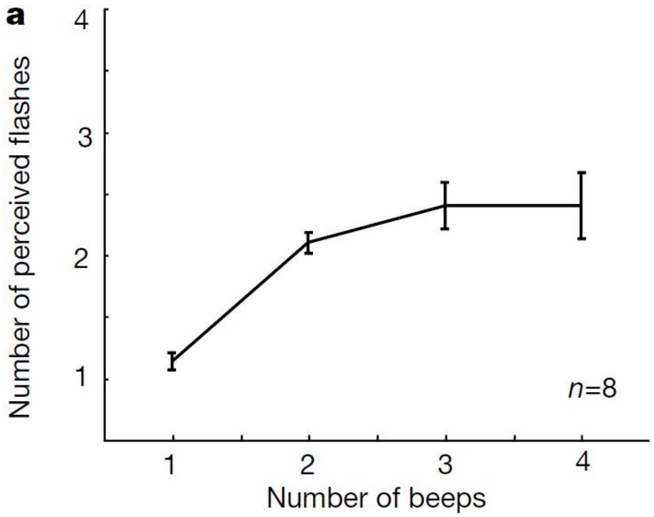

Figure 3.1: Illustration of the perceived number of visual flashes plotted against the number of beeps emitted from the audio source (Shams et al., 2000)

window to extract local features (edges, colours, etc.) within the image. These local features are further contextualised within the image (globally) by merging adjacent feature maps in the space through a process called pooling (Piczak, 2015).

Elements of the CNN such as the size of the convolution window (3x3, 5x5, etc.) and the stride (the step size of the convolution window across the input) can be adjusted to manage the size and complexity of the feature map. Figure 3.2 illustrates a simple example of convolution applied to a 7x7 input space.

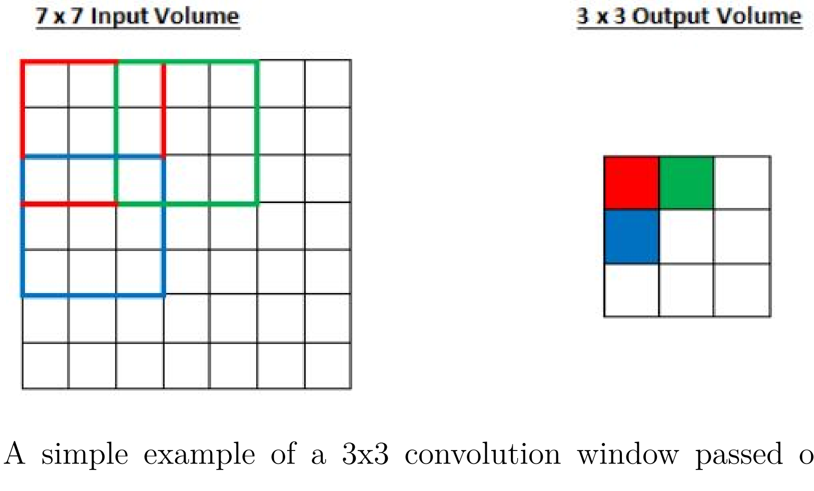

Figure 3.2: A simple example of a 3x3 convolution window passed over an input space with a stride of length 2 (source: https://adeshpande3.github.io/A-Beginner%27s-Guide-To-Understanding-Convolutional-Neural-Networks-Part-2/)

In practice, CNN layers are integrated with a number of different layer types to form a deep CNN architecture. These architectures typically consist of an input layer (an image for example), a number of convolutional and pooling layers, a dense fully connected layer, and an output layer (Piczak, 2015). Different versions and configurations of these architectures have been shown to demonstrate best in class performance at a variety of image recognition tasks such as facial attribute recognition (Zhong, Sullivan, & Li, 2016) or breast cancer classification from histopathological [2] images (Spanhol et al., 2016). Figure 3.3 illustrates a typical CNN deep learning architecture.

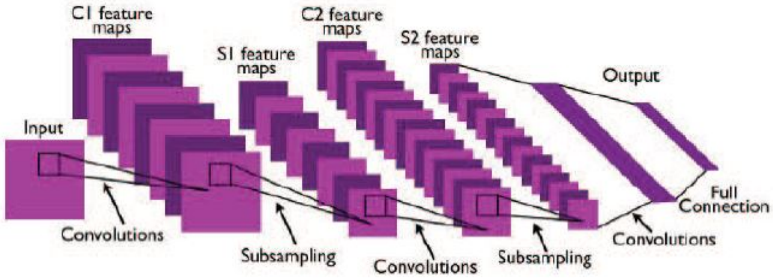

Figure 3.3: Illustration of a sample CNN architecture (Spanhol et al., 2016)

As a result of the increased processing power of modern computers (Simonyan & Zisserman, 2015), and the availability of large labelled image datasets, there have been significant advances in the classification power of CNN architectures across research domains by leveraging complex CNN models which have been pre-trained large datasets of labelled images (Krizhevsky et al., 2017) in what is commonly referred to as transfer learning. Due to the complexity of the networks and the volumes of diverse image data on which they are trained, these networks display best in class performance when it comes to identifying and extracting local features from input images. These models are widely used in image classification activities as they can reduce the amount of time required to develop and train a computer vision model while still achieving state of the art performance.

The InceptionV3 model is a popular CNN architecture used for transfer learning. This architecture is based on the GoogLeNet model (Szegedy et al., 2015) and is

[2] More information on the field of histopathology can be found at: https://www.rcpath.org/discover-pathology/news/fact-sheets/histopathology.html

known for its ability to map input images to a 2,048 dimensional feature space. The Inceptionv3 architecture, illustrated in figure 3.4 is a complex network consisting of 48 layers. Despite this complexity however, the network is more computationally efficient than other state of the art pre-trained networks. The model achieves this computational efficiency by avoiding extreme compression early in the network, increasing activations per layer and by balancing the width and depth of the network (Szegedy et al., 2016). The model has shown state of the art performance on the ImageNet Large Scale Visual Recognition Challenge (ISVLRC) with a top 1 error rate of 17.2% and has set a new state of the art benchmark when it comes to the ISVLRC dataset (Szegedy et al., 2016).

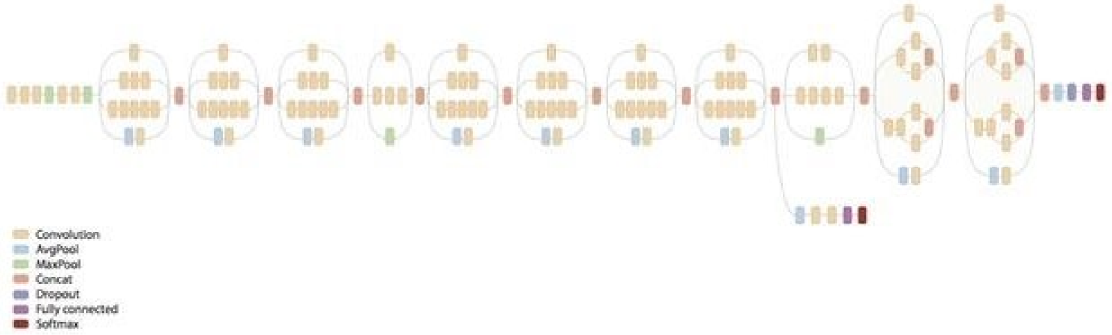

Figure 3.4: Visual representation of the InceptionV3 architecture developed by (Szegedy et al., 2016)

While the InceptionV3 architecture is a popular choice when it comes to transfer learning, there are a variety of other pre-trained architectures which are capable of state of the art performance such as the VGG19 model. This model is recognised for it architectural simplicity, containing only 19 layers as well as it's computational cost (Szegedy et al., 2016) when compared to other models such as the InceptionV3. Despite this architectural simplicity, the model has shown state of the art results on the ISVLRC dataset (24.7% top 1 error), this is achieved though the use of small convolution windows (3x3) and rectified linear unit (RELU) activation functions across all layers of the network (Simonyan & Zisserman, 2015).

As a result of their success on the ISVLRC dataset, both the InceptionV3 and

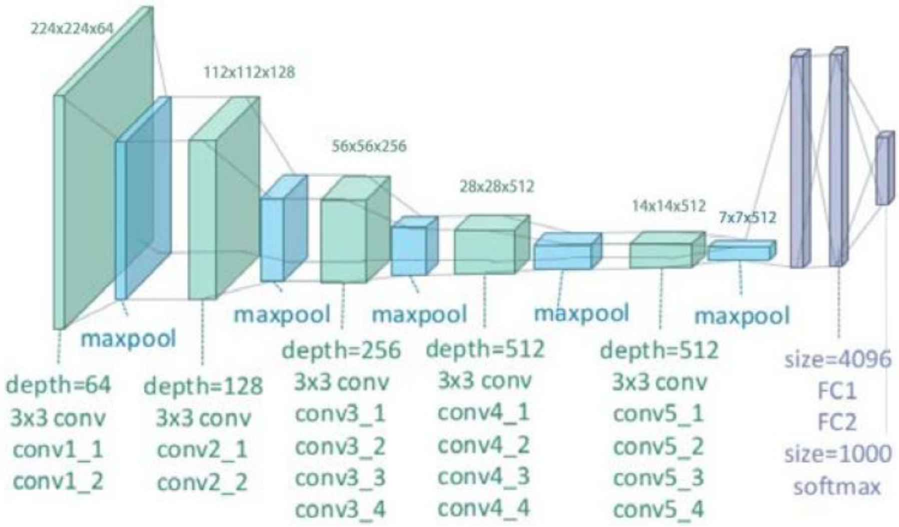

Figure 3.5: Visual representation of VGG19 architecture (Zheng et al., 2018)

VGG19 architectures are commonly employed on a range of image and audio (see section 3.4) classification tasks. For example, Esteva et al. (2017) demonstrated the power of the InceptionV3 architecture when retraining the network to diagnose conditions such as Melanoma and Carcinoma from with similar accuracy to that of a dermatologist. Similarly Carvalho, De Rezende, Alves, Balieiro, and Sovat (2017) used the VGG19 architecture to classify computer generated images of humans based on local features extracted from the eyes within the image.

Despite both models demonstrating state of the art performance when it comes to image recognition tasks, the increased computational cost required by the VGG19 model compared to the InceptionV3 architecture in addition to the superior performance of the InceptionV3 model on both ISVLRC and ImageNet datasets forces us to consider favouring the InceptionV3 architecture for this particular task, especially given the hardware limitations and complexity of the input.

3.3 Video classification

3.3.1 Handling the temporal component of video

Unlike images, videos are much more difficult to collect and annotate (Karpathy et al., 2014). As a result of this, there is a scarcity of large scale labelled video datasets on which we can develop pre-trained architectures. However, video classification can be considered an extension of traditional image classification, whereby a series of static images (frames) are tied together by a temporal component. While one can still use CNNs to analyse the static images, traditional CNNs have a major limitation in that they can only analyse spatial information and not temporal (Fan et al., 2016).

In order to analyse the temporal component of video recordings one can utilise a form of recurrent neural network (RNN) known as a long short term memory network (LSTM). The LSTM (illustrated in figure 3.6) was originally developed to overcome the back propagation through time problem which occurred in traditional RNN architectures whereby the error signals passed back through the temporal component of the network either blew up or vanished completely (Hochreiter & Schmidhuber, 1997). This problem was overcome with an novel (Hochreiter & Schmidhuber, 1997), and subsequently refined (Gers, Schmidhuber, & Cummins, 1999) gradient descent approach to enforce constant error flow through the network.

This constant error flow is achieved through hidden state which stores, modifies, and passes information within the network for an arbitrary amount of time (Fan et al., 2016), allowing the network to understand long-term temporal relationships within the data (Yue-Hei Ng et al., 2015). At each time step an input gate controls what new information enters the hidden state while a forget gate determines what existing information should be released from the hidden state.

As a result of their ability to store and understand long term temporal relationships, LSTMs and other variations such as gated recurrent units (Cho et al., 2014) have become a popular choice for modelling data which evolves over time [3].

[3] An excellent blog post regarding RNNs/LSTMs can be found at: https://colah.github.io/posts/2015-08-Understanding-LSTMs/

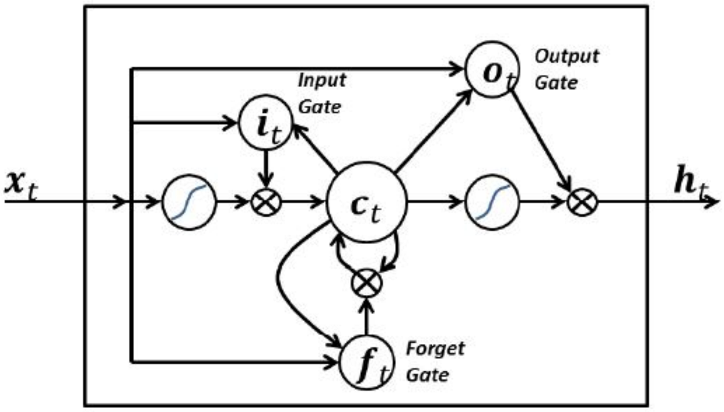

Figure 3.6: An illustration of a simple LSTM cell (Fan et al., 2016). In this example x_t represents the input at time step t, c_t represents the information stored in the hidden state, i_t represents input gate, o_t controls the output of the cells hidden state, h_t which is the activation function applied to c_t, and f_t represents the forget gate (Yue-Hei Ng et al., 2015)

3.3.2 Frame sub-sampling

Analysing and classifying video sequences can be very computationally demanding due to the large number of frames included in each video (Yue-Hei Ng et al., 2015). For example, a 4 second clip shot at 40 frames per second will create 160 individual frames. It is easy to see how this form of data can quickly inflate a dataset and the computational power required to analyse it. In order to overcome this problem, many state of the art approaches to video classification involve uniformly sub-sampling a pre-defined number of frames per second from the raw video input. Thoughts on the number of frames per second to sub-sample from input video varies across the research. For example Ramanathan et al. (2016) sub-sampled 6 frames per second from their 4 second long input clips. Conversely, Yue-Hei Ng et al. (2015) chose to sub-sample 1 frame per second from each of their 5 minute long input videos.

Given the fact that this study is focused on identifying scenes in which slam dunks occur, we will be focusing on short time frames. With this in mind, we fee; that the 1 frame per second approach proposed by Yue-Hei Ng et al. (2015) although computationally efficient, would not capture enough information about each scene to build an effective classifier for this study.

3.3.3 Hybrid networks

Once the frame sub-sampling process has been complete, the typical state of the art approaches involve analysing the extracted frames in a hybrid network model (illustrated in figure 3.7). These models are typically comprised of a number of CNN layers which create a feature map from each of the sub-sampled frames. These ”bottleneck” feature maps are then passed in sequential order through an LSTM architecture which analyses the evolution of the features over time before producing a final classification for the original video clip. Hybrid network architectures have demonstrated best in class performance for tasks such as human emotion recognition (Fan et al., 2016), sport classification (Karpathy et al., 2014) and tennis action recognition (Chow & Dibua, 2018).

This hybrid network approach makes sense intuitively as it combines strengths of CNNs and their ability to map an image input to a feature map as well as LSTMs, and their ability to analyse long term temporal relationships into a single model architecture rather than trying to force an LSTM to analyse raw image data over time, which is simply not feasible.

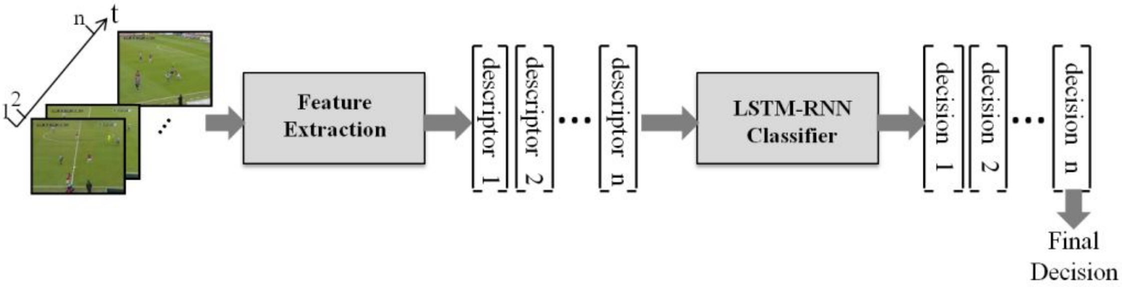

Figure 3.7: A simple overview of a hybrid network model to classify soccer scenes as illustrated by Baccouche et al. (2010).

3.4 Audio classification

Prior to analysing any audio signals using deep learning (or any other methods), it is important to understand that there are a number of preprocessing steps which can be applied to the raw audio frequencies in order to increase the classification power of any models built on top of the data. The first of these preprocessing steps typically applied is a technique known as downsampling. This involves reducing the sample rate of the raw audio frequency in an attempt to reduce the computational power required to analyse the audio. Typically raw audio files come with a sample frequency of 44.1kHz, these frequencies are commonly downsampled to 32kHz, 22.05kHz or 16kHz prior to analysing (Boddapati, Petef, Rasmusson, & Lundberg, 2017).

The second preprocessing step that is often performed to audio input data is to transform the raw audio input into filter banks or mel-frequency cepstrum coefficients (MFCCs). The filter banks transformation uses a set of triangular filters to decompose the audio signal into uniform frequency sub-bands with a bandwidth much lower than the original signal (Afonso, Tompkins, Nguyen, & Luo, 1999). Once the filter bank representation of the audio has been calculated, we can easily calculate the MFCC representation by simply applying a discrete cosine transform to the logarithim of the filter bank representation (Davis & Mermelstein, 1980).

It is worth noting that while there are a lot of studies which utilise either filter banks **or** MFCC's when analysing audio data, to the best of our knowledge there is a lack of research which explicitly compares the performance of these two methods when applied to real world data. This comparison is something that has peaked our curiosity while performing this literature review, and is something we explore in this study. Figure 3.8 illustrates the filter bank and MFCC extraction process from a raw audio input file.

With regard to application of deep neural networks for audio classification, the literature suggests that there are two common approaches. The first approach is to analyse the audio in it's raw/preprocessed waveform (typically extracted from a .wav file) using either a CNN, RNN/LSTM, or a deep belief network (DBN). Aytar,

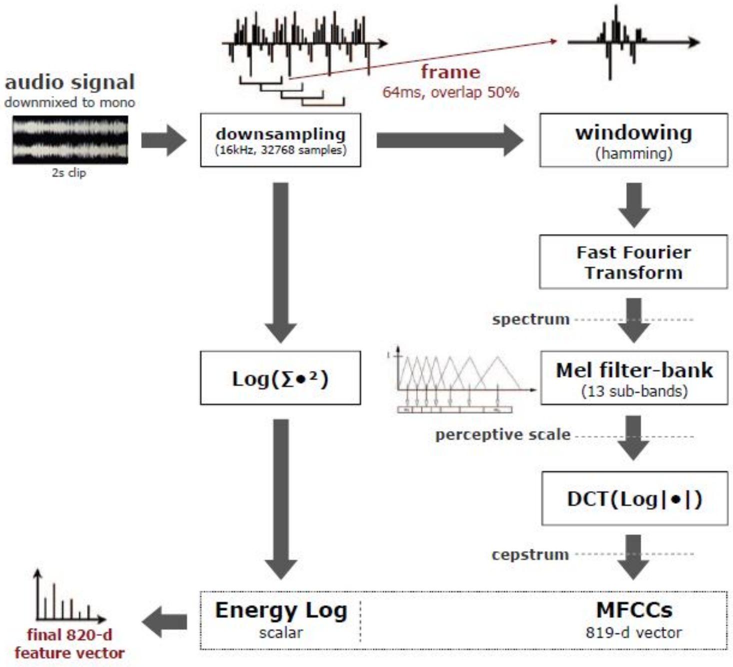

Figure 3.8: Visual flow diagram of MFCC and filter bank extraction process from .wav file input (Ballan et al., 2009)

Vondrick, and Torralba (2016) employed this approach to analyse and classify raw audio waves from over 2 million unlabelled video clips using CNNs. They citied using CNNs to reduce the feature space that the model is required to analyse, in addition their ability to handle variable length inputs. Conversely, Ballan et al. (2009) ustilised DBNs (a generative probabilistic model) and MFCCs to detect events which occur in soccer videos based on the audio features alone.

The second approach involves converting the audio signals into an image (known as a spectrogram). From here, one can leverage pre-trained CNN architectures to classify the visual representation of the audio. This approach is popular due to the classification power of pre-trained CNN models. This approach has been utilised effectively in tasks such as snore sound classification using bottleneck features [4] extracted from both GoogleNet and VGG19 (Amiriparian et al., 2017), environmental sound classification (Boddapati et al., 2017), and music genre classification (Costa, Oliveira, Koericb, &

[4] A pooled feature map of the input generated using deep neural networks, can be thought of as a method for dimension reduction (Yu & Seltzer, 2011)

Gouyon, 2011).

Intriguingly, it has also been suggested that the colour map used when converting filter bank or MFCC features into an audio spectrogram can impact the classification power of a model built on the data. Amiriparian et al. (2017) demonstrated this when they observed that models trained on viridis colour map spectrograms (see figure 3.9) performed significantly better than those trained on colour maps such as grey and jet.

While overall analysing the raw waveform may prove to be more computationally efficient, the amount of time required to develop and refine a network to classify this data may be significant. Given the reasonably short amount of time available to conduct this study, converting the audio features to spectrograms and leveraging one of the tried and tested CNN architectures discussed in section 3.3 may be a more feasible approach. Converting the audio to spectograms would also allow us to compare the performance of the viridis colour map against another colour and potentially further validate the results observed by Amiriparian et al. (2017).

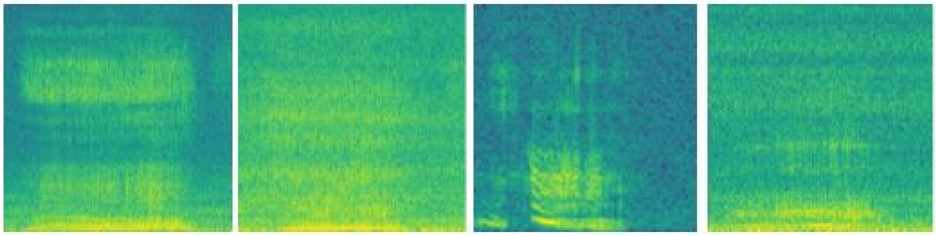

Figure 3.9: Viridis spectrograms representing an example of 4 different types of snore sounds (Amiriparian et al., 2017): velum (left), oropharyngeal lateral walls (centre left), tongue base (centre right), & epiglottis (right)

3.5 Multi-sensory deep learning networks

Although deep learning architectures have demonstrated state of the art performance when it comes to scene classification based on audio or visual features, there has been an increasing amount of research into further pushing the boundaries of this performance through the use of multi-sensory (sometimes referred to as multi-modal) networks.

Multi-sensory deep learning networks (illustrated in figure 3.10) analyse features from multiple modalities which describe the same event (audio and visual information extracted from a video for example). These networks process features from each modality through separate networks before merging features from these together in a process know as fusion. It has been suggested that the inclusion of audio information can increase the accuracy of video classification systems by ∼3% (Fan et al., 2016).

The stage of the model in which the fusion process takes place varies across the research performed in this area. Studies such as Simonyan and Zisserman (2014) into human activity recognition using static and temporally connect images as well as Fan et al. (2016) into video based emotion recognition using audio and video features perform a class score fusion at the end of their architectures. This type of fusion merges the outputs from final layer of each modal network and produces a final prediction based on an aggregation (maximum/mean class score) of the outputs from each model.

Conversely Tzirakis et al. (2017) propose fusing the models once feature extraction has taken place (prior to any class scores being calculated), from here the merged feature maps can be passed into a separate network (LSTM in this case) and output a final class score based on the LSTM output (see figure 3.10).

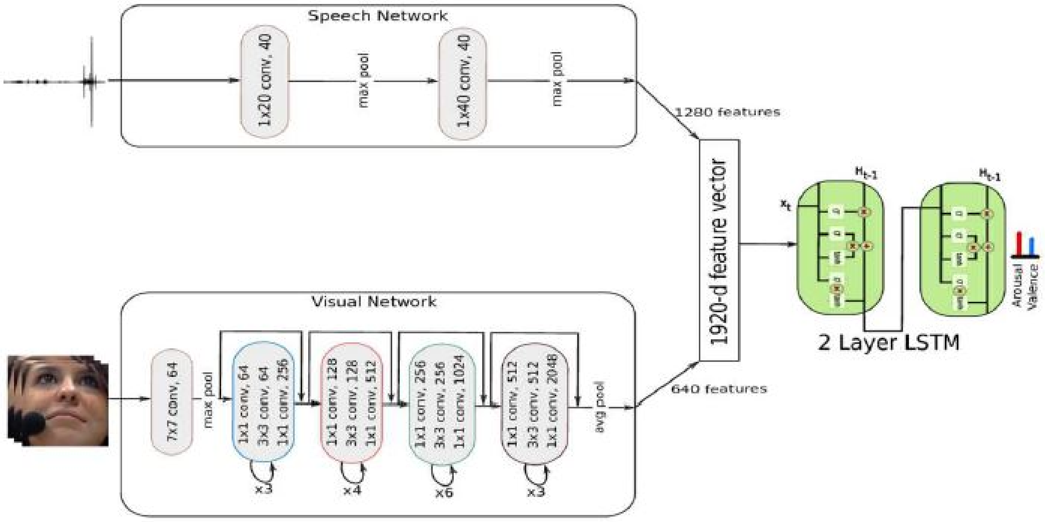

Figure 3.10: Illustration of a sample multi-sensory (audio-visual) network (Tzirakis et al., 2017)

Without having access to results which compare the performance of these two methods, it is difficult identify which of these methods is more effective, and which could demonstrate superior performance in this study. On the one hand, the class score fusion approach makes sense intuitively as we simply combine the predictions from each model/sensory system into a fusion architecture which produces the final classification for the model. However, if like in our case the output from each sensory system is low dimensional, it could be difficult for the fusion layer to model this data effectively and could potentially lead to underfitting. On the other hand, the pre-class score fusion makes less sense intuitively perhaps, but could provide more information to the fusion component and help avoid underfitting.

Researching both of these methods certainly gave us a lot to think about and forced us to consider employing both methods when conducting this study in an attempt to maximise our chances of developing an effective multi-sensory classifier.

3.6 Class imbalance

Class imbalance is a common problem in machine learning, it's presence in training data can be detrimental to the classification power of any models built on top of the dataset (Buda, Maki, & Mazurowski, 2018). While there has been extensive research into this problem within the context of traditional machine learning (Batista, Prati, & Monard, 2004), there is still relatively limited research knowledge available relating to the problem of class imbalance in the domain of deep learning and image classification (Buda et al., 2018).

This lack of research does not imply that class imbalance is not a problem within the field of deep learning, image, and audio classification. Many interesting applications of CNN architectures such as marine image classification in which habitats can feature high-biodiversity and low species density (Langenkamper, van Kevelaer, & Nattkemper, 2018), or the diagnosis of rare medical conditions using images (Esteva et al., 2017) illustrate the need for robust methodologies for handling dataset imbalance within the context of image and audio classification.

Thankfully these methodologies do not have to be developed from scratch and we as deep learning practitioners can leverage many of the tried and tested methods for handling class imbalance from traditional machine learning. The simplest examples of this involve using weighted loss functions when training deep neural networks (Sudre, Li, Vercauteren, Ourselin, & Cardoso, 2017) and oversampling minority classes. It should be noted however that in order to reduce the risk of overfitting in deep neural networks, it is suggested that oversampling should be performed to an extent that completely eliminates the class imbalance within the dataset (Buda et al., 2018).

Other methods for eradicating class imbalance involve using synthetic minority oversampling techniques (SMOTE) such as image augmentation. Image augmentation involves creating new images (or videos) by blurring, rotating, illuminating and flipping the original training images (illustrated in figure 3.11). This process works under the assumption that images and videos for the most part are translationally invariant. For instance an image of a man/woman standing beside a tree which is flipped on its horizontal axis is still an image of a man/woman standing beside a tree. This approach has been applied in applications such as marine image (Langenkamper et al., 2018) and plant classification (Pawara, Okafor, Schomaker, & Wiering, 2017). In both studies the use of image augmentation was shown to reduce the risk of models overfitting the training data. This approach appears to be directly applicable to our problem as a video of a slam dunk flipped on the horizontal axis is still easily recognisable as a slam dunk to the human eye.

It is also worth noting that there is research being performed into the interesting and exciting area of generative adversarial networks[5] (GANs) and their potential as an alternative, and superior approach to synthetic oversampling when compared to simple image augmentation (Douzas & Bacao, 2018). However, currently this technology is not at the level of sophistication required to be viable option for overcoming class imbalance in this dissertation.

While not commonly employed in the field of image/video/audio classification,

[5] A neural network architecture which is capable of generating new data with the same characteristics of the data which it is trained on (Goodfellow et al., 2014)

Figure 3.11: Illustration of a sample multi-sensory (audio-visual) network (Tzirakis et al., 2017).

studies have explored the use of one class classifiers using neural networks. In short, one class classification involves developing models which are trained solely on one class from the training data. A simple example of this would be building a website recommender system for a web user based on their search history (L. M. Manevitz & Yousef, 2001), in this case the training data consists solely of sites the user has visited in the past and not the ones they have not (i.e. no negative samples). One class support vector machine (SVM) classifiers are commonly used in applications such as anomaly detection (Pauwels & Ambekar, 2011).

Studies into the application of one class classifiers within the domain of deep learning have focused primarily on the use of auto-encoders[6]. These studies have shown that novel deep learning methods outperform than SVM and Naive-Bayes based approaches when it comes to tasks such as retrieving documents of interest (L. Manevitz & Yousef, 2007) and handwriting outlier detection from the MNIST dataset (Chalapathy, Menon, & Chawla, 2019).

Although a one class classifier could be considered a novel approach to solving our research problem since a slam dunk is a relatively rare event in a game of basketball. However, treating slam dunk identification as an anomaly detection problem would mean that the models and methodologies developed as part of this dissertation would be limited in their ability to scale towards classifying more common events such as jump shots and layups. Since we would like the models developed in this dissertation to provide a basis for future, more advanced scene classification going forward, we will not explore the use of one class classifiers.

[6] A form of deep learning architecture for which the models output is simply an approiximaton of its input (i.e. $x_i \approx x_o$)

3.7 Basketball analytics

Advances in global positioning systems (GPS) and player tracking technologies in the NBA/NCAA in recent years have led to the widespread availability of spatial data about each player which had previously been unavailable or was too time consuming to collect. The availability of this data has aided researchers in developing advanced metrics which more accurately quantify a player's performance on the court.

Goldsberry (2012) demonstrated that using this data one could develop novel metrics such as spread (spatial spread of scoring positions for a player across the court) and range (effective shooting range of a player across the court) in order to identify which players exhibit the most potent spatial shooting behaviours. These metrics are derived from a players success at the different shooting positions across the court (illustrated in figure 3.12). Conversely Goldsberry and Weiss (2013) utilised player spatial data to develop a defensive rating system (citing the lack of metrics to effectively measure a players defensive performance). This rating system involved analysing how players perform within 5 feet of the basket as well as how much they influence the success of shots taken in their proximity.

In addition to the development of advanced metrics, there has been a lot worked performed into testing the validity of phenomena such as the ”hot hand”. The hot hand is a belief held by many that a player is more likely be successful on a shot attempt following a series of previously successful shots. Studies such as Gilovich, Vallone, and Tversky (1985) have analysed successive shot (and free throw) patterns from a number of NBA teams to determine that there is no empirical evidence to suggest the hot hand exists despite 91% of subjects ($n = 100$) who were interviewed as part of their study believing in the idea. With this result, Burns (2001) uses simulations to suggest that if the problem is looked at from an adaptive thinking point of view the idea of the hot hand can be used as an effective method to allocate shots among a team citing the fact that streaks are predictive of a players shooting percentage.

Other areas of research relating the sport of basketball which interested us while reviewing the existing literature include:

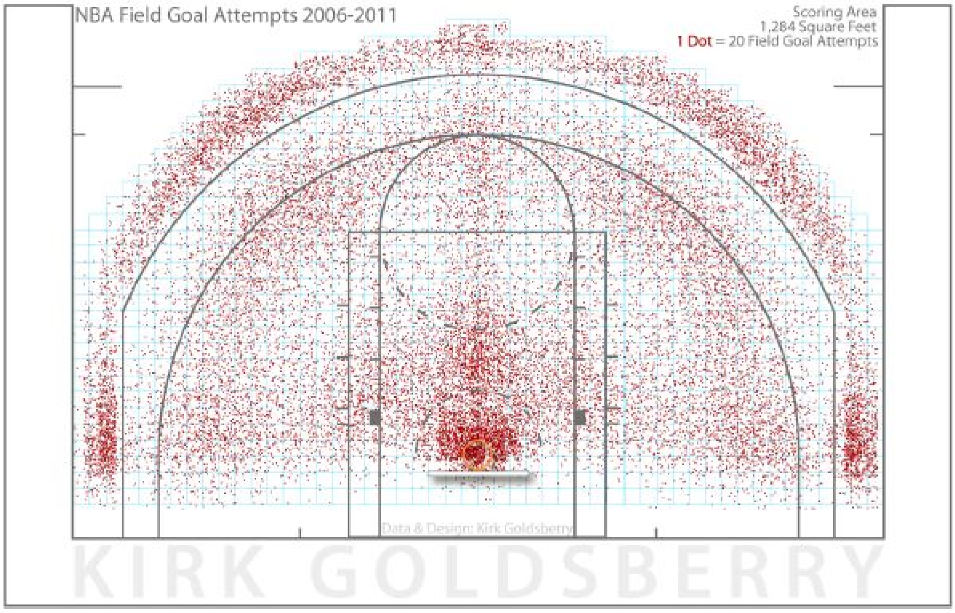

Figure 3.12: Shot map created by Goldsberry (2012). In this map the blue grids represent the 1,284 different scoring areas and the red dots illustrate shooting density within these areas.

- Using integer programming and enumerative techniques to optimise and automate the scheduling of games in the NCAA (Nemhauser & Trick, 1998; Henz, 2001).
- Analysing whether the position at which player is selected in the NBA draft [7] has an impact on the amount of playing time that player receives, and the length of their NBA career regardless of their production (Staw & Hoang, 1995).
- Exploring how some of the computer vision techniques discussed above can be used to identify key participants in events that occur on the count (Ramanathan et al., 2016)
- Classifying NBA scenes of interest based on audio recordings (Mi & Xue, 2018).

While much of this research is not related to our study specifically, it was important and indeed inspiring for us to explore just how useful analytics can be within the sport of basketball and understand some of the key research areas within the sport.

[7] An annual event where the top college prospects are selected by NBA teams

Chapter 4

Data collection & preparation

This chapter begins by discussing the dataset analysed as part of this dissertation. From here, we walk through the steps required to obtain and prepare both the audio and visual information for analysis before finally discussing the methods employed to overcome the class imbalance which existed within the dataset.

4.1 The Dataset

The dataset used to develop the models described in this dissertation is a subset of the NCAA Basketball dataset [1] created and used by Ramanathan et al. (2016). This dataset contains manually annotated event labels[2] for eleven key events (such as slam dunk success/failure, layup success/failure, free throw success/failure, steal, etc) which occur in 257 NCAA Basketball games played between different teams at different neutral venues.

For each key event identified in the dataset the annotators took note of features such as the YouTube ID of the source video, the time (in milliseconds) that the event occurred within the video, and the ball location at the start of the event. The full list of features captured in the NCAA Basketball dataset are described in Table 4.1.

[1] The dataset is available for download at: http://basketballattention.appspot.com/bball_dataset_april_4.csv

[2] Events were labelled using Amazon Mechanical Turk: https://www.mturk.com/

The game videos listed in the original dataset were roughly 90 minutes long and are available to view in full on YouTube.

Feature	**Description**
YouTubeId	The YouTube ID of the full game video. The video can be accessed by prefixing this ID with http://youtube.com/watch?v=
VideoWidth and VideoHeight	The height and width of the video in pixels
ClipStartTime and ClipEndTime	The timestamp (in milliseconds) of the key event clip which Ramanathan et al. (2016) used when training their model(s)
EventLabel	Slam dunk success/failure, steal success, 3-pointer success/failure, etc
EventStartTime	The timestamp (in milliseconds) in which the event begins (e.g. when a player begins his shooting motion)
EventStartBallX and EventStart-BallY	The X and Y coordinates of the ball position (as a fraction of the video height and width) with the origin in the top left corner
EventEndTime	The timestamp (in milliseconds) at which the event ends (e.g. when the ball enters the hoop on a successful slam dunk)

Table 4.1: Full set of features described in dataset created by Ramanathan et al. (2016)

The subset of data analysed as part of this study is limited to the 10 games in which the most slam dunk successes occur. The reasons for limiting the data to these 10 games are twofold: Firstly, slam dunks are a relativity rare occurrence in a game of basketball as they typically require a lot of space to execute (as discussed in section 2.2). As a result of this, we must understand that our dataset (which is only interested in the occurrence of a successful slam dunk) is going to be imbalanced and that this imbalance is going to have a negative impact on any classification models we look to develop. While focusing on this subset wouldn't solve the imbalance problem alone (discussed more in section 4.2), it would prevent us from further adding to the imbalance by analysing games where little or no slam dunks occur.

Secondly, given the short time frame available to perform this study, it was determined that training a multi-sensory hybrid network model on the full 257 game dataset would require a significant amount of time and would result in less time being available to refine, optimise, and compare the model(s). While it was acknowledged that analysing only a subset of the data would more than likely result in reduced classification power, the significant time savings allowed us to focus our efforts on experimenting with different architectures and fine tuning the parameters.

Once the 10 videos containing the most successful slam dunk occurrences had been identified (using a simple table aggregation in Python), these games were downloaded from YouTube using the pytube[3] Python library and stored in .mkv format. Figure 4.1 illustrates the number of slam dunks which occur in the 10 games chosen for this study

4.2 Video segmentation

Prior to analysing the game footage it was necessary to segment each of the full length games into a series of n second long scenes. Based on our review of the existing literature and our understanding of the data, we decided to segment each game into a series of 4 second long clips. We felt that 4 seconds was long enough to give some context as to what is going on within a scene while still being in a localised timeframe (Ramanathan et al., 2016).

Once this decision was made, we utilised a Python wrapper for FFMPEG [4] to

[3] Library documentation: https://python-pytube.readthedocs.io/en/latest/

[4] A command line tool for audio and video processing

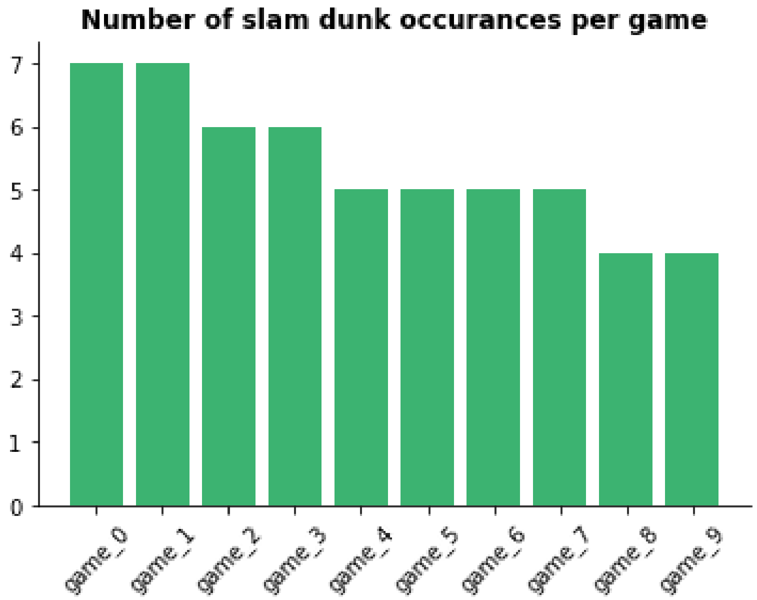

Figure 4.1: Illustration of number of slam dunks per game analysed in this study.

segment each full game video into a series of 4 second long clips [5].

In order to keep track of our clips (and to help us with sub-sampling these into a series of frames), we created a data frame which contained information on the source video location, the 4 second clip location, clip start time and end time (in microseconds). From here we were able to join the 'slam dunk success' labels from the original dataset (described in section 4.1) on the slam dunks EventEndTime between our clip start time and clip end time. We then used this to create a binary slam dunk indicator column. Listing 4.1 demonstrates the join performed in Python/SQL code.

```
import pandas as pd
import pandasql as ps

sqlcode = '''
select seg.*
  ,case when og_dataset.EventLabel is null then 0
    else 1 end as slam_dunk_indicator
```

[5] Note that the final scene in each video was less than 4 seconds long, however these were excluded from our analysis

```
from segmented_video_info seg

left join mapping_table map
  on a.game = b.game_id

left join ncaa_df og_dataset
  on map.YoutubeID = og_dataset.YoutubeID
  and seg.start_time <= og_dataset.EventEndTime
  and seg.end_time >= og_dataset.EventEndTime'''

segmented_video_info = ps.sqldf(sqlcode,locals())
```

Listing 4.1: Python/SQL code used to join slam dunk success labels from the full dataset to the subset of data used in this study.

Upon segmenting the full game videos into a collection of 4 second long clips, it was then necessary to sub-sample these clips into a series of frames which would be analysed by the CNN component of the computer vision models. After experimenting with a number of different frames per second to determine which one gives the most context to the event occurring in each clip while also keeping our data set size manageable, we settled on extracting 26 frames from each video.

These 26 frames included the first and last frame from each clip which we deemed were important in the case where a slam dunk occured at the very end of one clip or the start of another, in addition to 6 frames per second sampled uniformly across each of the 4 seconds of video footage as proposed by Ramanathan et al. (2016). Using the Python FFMPEG wrapper, we iteratively extracted these frames and stored them in .PNG format. An example of frame sub-sampling on a slam dunk scene is illustrated in figure 4.2.

Number of games	**Number of video segments**	**Number of frames**
10	11,835	307,710

Table 4.2: Summary of the dataset based on number of games analysed, segments within each game, and frames within each segment.

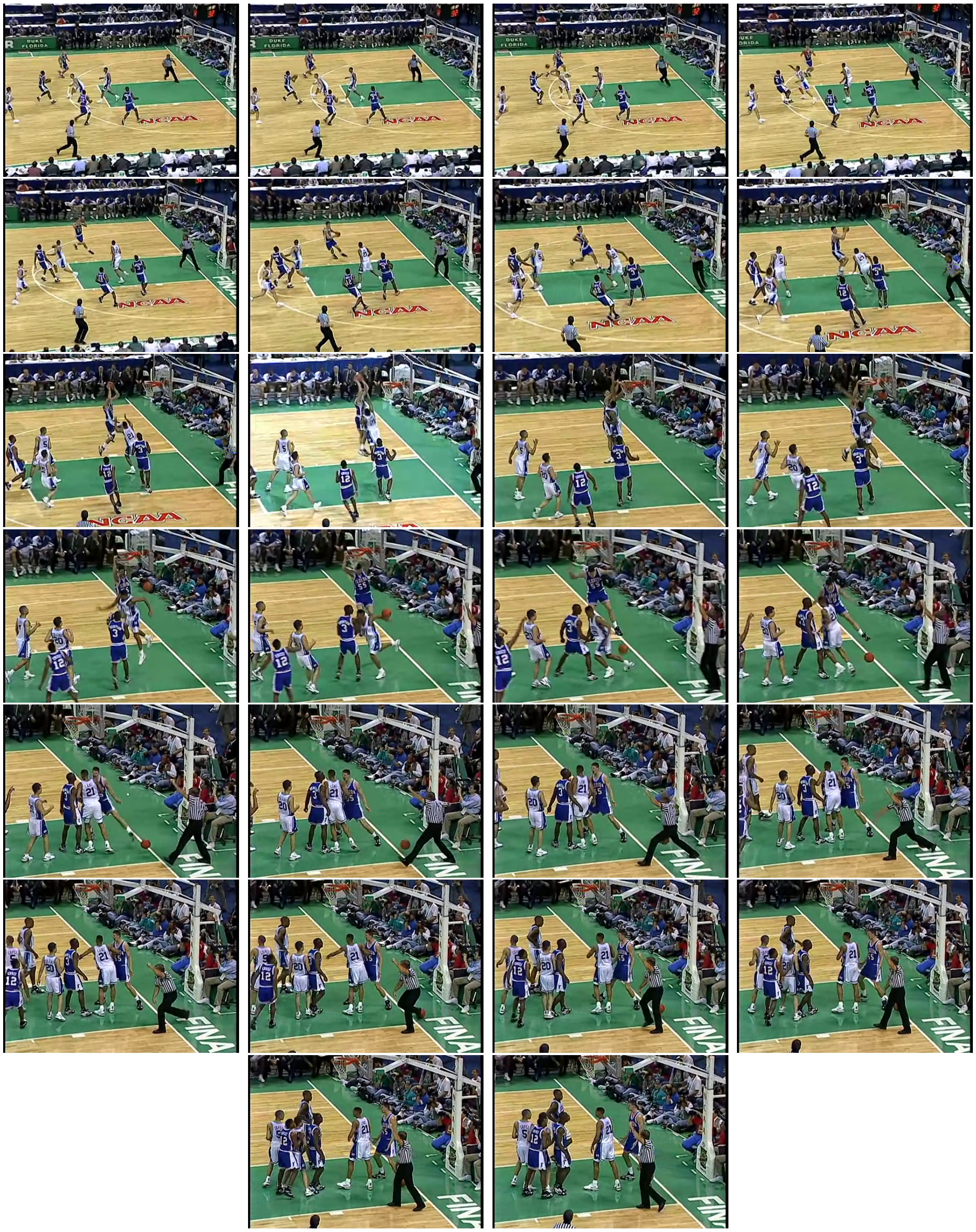

Figure 4.2: An example of 26 frames extracted from a slam dunk scene in the dataset.

Once again, to keep track of the sub-sampled frames we created a data frame which contained information on each frames location, source video, event label, etc. This data frame would provide the basis for training our computer vision models and the vision component of our multi-sensory models. The full list of features within this data frame are detailed in table 4.3

In order to further combat the class imbalance discussed in section 4.1 we decided that it was necessary to introduce additional positive slam dunk samples into our dataset. Rather than downloading and parsing through additional data, or synthetically generating additional samples using a generative adversarial network (which would be an entire research project in itself), we decided to introduce additional samples by augmenting the current slam dunk frames.

Prior to augmenting the images, it was important to first divide our dataset into train, test and validation partitions. It was vitally important to ensure that this split was done prior to introducing any synthetic samples to our dataset as we were only interested in augmenting the images within the training set. The reason for this is that if we augmented the images prior to splitting the data into train, test and validation sets, it is likely that different version of the same clips would end up in both the training and test sets. If this occurred, it would have meant that our models were being trained and tested on variations of the same data. Such an occurrence would yield misleading test results and would invalidate any findings made as part of this dissertation.

When it came to splitting the dataset into train, test and validation partitions, we began by obtaining a list of distinct 4 second long video clips and their corresponding slam dunk indicator. Next, using Python we applied stratified random sampling to split the clips up into training (60%), validation (10%), and testing (30%) datasets (see figure 4.3). Once we identified which clips belonged to our training, validation and testing sets we simply updated our frames information table (see table 4.3) to include a ”data_subset” column that indicated which partition each set of frames belonged to.

Train 60%
Test 30%
Val 10%

Figure 4.3: Illustration of train, test, and validation split on dataset.

Once the dataset was partitioned into training, validation and testing sets, we were able to begin augmenting the minority class images within the training set. To

Feature	**Description**
frame_location	The location on the drive of the extracted frame (in .PNG format)
frame_number	A number between 1 and 26 indicating the order of the frame within the sequence of extracted frames.
game	Which of the 10 games analysed the frame is part of (game_0,game_1,etc.)
source_video_location	The location on drive of 4 second clip the frame is extracted from
start_time	The timestamp (in milliseconds) that the 4 second clip starts within the original game footage
end_time	The timestamp (in milliseconds) that the 4 second clip ends within the original game footage
slam_dunk_indicator	A binary value indicating whether or not the frame belongs to a 4 second clip which contains the occurrence of a slam dunk (1 implying a slam dunk has occurred in this clip).
data_subset	One of train, test or validation.

Table 4.3: Description of features contained in the frames information dataframe created for this study.

do this, we followed a similar process to that suggested by Langenkamper et al. (2018). We began by flipping each image along the horizontal axis to create a ”backwards” version of each frame. We were comfortable doing this, as opposed to flipping the images vertically since an image of a slam dunk flipped on the horizontal axis is still easily recognisable as a slam dunk whereas the same image flipped vertically is not recognisable as a slam dunk. Using this simple augmentation we were able to double the number of positive class samples in our dataset.

From here we performed two additional augmentations to each of our positive class frames:

- Firstly, we applied a Gaussian blur[6] to each image using the Python ImageFilter module with a blur radius of 2.
- Secondly, we applied a digital unsharp masking to each image. This is a technique whereby a Gaussian blur version of the image is subtracted from the original image to create a new image.

Both of these techniques alter each of our slam dunk frames enough that our model could interpret them as new images, while still preserving many of the key features which are present in slam dunk scenes (e.g. a player hanging from the hoop).

Following these augmentations we were able to increase the number of positive samples in our training set from 32 (0.4% of the training dataset) to 192 (2.6% of the training dataset). While this increase was encouraging, we were still concerned that the amount of imbalance which remained in the dataset was a problem and was something we would need to address with oversampling, as suggested by Buda et al. (2018).

[6] More infotmation and examples of Gaussian blurs can be found at: https://www.sciencedirect.com/topics/engineering/gaussian-blur

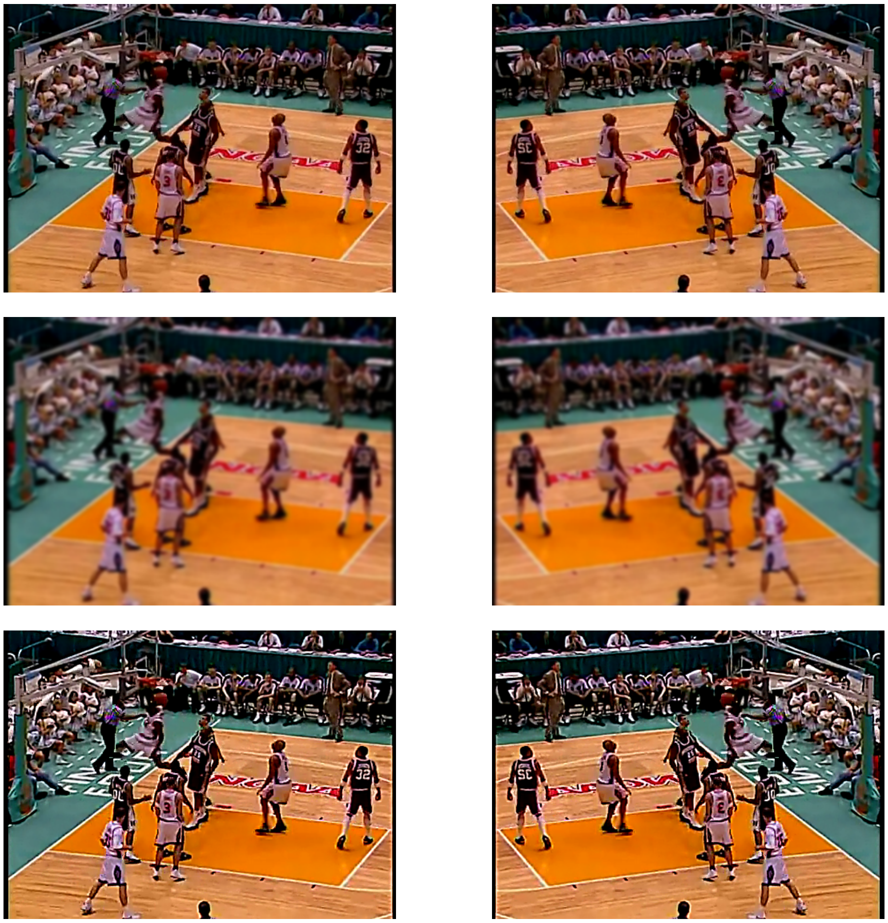

Figure 4.4: An example of each of the 5 image augmentations performed to the minority class data in this study: Original image (top left), flipped (top right), blurred (middle left), blurred & flipped (middle right), unsharpened (bottom left), blurred & unsharpened (bottom right).

To implement oversampling we began by using Python to split the training data into two separate data frames, one containing the majority class information (non-slam dunks) and one containing the minority class (slam dunks). From here we used the resample() function within the sklearn [7] library to randomly resample our augmented and non-augmented minority class data until it contained the same number of samples as the majority class (n=7,024). Once resampled, we combined the majority and resampled minority class data frames.

From here, the training dataset contained 14,048 samples with a 50:50 class split

[7] Library documentation: https://scikit-learn.org/stable/

and reduced the risk of class bias within our models. As with the image augmentation process, it is worth highlighting that it was important to hold off on oversampling our minority class data until after the data had been split into training/validation/testing sets. Doing this ensured that samples could not appear in both the training and testing datasets and distort our results.

4.3 Audio Spectrograms

Once the video segmentation process was complete, creating the audio spectrograms required for the analysis was a reasonably straightforward process. This process began by iterating through each of the 4 second long clips described in section 4.2 and extracting the audio from each clip into .wav format. To do this we once again utilised the Python FFMPEG wrapper (see listing 4.2) to iterate through each video clip and extract the single band clip audio to a .wav file with a sample rate of 44.1kHz and a bit rate of 160kbps as suggested by Urbano, Bogdanov, Herrera Boyer, Gomez Gutierrez, and Serra (2014). An example of the a .wav representation for both a dunk and non-dunk scene can be seen in figure 4.5.

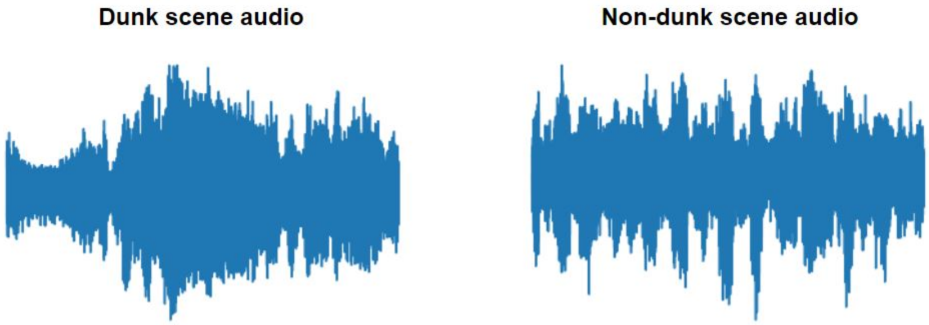

Figure 4.5: Illustration .wav files for slam dunk (left) and non-slam dunk (right) scenes

Additionally, since our audio information was coming from 10 different basketball games recorded across a number of years, we felt that it would be necessary to normalise the audio information in some way at a later stage to enable for meaningful analyses and comparisons across games. To facilitate this, we extracted the audio from each of the 90 minute full game videos to .wav format and stored these in a separate directory. Having access to the full game audio information allowed us to normalise the audio within each game and compare/analyse audio across games on the same scale (discussed later in this section).

Similar to the video segmentation process (see section 4.2), we created a pandas data frame to keep track of the source video locations and location for the corresponding audio files and spectrograms to follow. This table helped us avoid the issue of temporal mismatching between the visual (frames) and audio features when training the multi-sensory models, while also ensuring the training, validation, and testing partitions were consistent across the different sensory sources.

```python
import subprocess as sp
from moviepy.config import get_setting
from moviepy.tools import subprocess_call

def extract_audio(source_video_loc, audio_output_loc):
  cmd = ['ffmpeg', '-i', source_video_loc, '-ab', '160k', '-ac', '1',
 '-ar', '44100', '-vn', audio_output_loc]
  sp.call(cmd, shell=True)
```

Listing 4.2: Python function to extract single band audio from video clip at a sample rate of 44.1kH and bit rate of 160kbps.

Upon extracting the audio from each of the 4 second long video clips, we were ready to begin converting the raw audio files to a spectrogram representation. To give us some variety in the types of data we could use in our models, and to allow us to compare the performance of models built on different visual representations of the audio, we decided to create 4 different spectrograms for each of our 4 second long audio clips:

1. A **filter banks** representation of the audio plotted using a **hot** colour map.

2. A **filter banks** representation of the audio plotted using a **viridis** colour map.

3. An **MFCC** representation of the audio plotted using a **hot** colour map.

4. An **MFCC** representation of the audio plotted using a **viridis** colour map.

The inspiration behind these choices came from papers/studies researched as part of the literature review (see section 3.4) for this study. Studies such as Afonso et al. (1999) use of filter banks in electrocardiogram [8] (ECG) beat detection and Ballan et al. (2009) who utilised MFCC features in soccer scene classification inspired us to compare the effectiveness of filter bank vs. MFCC features when modelling audio information in a multi-sensory model.

As discussed in the literature review, the genesis for creating both hot and viridis versions of each image stems from the findings made by Amiriparian et al. (2017) when they identified that a viridis spectrogram yielded more effective results than a jet or grey representation. With these results in mind we thought that it would be a good idea to test and compare the performance of the viridis and hot colour maps when analysed within our multi sensory networks.

A naive approach towards extracting filter bank and MFCC features from the audio clips would be to simply take the raw audio extracts and pass them through a librosa [9] or python speech features [10] function. However, in order to increase the effectiveness of our multi-sensory models we decide to perform two pre-processing steps prior to extracting filter bank and MFCC features from the audio.

The first of these pre-processing steps was to simply downsample the audio frequency. As discussed in section 3.4, downsampling audio frequencies is a popular technique used in speech and audio processing which helps reduce the computational power required to analyse the raw frequencies in addition to reducing the amount of noise within the data. For this study, the raw audio frequencies were downsampled from 44.1kHz to 16kHz (similar to Ballan et al. (2009)). To perform the downsampling we utilised the load() function within the librosa library which allowed us to specify a sampling rate (via the sr parameter) when reading each of the .wav files.

[8] More information on ECG can be found at: https://irishheart.ie/your-health/heart-stroke-tests-procedures/ecg/

[9] Library documentation: https://librosa.github.io/librosa/

[10] Library documentation: https://python-speech-features.readthedocs.io/en/latest/

The second pre-processing step undertaken was to calculate the mean of both the filter bank and MFCC features from each full length audio file. As mentioned earlier in this section, we felt it was necessary to normalise our audio data in some way given the fact that our full game clips are coming from different audio sources recorded across a number of years. With this in mind, it was decided that we would normalise the filter bank and MFCC features extracted from each clip by subtracting the mean filter bank/MFCC feature value for the corresponding full game audio from each feature in the 4 second long audio clips (see Eq. 4.1 for formula used to normalise our MFCC features[11]).

$$\tilde{M}_{G,i} = M_{G,i} - \bar{M}_G \tag{4.1}$$

where:

$\tilde{M}_{G,i}$ = Normalised MFCC feature from game G at step i
$M_{G,i}$ = MFCC feature at from game G step i
$\bar{M}_G$ = Mean MFCC feature value from full game G

To calculate the mean filter bank and MFCC values across each game we iteratively read in the full game audio files (mentioned above) using librosa, downsampling each file to 16kHz. Once downsampled, we utilised the logfbank() and mfcc() functions available within the python speech features library with a fast Fourier transform (FFT) size of 1,103 to extract an array of filter bank and MFCC features from the full game audio clips. Once extracted we simply took the mean of these arrays and wrote them to a pandas data frame along with the game ID and source audio location. We would then join this data frame onto the audio data frame discussed earlier in this section to store the mean filter bank and MFCC information.

Once we had a mechanism for downsampling the audio clips in place, a dataframe containing information each of audio files, and corresponding mean values it allowed us to develop a series of relatively straightforward functions which could create and save a normalised spectrogram given some input audio data (see listing 4.3). With

[11] Note that the equation used to normalise filter bank features is identical, however we decided to illustrate just one equation to avoid unnecessary repetition.

these functions in place, it allowed to iterate through each audio file in our dataset and create the 4 spectrograms mentioned above. Figure 4.6 illustrates an example of each normalised spectogram type created for both a slam dunk and a non-slam dunk scene.

```python
import numpy as np
import librosa
from python_speech_features import mfcc, logfbank

def create_mfcc_viridis_mean_norm(input_loc, output_loc, game_mean):

  y, sr = librosa.load(input_loc, sr=16000)
  mfcc_feat = mfcc(y,sr,nfft=1103).T

  # mean normalise
  mfcc_feat_norm = mfcc_feat - game_mean
  fig = plt.figure()
  plt.imshow(mfcc_feat_norm, interpolation='nearest', cmap=cm.viridis, origin='lower', aspect='auto');
  plt.axis('off')

  # save the spectrogram
  fig.savefig(output_loc, bbox_inches='tight')

  # clear the figure (for memory management and efficiency)
  plt.clf()
  plt.close(fig)
```

Listing 4.3: Python function to create spectrogram of input audio using the viridis colour pallet and normaised MFCC features.

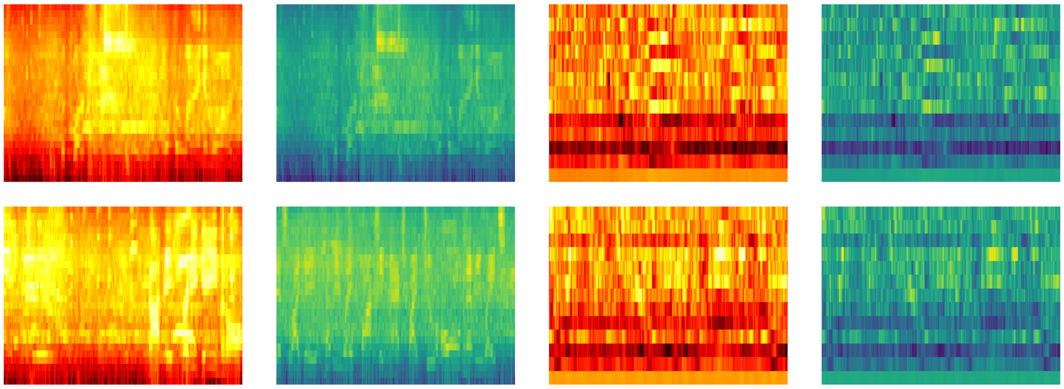

Figure 4.6: Examples of normalised filter banks (left & middle left) and MFCC (right and middle right) spectrograms for slam dunk (top) and non-slam dunk (bottom) scenes.

Finally, in order to ensure consistency between the audio and video datasets, we split the full audio data into train, validation and test partitions using the same proportions (60/10/30) and random seed as the visual data. We also oversampled the minority class in the audio training data using the same random seed that was used when oversampling the video data.

Once this was complete we compared the training (oversampled), validation, and testing datasets between the two sensory sources to ensure no temporal mismatching had taken place by comparing row counts, class balance, and the order of event labels between the two datasets. No data quality issues were identified.

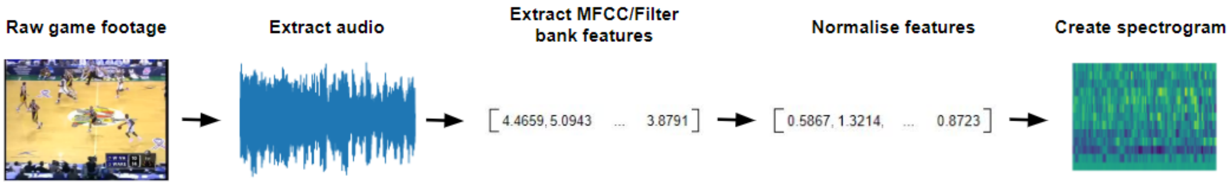

Figure 4.7: Flow chart of process required to create spectrograms.

Chapter 5

Experiment design and methodology

This chapter details the design of both the computer vision and multi-sensory models. For each model we describe the end to end architectures before detailing the performance metrics and statistical tools used to asses and compare their performance.

5.1 Computer Vision Model

The computer vision model provided the basis for our study. This model would not only form a component of the multi-sensory model, but would also be compared to the multi-sensory model in order to test our hypothesis (see section 6.1). The computer vision model was a standard a hybrid network architecture comprised of both CNN and LSTM layers. This model can be thought of as the eyes of our overall architecture and how it operates can be broken up into 3 steps:

1. Extract bottleneck features from frames using InceptionV3 architecture
2. Understand the temporal evolution of extracted features using an LSTM layer
3. Classify video

Before passing any of the frames in our dataset through the Inceptionv3 portion of the model, it was important to configure the architecture parameters in order to apply transfer learning and incorporate the architecture into our model. The first configuration performed on the architecture was removing the final dense layer of the original model. Since we were developing a hybrid (CNN-LSTM) network model, we were not interested in making any predictions using the InceptionV3 model alone. As a result of this we could remove the prediction layer and replace it with a pooling layer.

In our experiments (described in section 6.1.1) this pooling layer took one of two forms (maximum and mean) and was used to condense the features extracted in the penultimate layer of the network into a 2,048 dimensional vector. This vector of bottleneck features would subsequently be passed into the LSTM component of our model before a final classification is produced.

The second configuration required was specifying the weights (illustrated in figure 5.1) to use within the InceptionV3 model. Here we had one of two options:

1. Implement the default ”ImageNet” weights within the network. These weights were derived through backpropogation when training the InceptionV3 architecture ImageNet dataset.

2. Retrain the network (or a portion of the network) on our basketball image dataset and derive a new set of weights for the model.

Since the ImageNet dataset [1] contained a number of images related to basketball and slam dunks in addition to pictures of similar team sports such as netball, we were confident that the default ImageNet weights would be suitable for extracting features from our input images and configured the architecture accordingly.

[1] More information on the dataset can be found at: http://image-net.org/index

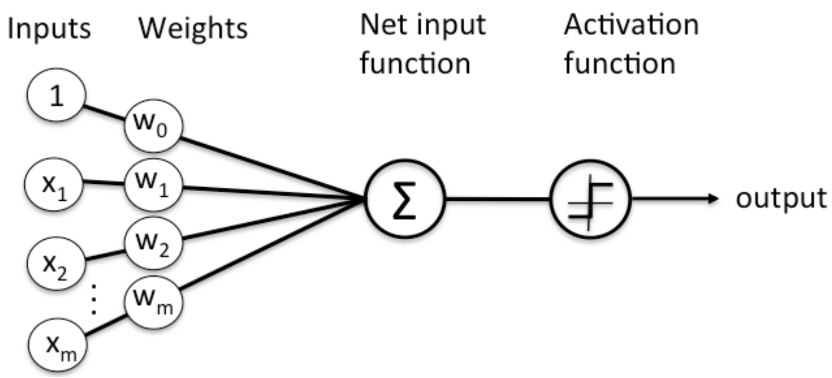

Figure 5.1: Illustration of a simple neural network node describing the role of weights within the network (source: https://skymind.ai/wiki/neural-network).

Once the network was configured, it was necessary to reshape our images to the dimensions required by the input layer of the InceptionV3 architecture. To do this, we utilised the Keras image pre-processing functions to reshape each image to 229x229 pixels, extract the RGB values (from 0 to 255) from each pixel into 3 separate arrays (one containing the red values, one containing the green, and one containing the blue), and normalise these values in the range [-1,1].

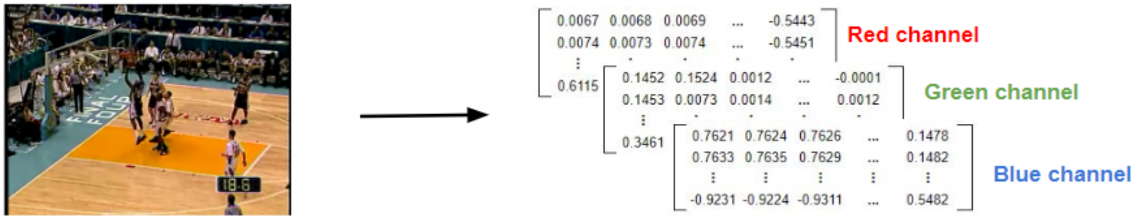

Figure 5.2: Illustration of how a frame extracted from a video segment gets converted into normalised 229x229 RGB arrays.

From here, we were ready to begin training the computer vision model. To do this, we iteratively went through each of the 4 second long clips in our training set passing each of the 26 sub-sampled frames through the reconfigured InceptionV3 architecture in order. The model produced a 2,048 dimensional vector for each frame in the clip.

Next, we concatenated vector representation for each frame in a clip together. This resulted in a 26x2,048 dimensional array which represented the extracted visual features from a given training clip as they evolve over time. Finally, this 26x2,048 dimensional array was then appended to a larger array (referred to as the feature array henceforth) which would house each of the 26x2,048 dimensional arrays for each clip in the training dataset.

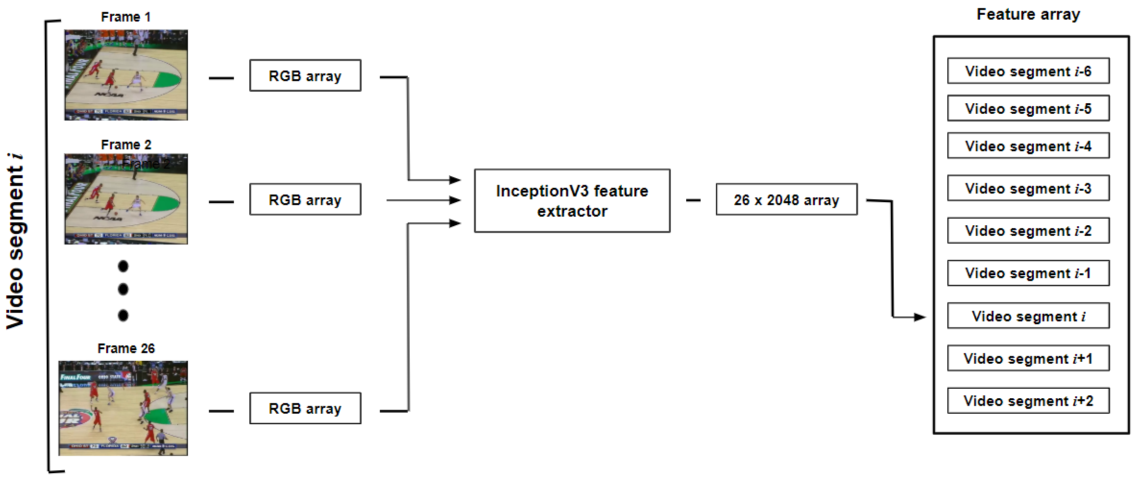

Figure 5.3: Flow diagram of the video clip feature extraction process.

It is worth noting here that as we processed more and more clips through the InceptionV3 network and appended them to our feature array, we observed that the feature extraction process began to slow down significantly. This was due to the increasingly large size of the feature array being stored in memory on our machine.

To overcome this, we began processing clips through the InceptionV3 network in batches of 500 clips (13,000 frames) at a time. Each 500 clip feature array (referred to as sub-feature arrays henceforth) was then saved to our hard drive before being wiped from memory to begin processing the next batch of frames. Once every frame in the training set had been processed through the InceptionV3 network, we were able to concatenate each of the sub-feature arrays together to create the feature array. This approach resulted in significant time savings when developing the model and allowed us to reconfigure/tune the LSTM portion of the network without having to retrain the entire network.

Once each clip had been processed by the InceptionV3 portion of the network, we were able to begin passing each of our 26x2,048 dimensional arrays through the LSTM portion of the model. This LSTM architecture analysed the temporal relationship and evolution of our extracted features.

The activation function used within this component of the network was the leaky rectified linear unit (RELU) activation function. We chose this function not only for it's low computational cost but also since unlike the standard RELU function, it maintains all of the feature information (Zhang, Zou, & Shi, 2017). We felt that this retention of additional information would help the classifier while also facilitating the use of pre-class score fusion within our subsequent multi-sensory models by outputting more signal from the LSTM component. A mathematical comparison of the RELU and leaky RELU activation functions can be found in equations 5.1 (RELU) and 5.2 (leaky RELU), while a visual comparison can be seen in figure 5.4.

$$f(x) = max(0, x) \tag{5.1}$$

$$f(x) = max(0.1x, x) \tag{5.2}$$

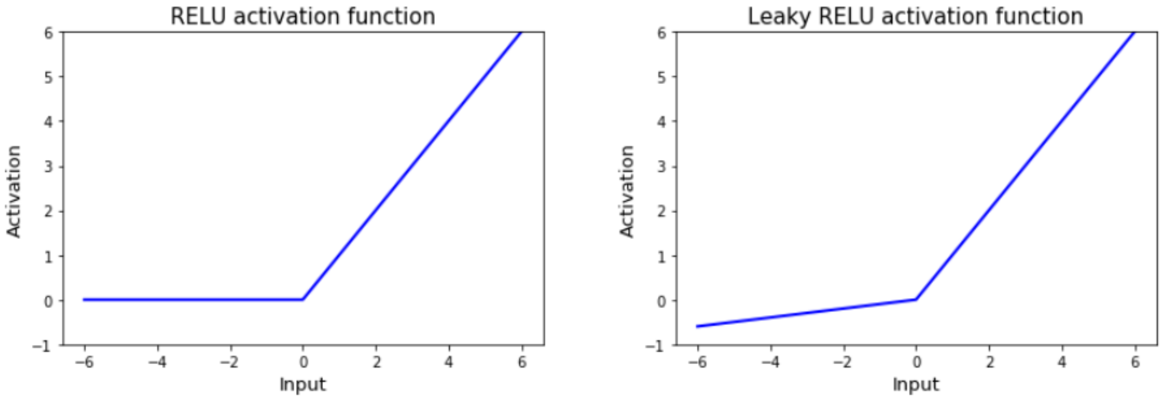

Figure 5.4: Comparison of RELU and leaky RELU activation functions.

The output from the LSTM portion of the network was then passed into a single two node dense layer with a softmax activation function. This layer determined the final classification output for the model by essentially assigning a probability that the video analysed by the model belongs to the non-dunk or dunk class. Once these probabilities (which sum to 1) were calculated, the final classification from the model was deemed to be the class with the highest probability.

Note that while we could have used other activation functions for the final layer of the model (such as the sigmoid or hyperbolic tangent activation functions), we chose to use the softmax activation function as we felt having a higher dimensional output (two versus one), and a spread of probabilities across both classes in the output layer would help us when performing any class score fusion within the multi-sensory model (see section 5.2). Figure 5.5 provides an illustration of the computer vision model at a high level.

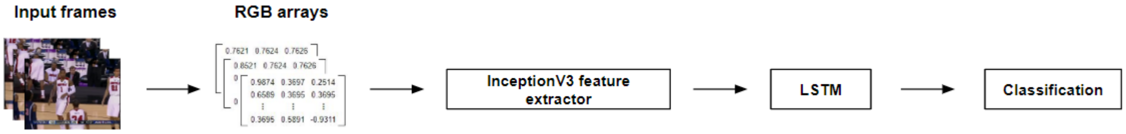

Figure 5.5: High level visual summary of the computer vision model architecture.

As mentioned in section 1.5, the performance of each computer vision model is assessed based on 4 key performance metrics:

1. Mean class precision (MCP)
2. Mean class recall (MCR)
3. True positive rate (TPR)
4. True negative rate (TNR)

These 4 metrics were chosen due to the large amount of imbalance within the testing set. As a result of this, relying on a single measure such as accuracy would not provide a fair assessment of how the model was performing.

For example in our case the model could run at 99% accuracy but not classify a single slam dunk scene correctly. With this in mind, we felt that the weighted MCP and MCR metrics, similar to that suggested by Murray, Renals, Carletta, and Moore (2006) and available within the scikit-learn module[2], along with TPR/TNR as suggested by Batista et al. (2004) would help us gain a better understanding of how our models performed on this particular dataset at a high level while also helping us identify each models strengths and weaknesses.

[2] Library documentation: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.classification_report.html

The formulas used to calculate these metrics are detailed in equations 5.3, 5.4, 5.5, and 5.6:

$$MCP = \frac{((Precision_{non-dunk})(Support_{non-dunk})) + ((Precision_{dunk})(Support_{dunk}))}{N} \tag{5.3}$$

where:

$$\begin{aligned} &Precision_i &&= \text{Model precision for class } i \\ &Support_i &&= \text{Number of samples for class } i \text{ in the testing set} \\ &N &&= \text{Total number of samples in the testing set} \end{aligned}$$

$$MCR = \frac{((Recall_{non-dunk})(Support_{non-dunk})) + ((Recall_{dunk})(Support_{dunk}))}{N} \tag{5.4}$$

where:

$$\begin{aligned} &Recall_i &&= \text{Model recall for class } i \\ &Support_i &&= \text{Number of samples for class } i \text{ in the testing set} \\ &N &&= \text{Total number of samples in the testing set} \end{aligned}$$

$$TPR = \frac{TP}{TP + FN} \tag{5.5}$$

where:

$$\begin{aligned} &TP &&= \text{Number of true positives predicted by the model} \\ &FN &&= \text{Number of false negatives predicted by the model} \end{aligned}$$

$$TNR = \frac{TN}{TN + FP} \tag{5.6}$$

where:

$$\begin{aligned} &TN &&= \text{Number of true negatives predicted by the model} \\ &FP &&= \text{Number of false positives predicted by the model} \end{aligned}$$

5.2 Multi-Sensory Model

The multi-sensory model(s) developed as part of this study can the thought of as a two stream network in which stream 1 is simply the computer vision model described above (in section 5.1). This stream, as before, models the visual component of the input video through a hybrid network model.

Stream 2 of the network (illustrated in figure 5.6) analyses the audio information from each scene in the form of spectrograms (described in section 4.3) and can be thought of as the ears of our architecture.

Since the each audio spectrograms model the audio frequency over each 4 seecond long video, the models developed as part of the audio stream did not require a mechanism for analysing time series data (such as an LSTM). As a result of this the architecture used to model the audio information within this study was a pre-trained CNN architecture [3], similar to that used in our computer vision model for bottleneck feature extraction.

As with the computer vision model, it was necessary for us to configure the pre-trained architectures parameters in order to effectively utilise the power of transfer learning. Similarly to the computer vision model, we removed the classification layer from the pre-trained architecture to facilitate our binary classes (dunk and non-dunk). However, we also introduced a 128 node fully connected layer into the pre-trained architecture prior to the classification layer. This layer would not only help improve the performance of the model, but would also facilitate pre-class score fusion by ensuring that the output from the penultimate layer of both the vision and audio streams were both the same length ($l = 128$). Once again, it is worth highlighting that the activation function used for this layer was the leaky RELU activation function.

Similar to before, rather than completely retraining the InceptionV3 or VGG19 architectures, we utilised the default ImageNet weights for the pre-trained portion of the audio model. Instead, we allowed the model to train the additional 128 node fully connected layer as well as the classification layer. While this could be considered a

[3] We experimented with both InceptionV3 and VGG19 architectures, see chapter 6 for more discussion on this

limitation, or weakness in the model (and an area for future improvement), hardware limitations restricted the amount of retraining we could perform within the audio stream as part of the multi-sensory network.

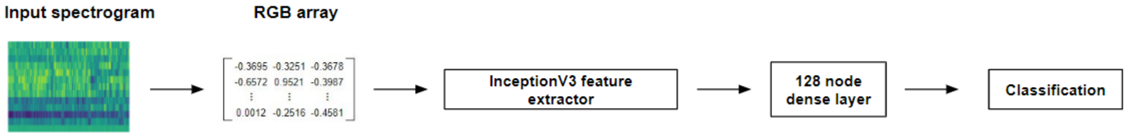

Figure 5.6: Illustration of the audio steam which makes up our multi-sensory architecture.

Once each of the sensory streams were in place, we needed a mechanism for combining the information captured by the eyes and ears of the model before making a final prediction. To accomplish this we added a fusion layer to the model. While we experimented with a number of different forms of fusion (discussed in more detail in chapter 6), this layer can be simply thought of as a layer which takes the output vectors from each of our sensory streams, and combines them into a single vector via some simple mathematical operation (addition, mean, subtraction, etc.).

Once the multi-sensory data had been processed through each stream of the network and merged into a single vector, a simple (no more than 2 hidden layers) dense architecture analysed the information within the vector before making the final classification for the model [4]. An overview of the end-to-end multi sensory architecture is illustrated in figure 5.7.

[4] Note: We also explored using support vector machines for this component of the model instead of a dense neural network and will be discussed in 6

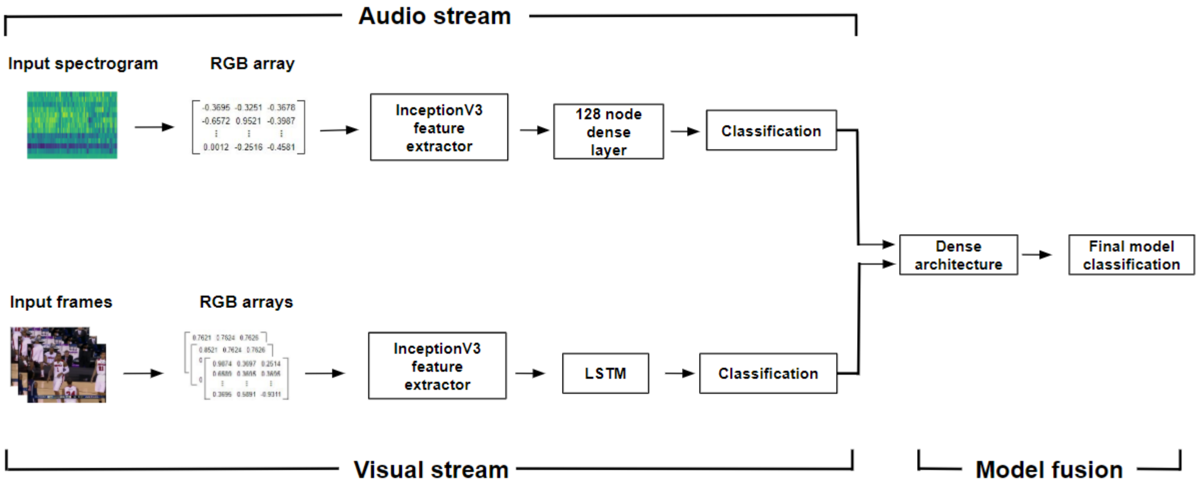

Figure 5.7: High level summary of the end to end multi-sensory model architecture.

As with the computer vision model discussed above, the multi-sensory model was assessed based on MCP, MCR, TPR, and TNR. Having the same set of metrics, and a confusion matrix calculated for both models (trained and tested on the same data) allowed us to informally compare and assess the performance of both the computer vision model and multi-sensory models (identify which model is more precise, which has a lower false positive rate, etc.).

When it came to testing our hypothesis, we used McNemear's test to formally compare the errors between the models for statistical significance. McNemar's test is a non-parametric statistical test which compares the disagreements between two sets of model predictions using a contingency table (5.1).

McNemear's test has been proposed as a suitable statistical tool for comparing the performance of two models which are computationally expensive to train, such as deep neural networks (Dietterich, 1998). In cases where the models are computationally expensive and time consuming to train, traditional methods such as cross validation are not feasible.

		Model 2	
		Correct	Incorrect
Model 1	Correct	a	b
	Incorrect	c	d

Table 5.1: Example of a contingency table in which a represents the number of instances Model 1 and Model 2 prediction correctly, b indicates the number of instances that model 1 predicted correctly which model 2 predicted incorrectly, etc.

In order to accept or fail to reject the null hypothesis, we compared the McNemear's test statistic to a χ^2 distribution with 1 degree of freedom and a significance level of $\alpha = 0.05$ (see figure 5.8). If the improvements in the multi-sensory model across all 4 metrics were deemed to be statistically significant, we could reject H_0 and accept that the multi sensory model is a more effective classifier of slam dunk sequences when compared to a computer vision model.

Eq. 5.7 describes the traditional calculation of the McNemar's test statistic. However, since the differences in errors between our models were small ($b + c < 25$), we calculated the test statistic using a modified formula (Eq. 5.8) proposed by Edwards (1948). This formula corrects for continuity and more accurately approximates the χ^2 distribution for smaller sample sizes.

$$\chi^2 = \frac{(b-c)^2}{b+c} \tag{5.7}$$

$$\chi^2 = \frac{(|b-c|-1)^2}{b+c} \tag{5.8}$$

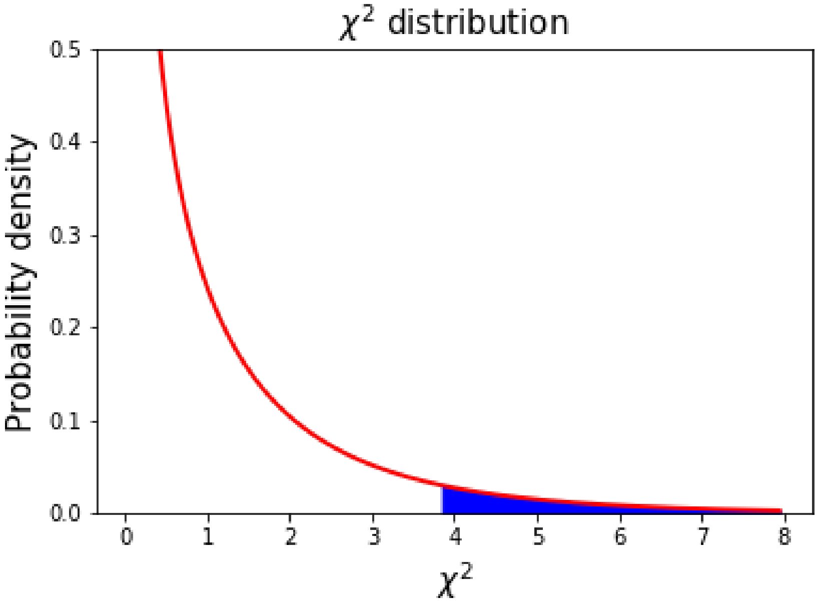

Figure 5.8: Illustration of the χ^2 with 1 degree of freedom and significance level of $\alpha = 0.05$ (critical region shaded in blue).

Chapter 6

Results, evaluation and discussion

This chapter details and discusses the results from each experiment conducted as part of this dissertation. We begin by detailing the results obtained when each model developed was applied to the testing set.

From here, we compare the performance of the computer vision model to the multi-sensory model before comparing the performance of multi-sensory models developed using different audio features.

Finally, we discuss these results and what they mean in the context of the research objectives outlined in section 1.4.

6.1 Results

6.1.1 Computer Vision Model

6.1.1.1 Experiment CV-1: InceptionV3 with mean pooling bottleneck features and 128 node LSTM

The first computer vision experiment conducted employed the InceptionV3 architecture with a mean-pooling bottleneck layer to create a feature map for each frame in our training datasets, these feature maps where then analysed by an LSTM layer containing 128 hidden units, similar to the architecture used by Chow and Dibua (2018) in their study into tennis action recognition. This LSTM layer also included

L2 regularisation ($\lambda = 0.03$) and a dropout component ($p = 0.5$) to reduce the risk of over-fitting.

This model was trained using the Adam optimiser with a dynamic learning rate [1]. This feature allowed the model to reduce the learning rate while training if the validation loss has had not improved over 5 iterations.

The model was trained using the standard categorical cross entropy loss function for 100 epochs. Once again, to avoid over-fitting a callback was implemented while training to save the best version of the model. This callback identified which epoch produced the model with the lowest validation loss and saved the corresponding weights to the hard drive on our machine. From figure 6.1 we can see that the best model produced during this experiment occurred after roughly 40 epochs.

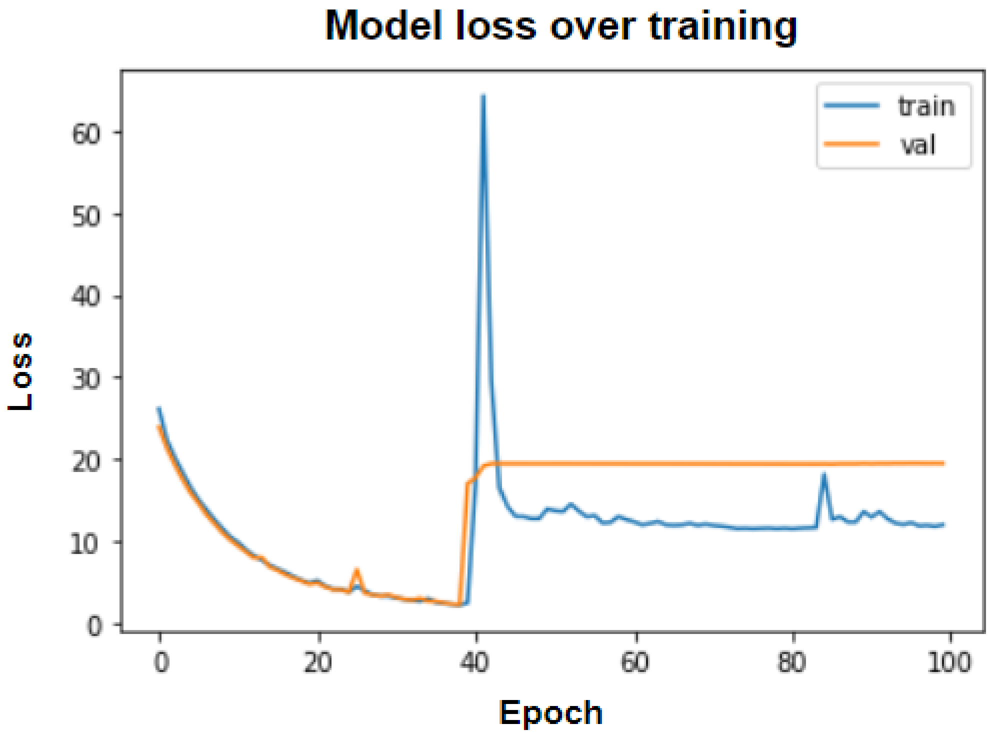

Figure 6.1: Illustration of training and validation loss over training for the model described in experiment CV-1.

Table 6.1 describes the confusion matrix generated when this model was applied to

[1] Reduced learning rate documentation: https://keras.io/callbacks/#reducelronplateau

the testing dataset. Examining this table we can see that the model correctly classifies 7 out of 16 slam dunk scenes (TPR = 0.4375), and 3,362 out of 3,512 non-slam dunk scenes (TNR = 0.9573). On the other hand we can see that this model incorrectly classified 9 slam dunk scenes as a non-slam dunk scene and 150 non-slam dunks as slam dunks.

		Predicted	
		Non-dunk	Dunk
Actual	Non-dunk	3,362	150
	Dunk	9	7

Table 6.1: Confusion matrix produced when the model developed as part of experiment CV-1 was applied to the training set.

Table 6.2 allows us to more formally assess the models performance in terms of the metrics described in section 5.1. From this table we can see that while the model operated with an MCP of 99.29%, and an MCR of 95.49%, the model performs particularly poorly at returning relevant slam dunk scenes. This is highlighted by the extremely low precision score for the slam dunk class of 4.46%.

	Precision	**Recall**	**TPR**	**TNR**	**Support**
Non-dunk	0.9973	0.9573			3,512
Dunk	0.0446	0.4375			16
Total/Avg	0.9929	0.9549	0.4375	0.9573	3,528

Table 6.2: Summary of model metrics from experiment CV-1 which includes precision and recall metrics for each class, MCP, MCR, TPR, TNR, and support.

6.1.1.2 Experiment CV-2: InceptionV3 with maximum pooling bottleneck features and 128 node LSTM

With the results from section 6.1.1, we decided to explore the effect of using a maximum pooling layer as the final layer of the InceptionV3 feature extractor before passing these features into our LSTM architecture.

Surprisingly, when we examine the confusion matrix in table 6.3 and the evaluation metrics described in table 6.4 we can see a decrease in performance based on all of our model evaluation metrics. This decrease in performance can mainly be attributed to the high type II error rate within the non-dunk class (825). This high type II error rate resulted in a 76.51% TNR. We can also see the effects this has on the dunk class metrics, dropping the TPR to 6.25%.

		Predicted	
		Non-dunk	Dunk
Actual	Non-dunk	2,687	825
	Dunk	15	1

Table 6.3: Confusion matrix produced when the model developed as part of experiment CV-2 was applied to the training set.

	Precision	**Recall**	**TPR**	**TNR**	**Support**
Non-dunk	0.9944	0.7651			3,512
Dunk	0.0012	0.0625			16
Total/Avg	0.9899	0.7619	0.0625	0.7651	3,528

Table 6.4: Summary of model metrics from experiment CV-2.

Finally, we can see from figure 6.2 that this model struggled to converge on an acceptable loss for both the training and validation sets (it never drops below 20). This further suggests that the model struggled with learning the required features when using the maximum pooling feature maps.

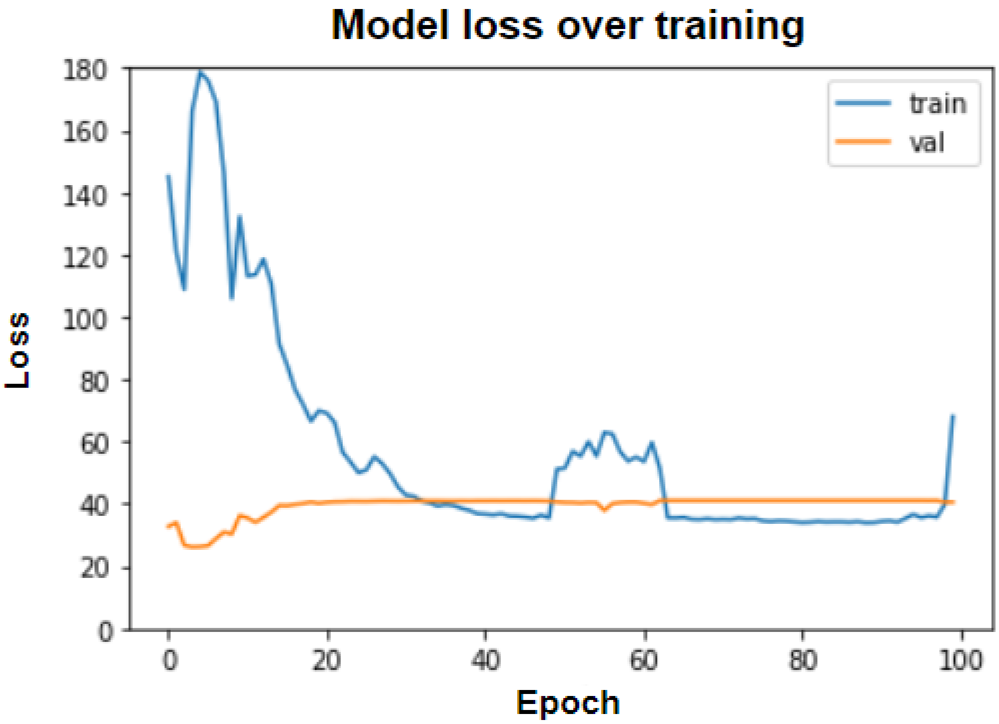

Figure 6.2: Illustration of training and validation loss over training for the model described in experiment CV-2.

6.1.1.3 Experiment CV-3: InceptionV3 mean pooling bottleneck features and 256 node LSTM

In this experiment, we increased the number of hidden units within the LSTM portion of the model described in experiment CV-1 (section 6.1.1.1) to 256 and explored what effect this had on the classification power of the model.

Tables 6.5 and 6.6 describe the confusion matrix and summary metrics for this model. Examining these tables we can see that while this model had a lower TPR, correctly classifying only 3 out of 16 slam dunk scenes (18.75%) when compared to the 128 hidden unit model, this model performed well at correctly classifying non-slam dunk scenes (3,456 out of 3,512) with a TNR of 98.41%.

The models proficiency for correctly classifying non-slam dunks scenes resulted in an increase in slam dunk class precision (5.08%) when compared to the 128 hidden unit model described in experiment CV-1 (4%). This increase in precision resulted in a higher proportion of relevant positive samples being retrieved by the model.

		Predicted	
		Non-dunk	Dunk
Actual	Non-dunk	3,456	56
	Dunk	13	3

Table 6.5: Confusion matrix produced when the model developed as part of experiment CV-3 was applied to the training set.

	Precision	**Recall**	**TPR**	**TNR**	**Support**
Non-dunk	0.9962	0.9841			3,512
Dunk	0.0508	0.1875			16
Total/Avg	0.9919	0.9805	0.1875	0.9841	3,528

Table 6.6: Summary of model metrics from experiment CV-3.

We can see from the training/validation loss plotted in figure 6.3 that there are no signs to suggest underfitting with this model and that the model converged to its minimum loss after ∼70 epochs.

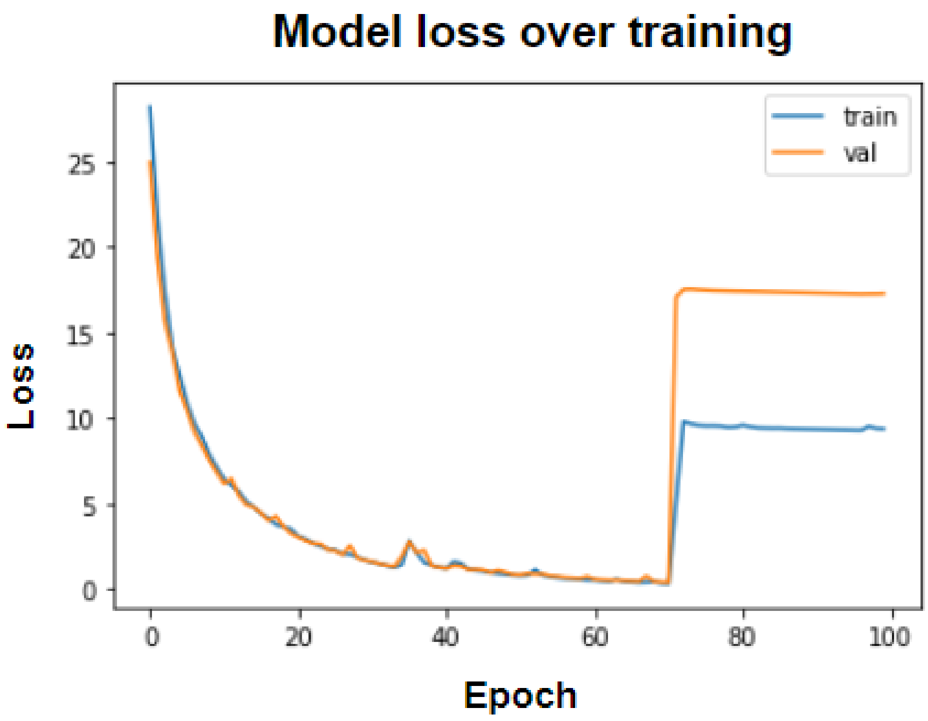

Figure 6.3: Illustration of training and validation loss over training for the model described in experiment CV-3.

6.1.1.4 Other computer vision experiments

In this subsection we briefly summarise a number of other computer vision experiments conducted as part of this study. For the most part, these experiments yielded no results worthy of us exploring these models any further as part of a multi-sensory network. Nonetheless, we are including them in this dissertation to inform readers of potential avenues to avoid or areas to improve when performing similar research. A summary of how each model from these experiments performed on the testing set found in table 6.7, in which:

- TN $\implies$ True negative count
- FN $\implies$ False negative count
- FP $\implies$ False positive count
- TP $\implies$ True positive count

Experiment CV-4: Reversing the idea of experiment CV-3, in this experiment we explored the effect of reducing the number of hidden units within the LSTM layer to 64. The model created as part of this experiment was unable to correctly classify any of the slam dunk scenes within the testing dataset while incorrectly classifying 16 non-slam dunk scenes.

Experiment CV-5: This experiment involved exploring how a 2,048 hidden unit LSTM with the tanh activation function performed when processing InceptionV3 mean pooling bottleneck features. This model performed exceptionally poorly when compared to other computer vision models at classifying non-slam dunk scenes (correctly classifying only 2,019 out of 3,512). As a result of this, despite classifying 6 out of 16 slam dunk scenes correctly the slam dunk class precision of this model was a disappointing 0.4%.

Experiment CV-6: In this experiment we examined what effect removing any regularisation from the LSTM component had on the overall classification power of the model. To test this, we modified the architecture described in experiment CV-1 (section 6.1.1.1) to remove L2 regularisation from the LSTM layer. Perhaps unsurprisingly, this meant that the model did not generalise well to unseen data and resulted in the model not classifying any slam dunk scenes (0% TPR) from the test set correctly while also misclassifying 8 non-slam dunk scenes as slam dunks.

Model ID	**TN**	**FP**	**FN**	**TP**
CV-4	3,488	24	16	0
CV-5	2,019	1,493	10	6
CV-6	3,504	8	16	0

Table 6.7: Summary of confusion matrices produced when models described in the other computer vision experiments section were applied to the testing set.

6.1.2 Multi-Sensory Model

6.1.2.1 Experiment MS-1: Mean pooling InceptionV3 features, 128 node LSTM computer vision model, MFCC viridis audio features and mean class prediction fusion layer

Given the performance of the model described in experiment CV-1 (relatively high true positive rate compared to other models and high number of false negatives), we decided that this would be an ideal model to develop a multi-sensory network on and compare results.

For the audio component of the model, we trained the audio architecture described in section 5.2 on the normalised MFCC spectrograms plotted using the viridis colour map. Table 6.8 details the confusion matrix for the audio stream. This portion of the model correctly classified a meagre 1 out of 15 slam dunk cases (TRP = 6.25%), while classifying 3,504 out of 3,512 non-slam dunk scenes correctly (TNR = 99.77%).

		Predicted	
		Non-dunk	Dunk
Actual	Non-dunk	3,504	8
	Dunk	15	1

Table 6.8: Confusion matrix obtained when testing set is applied to audio stream of the multi-sensory network MS-1.

The class predictions from each stream were combined in the fusion layer by taking the mean prediction for each class between the models. This portion of the model was a dense architecture containing 2 hidden layers both utilising L2 regularisation ($\lambda = 0.01$), and dropout ($p = 0.5$).

From here, the model was trained over 2,000 epochs with a batch size of 128. As with the computer vision models described in section 6.1.1, we utilised the cross entropy loss function and Adam optimiser (with a dynamic learning rate). Once again we implemented a model callback to ensure the model with the lowest validation loss was saved to the hard drive.

Table 6.9 and figure 6.4 illustrate the multi-sensory model's confusion matrix and training/validation loss over time respectively. When examining these model outputs, two items stand out:

1. This version of the multi-sensory model appears have performed extremely well at classifying non-slam dunk scenes, correctly classifying an impressive 3,511 out of 3,512 scenes.
2. The fact that the validation loss was lower than the training loss at all points during training suggests that this particular model was underfit and would need to be refined.

		Predicted	
		Non-dunk	Dunk
Actual	Non-dunk	3,511	1
	Dunk	14	2

Table 6.9: Confusion matrix produced when the model developed as part of experiment MS-1 was applied to the training set.

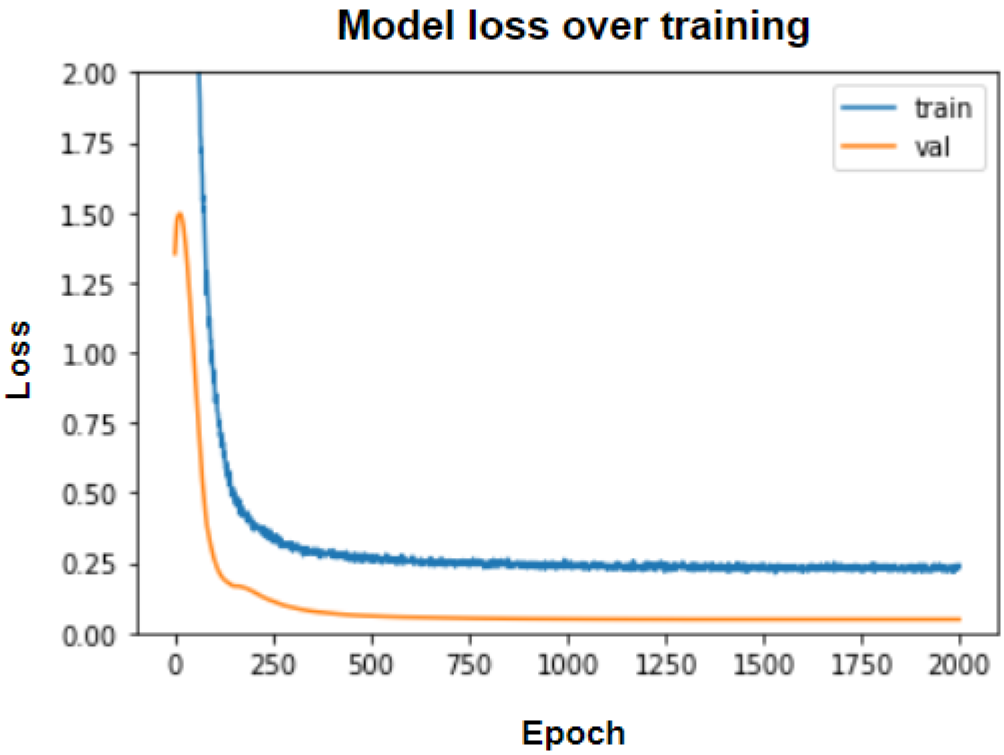

Figure 6.4: Illustration of training and validation loss over model training for the model described in experiment MS-1.

When we take a deeper dive into the performance metrics we can see confirmation that this model performs extremely well when classifying non-dunk scenes, boasting a 99.94% TNR. We can also see that although the model performs quite poorly at identifying the slam dunk scenes as is evidenced by the models 12.5% TPR, the model does a decent job at returning relevant samples with a precision value 66.67% for the slam dunk class. It's worth highlighting however that this metric may be slightly misleading due to the small number of positive predictions ($n = 3$) made by the model.

	Precision	**Recall**	**TPR**	**TNR**	**Support**
Non-dunk	0.9960	0.9997			3,512
Dunk	0.6667	0.1250			16
Total/Avg	0.9945	0.9957	0.1250	0.9997	3,528

Table 6.10: Summary of model metrics from experiment MS-1.

6.1.2.2 Experiment MS-2: Mean pooling InceptionV3 features, 128 node LSTM computer vision model, MFCC viridis audio features, mean class prediction fusion layer and reduced regularisation

While the precision of the model described above in section 6.1.2.1 was encouraging, the fact that the model appeared to be underfit was a cause for concern and was something we wanted to overcome in this experiment.

We understood that over-regularisation is one of the main contributors to underfitting in deep neural networks (Alsheikh, Niyato, Lin, Tan, & Han, 2016). With this in mind, we decided to modify the fusion component of the model described in section 6.1.2.1 by removing any regularisation from both hidden layers while also reducing the probability of dropout within the first hidden layer to $p = 0.3$ and removing the dropout component of the second hidden layer.

When we examine the confusion matrix for this model (table 6.11) we can see that the model performs similarly to the underfit model described in experiment MS-1 (section 6.1.2.1). Once again we can see that the model correctly classifies 2 out of 14 slam dunks (TPR = 12.5%). However, in this case the model incorrectly classifies 2 out of the 3,512 non-slam dunk scenes (TNR = 99.94%). Encouragingly, when we turn our attention to the training/validation loss over epochs (illustrated in figure 6.5) we can see that the reduced regularisation in our fusion layers remedied the underfitting which was present in experiment MS-1.

		Predicted	
		Non-dunk	Dunk
Actual	Non-dunk	3,510	2
	Dunk	14	2

Table 6.11: Confusion matrix obtained when testing set is applied to audio stream of the multi-sensory network MS-2.

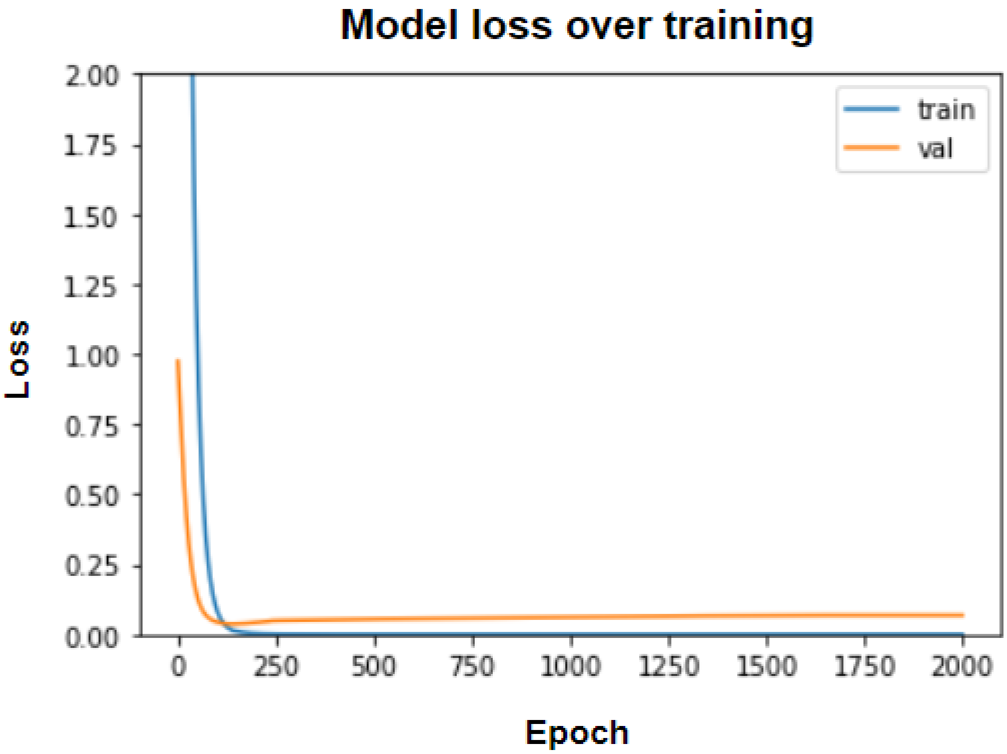

Figure 6.5: Illustration of training and validation loss over model training for the model described in experiment MS-2.

As expected when examining the confusion matrix, the model assessment metrics are largely similar to that of the underfit model. We observed an MCP of 99.37% and MCR value of 99.54%. Comparing the results to the previous model (MS-1) we noticed that the precision for the dunk class has dropped from 66.67% to 50%. This of, course is a result of the model incorrectly classifying an additional non-slam dunk scene as a slam dunk. While this decrease in precision is disappointing, it is still an improvement over that of the corresponding computer vision model (4.46%).

	Precision	**Recall**	**TPR**	**TNR**	**Support**
Non-dunk	0.9960	0.9994			3,512
Dunk	0.5000	0.1250			16
Total/Avg	0.9937	0.9954	0.1250	0.9994	3,528

Table 6.12: Summary of model metrics from experiment MS-2.

6.1.2.3 Experiment MS-3: Mean pooling InceptionV3 features, 128 node LSTM computer vision model, MFCC hot audio features, mean class prediction fusion layer and reduced regularisation

In this experiment we decided to investigate how the multi-sensory network described in section 6.1.2.2 performed when the audio portion of the network was trained using normalised MFCC features plotted with the hot colour map (as opposed to the viridis colour map used in sections 6.1.2.1 and 6.1.2.2).

Taking a look at the confusion matrix produced by the audio portion of this model, we can see that the model performed poorly at identifying slam dunk scenes with a TPR of 0%. Intriguingly, the confusion matrix suggested that this audio model performed better than the model trained using the MFCC viridis spectrograms at classifying non-slam dunk scenes by correctly identifying 3,507 out of 3,512 non-slam dunk scenes (TNR = 99.86%).

		Predicted	
		Non-dunk	Dunk
Actual	Non-dunk	3,507	5
	Dunk	16	0

Table 6.13: Confusion matrix obtained when testing set is applied to audio stream of the multi-sensory network MS-3.

Moving over to the multi-sensory model results, we can quickly see from the confusion matrix (described in table 6.14) that this model did not perform well when compared to the models described in 6.1.2.1 and 6.1.2.2. The multi-sensory MFCC hot model was unable to correctly classify any of the slam dunk scenes (TPR = 0%) while correctly classifying all but 3 of the non-slam dunk scenes (99.15%). We can also see from the model loss over time, that this model does not appear to be underfit.

		Predicted	
		Non-dunk	Dunk
Actual	Non-dunk	3,509	3
	Dunk	16	0

Table 6.14: Confusion matrix produced when the model developed as part of experiment MS-3 was applied to the training set.

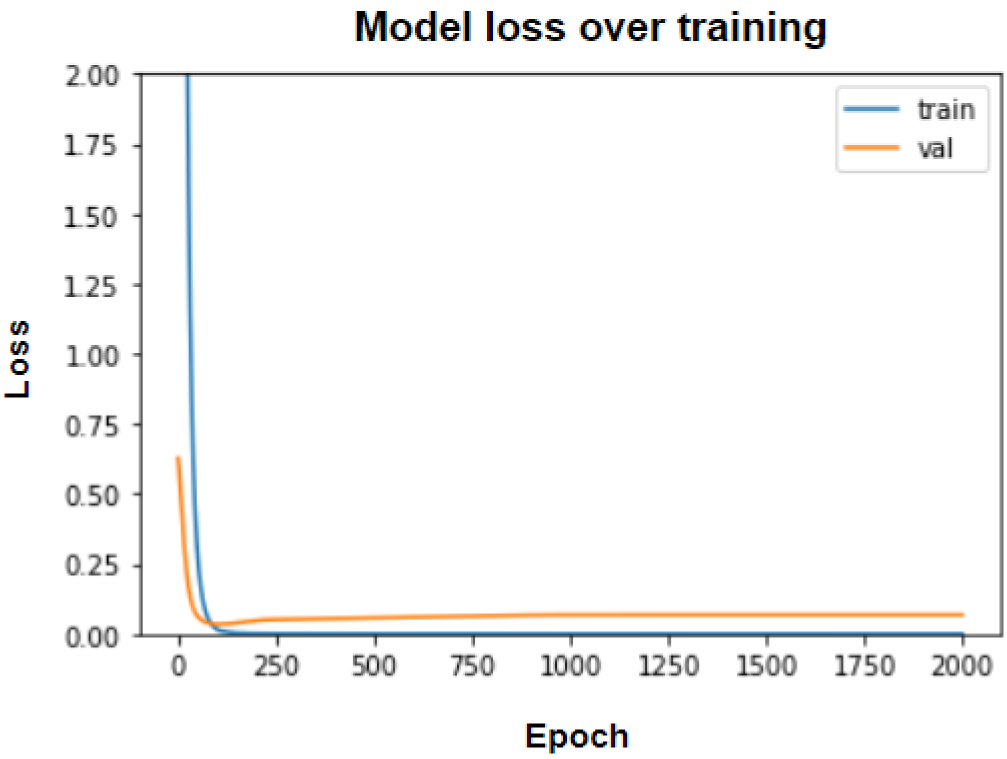

Figure 6.6: Illustration of training and validation loss over model training for the model described in experiment MS-3.

Taking a closer look at the model performance we can see that this model operates with an MCP of 99.89% and an MCR of 99.46%.

Considering that everything in this model apart from the audio colour map is identical to that from the model described in section 6.1.2.2, these result suggests that perhaps models built on the viridis colourmap perform better than that of the hot map and could support the findings made by Amiriparian et al. (2017). This idea will be formally tested in section 6.2.

	Precision	Recall	TPR	TNR	Support
Non-dunk	0.9954	0.9991			3,512
Dunk	0.0000	0.0000			16
Total/Avg	0.9909	0.9946	0.0000	0.9991	3,528

Table 6.15: Summary of model metrics from experiment MS-3.

6.1.2.4 Experiment MS-4: Mean pooling InceptionV3 features, 256 node LSTM computer vision model, MFCC viridis audio features, mean class prediction fusion layer and reduced regularisation

Given the performance of the 256 node LSTM computer vision model described in section 6.1.1.3 compared to the 128 node model described in section 6.1.1.1 (lower true positive rate and false positive rate) in addition to the performance of the multi-sensory model described in experiment MS-2 (section 6.1.2.2), we decided to combine these models and explore how the addition of the MFCC viridis audio stream impacts the results of a computer vision model with a lower false positive rate.

When we take a look at the confusion matrix described in table 6.16 we can see that the model successfully classifies 2 out of 16 slam dunk scenes (TPR = 12.5%) while incorrectly classifying 22 non-slam dunk scenes (TNR = 99.37%). Although these results are not as precise as those achieved by the 128 node multi-sensory model (section 6.1.1.1), the proportion of relevant slam dunk scenes returned is greater than that of the corresponding vision only model (6.1.1.3).

		Predicted	
		Non-dunk	Dunk
Actual	Non-dunk	3,490	22
	Dunk	14	2

Table 6.16: Confusion matrix produced when the model developed as part of experiment MS-4 was applied to the training set.

Looking at the performance metrics highlighted in table 6.17, we can see that this model operated with an MCP of 99.19%, which equals that of the comparable computer vision model. However, we observed that this model edged out the computer vision model in terms of MCR with a score of 98.99% (compared to 98.05%).

	Precision	**Recall**	**TPR**	**TNR**	**Support**
Non-dunk	0.9960	0.9937			3,512
Dunk	0.0833	0.1250			16
Total/Avg	0.9919	0.9899	0.1250	0.9937	3,528

Table 6.17: Summary of model metrics from experiment MS-4.

While this model did a good job at increasing the relevance of positive samples retrieved when compared to the computer vision model, the false positive rate did not decrease to the same extent that we saw in the multi-sensory model described in section 6.1.2.2. Finally, we can see from figure 6.7 that there were no signs of underfitting in the model.

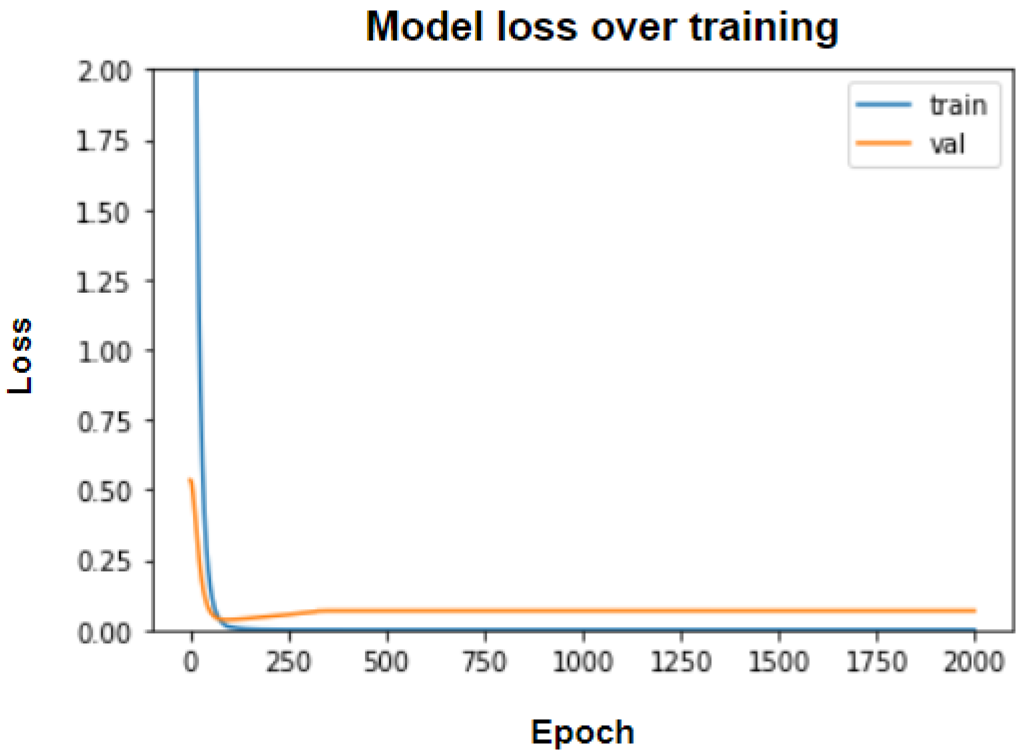

Figure 6.7: Illustration of training and validation loss over model training for the model described in experiment MS-4.

6.1.2.5 Other multi-sensory experiments

Similar to section 6.1.1.4, here we briefly summarise some of the other multi-sensory models experimented with as part of this dissertation. While we don't delve into these experiments in as much detail as the others discussed in this section, they could provide starting point for future research and further model refinement. A summary of each model's performance on the testing set is available in table 6.19.

Experiment MS-5: In this experiment we explored the impact of using filter bank spectrograms plotted with the viridis colour map as part of our audio stream as opposed to the MFCC spectrograms described above. In order to allow for meaningful comparison between the models we simply retrained the model described in experiment MS-2 (section 6.1.2.2) with the filter bank viridis spectrograms rather than designing a new architecture.

Taking a look at the performance we can see that this model correctly classified 1 out of 15 slam dunk sequences (TPR = 6.25%), while correctly classifying 3,508 out of 3,512 non-slam dunk scenes (TNR = 99.88%). We also observed a slam dunk class precision of 20% and a non-slam dunk class precision of 99.57% averaging to an MCP of 99.12%.

Experiment MS-6: Once again, to allow us to compare the performance of the hot and viridis colour maps against each other we decided to retrain the audio component of the model described in experiment MS-5 using the filter bank hot spectrograms. Examining the results summary we can see that this model, as with MS-5 correctly classified only 1 out of 15 slam dunk scenes (TPR = 6.25%). However, unlike the viridis model, this model misclassified 8 non-slam dunk scenes (TNR = 99.77%). Finally, in this experiment we observed slam dunk class precision of 11.11% and a non-slam dunk class precision of 99.57% for an MCP of 99.12%.

Experiment MS-7: In this experiment we decided to test the effects of performing pre-class score fusion between the visual and audio streams of our architecture instead of performing fusion after each stream had made a prediction (as was the case in each model discussed above).

To achieve this we removed the final dense layer from both our audio and visual streams, this meant that each model outputted a feature map in the form of a 128 dimensional vector. These vectors are combined into a single 128 dimensional vector in our fusion layer by taking the mean of the two vectors.

For this experiment we once again used the 128 hidden unit LSTM as the computer vision component of the model (CS-1) while we trained the audio stream using the MFCC viridis spectrograms.

This model correctly classified 3 out of 16 slam dunk scenes (TPR = 18.75%) while also classifying 3,476 non-slam dunk scenes correctly (TNR = 98.97%). In terms of the relevance of samples returned, this model obtained a precision score of 7.69% for the slam dunk class and 99.62% for the non-slam dunk class, this resulted in an MCP score of 99.02%.

Experiment MS-8: This experiment tested the classification power of the model described above in experiment MS-7 when the audio component is trained using MFCC hot spectrograms instead of MFCC viridis.

Similar to other viridis versus hot comparisons, we noticed that this model exhibited inferior performance when it came to classifying slam dunk scenes, being unable to classify a single slam dunk scene correctly (class precision & recall = 0%). The model was able to correctly classify 3,503 non slam dunk scenes correctly for a TNR of 99.74% and a class precision score of 99.54%.

Experiments MS-9 & MS-10: Continuing with the idea of pre-class score fusion and our desire to compare MFCC and filter bank spectrograms, in these two experiments we tested the performance of a pre-class score fusion model in which the audio portion was trained on viridis filter bank spectrograms (MS-9) and hot filter bank spectrograms (MS-10).

In experiment MS-9 we observed that the viridis filter bank model correctly classified 1 slam dunk scene (TPR = 6.25%), while misclassifying 4 non-slam dunk scenes as slam dunks (TNR = 99.86%). Similarly, the model developed in MS-10 using hot filter bank spectrograms classified 1 slam dunk scene correctly (TPR = 6.25%), while misclassifying 10 non-slam dunk scenes as slam dunks (TNR = 99.72%).

Experiments MS-11 & MS-12: In these experiments we explored the idea of using a support vector machine (SVM) within the fusion component of the model instead of a dense neural network. Once again, the computer vision component of this architecture was the 128 hidden unit LSTM model, while the audio stream was trained using the MFCC viridis spectrograms.

In experiment MS-11 we applied the SVM to a concatenation of the output vectors from each model (a 4 dimensional vector). Conversely, in experiment MS-12 we concatenated the outputs from the penultimate layers of both models to create a 256 dimensional vector which would then be analysed by the SVM component (illustrated in figure 6.8).

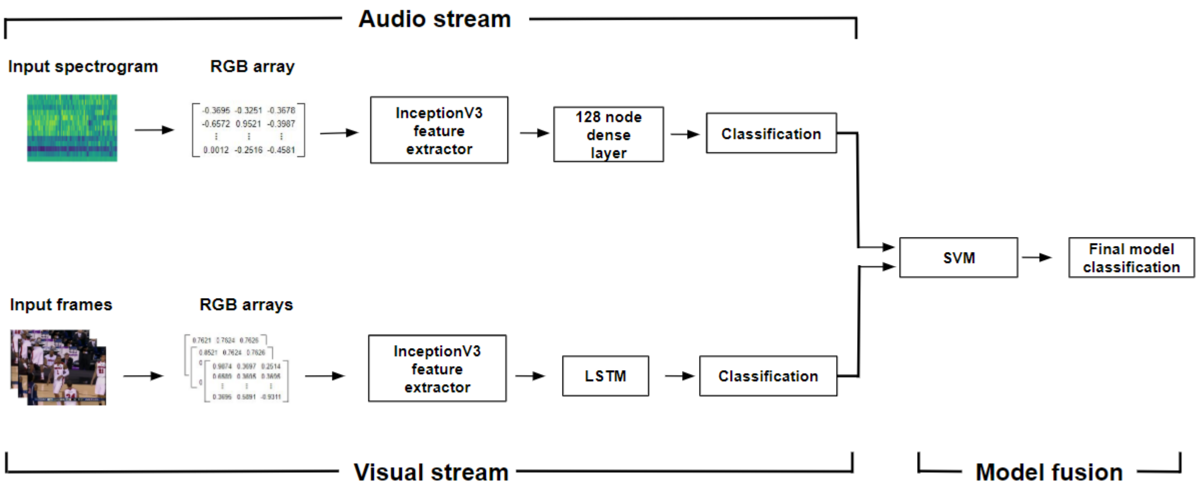

Figure 6.8: Illustration of updated architecture required to incorporate SVM component.

In both cases, we used a grid search algorithm [2] to determine the models optimum hyper-parameters (detailed in table 6.18)

Model ID	Cost	Kernal	Degree
MS-11	1	Polynomial	2
MS-12	10	Polynomial	5

Table 6.18: Summary of hyper-parameters used in SVM experimentation.

However in spite of the grid search algorithm, neither model was able to yield any successful results. The model trained in MS-11 was unable to classify a single slam dunk correctly, while classifying each non-slam dunk correctly. Similarly, the model developed in MS-12 was unable to classify any slam dunk scenes correctly while also misclassifying 5 non-slam dunk scenes as slam dunks.

These results suggest that the decision boundary of the training data was too complex to model using our simple SVM implementation.

Experiment MS-13: For this experiment we returned to implementing class score fusion, however instead of fusing the two dimensional vectors by taking the mean

[2] Python implementation of grid search algorithim: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html

between them, we decided to sum these vectors together. The vision stream of this model was the 128 hidden unit LSTM model which we are familiar with by now, while the audio stream was trained using the MFCC viridis spectrograms.

Interestingly, this model performed identically to the mean fusion model described in MS-2 correctly classifying 2 slam dunk scenes (TPR = 12.5%) while correctly classifying 3,510 non-slam dunk scenes (TNR = 99.43%).

Experiments MS-14 & MS-15: In these two experiments we toyed with the idea of using the VGG-19 pre-trained architecture instead of the InceptionV3 model within the audio stream. As far as recoding went, Keras made this switch quite simple and the only update required by us was to reshape each spectrogram to 224x224 pixels, as required by the models input layer. For this architecture we once again utilised a pre-class score mean fusion layer (similar to experiments MS-7 & MS-8).

For comparative purposes we tested this approach using both MFCC viridis and MFCC hot spectrograms. Neither model performed well at identifying slam dunk scenes with both models being unable to identify a single slam dunk scene correctly. The MFCC viridis model misclassified 6 non-slam dunk scenes (TNR = 99.88%) while the MFCC hot model misclassified only 2 out of 3,512 non-slam dunk scenes (TNR = 99.94%).

Model ID	TN	FP	FN	TP
MS-5	3,508	4	15	1
MS-6	3,504	8	15	1
MS-7	3,476	36	13	3
MS-8	3,503	9	16	0
MS-9	3,507	5	15	1
MS-10	3,502	10	15	1
MS-11	3,512	0	16	0
MS-12	3,507	5	16	0
MS-13	3,510	2	14	2
MS-14	3,506	6	16	0
MS-15	3,510	2	16	0

Table 6.19: Summary of confusion matrices produced when models described in the other multi-sensory experiments section were applied to the testing set.

6.2 Model comparison and evaluation

In this subsection we formally compare the performance metrics of our computer vision and multi sensory models while also testing these results for statistical significance. Although we experimented with a number of different multi-sensory architectures in section 6.1.2, we will only formally test the hypotheses defined in chapter 1 on what we deemed to be the two most effective multi-sensory architectures. The reason for this is simply that many of the multi-sensory models did not work when it came to classifying slam dunk scenes. To ensure the comparisons made in this section are fair and meaningful, each multi-sensory model tested is only compared to its corresponding computer vision model.

In addition to this, we also briefly compare the performance metrics from multi-sensory architectures trained on different representations of the audio features. While this was not something we set out to test or prove when we started this dissertation, it was something that intrigued us early on when performing our literature review and continued to fascinate us as we carried out our multi-sensory experimentation.

All model comparisons performed in this section are tested with a significance level of $\alpha = 0.05$.

6.2.1 Computer vision vs. Multi-sensory models

6.2.1.1 128 node LSTM vs. MFCC viridis

Table 6.20 summarises the key performance metrics for both the 128 hidden unit computer vision model developed in experiment CV-1 (section 6.1.1.1), and the multi-sensory architecture developed in experiment MS-2 (section 6.1.2.2).

Examining this summary we can see that the multi-sensory model outperformed the computer vision model in MCP, MCR, and TNR while the computer vision model outperformed the multi-sensory when it came to identifying the most slam dunk scenes (TPR).

Model ID	**MCP**	**MCR**	**TPR**	**TNR**
CV-1	0.9929	0.9549	0.4375	0.9573
MS-2	0.9937	0.9955	0.1250	0.9994

Table 6.20: Performance comparison between models developed in experiments CV-1 and MS-2.

With regard to statistical significance, we calculated the McNemar's test statistic using the contingency table described in table 6.21. This produced a test statistic value of $\chi^2 = 131.79$ which corresponded to a p-value < 0.001. This indicated that the differences in errors observed between the two models were indeed statistically significant. However, since this multi-sensory model did not out perform the computer vision model in all 4 of the performance metrics defined in our hypothesis, we failed to reject H_0.

		Multi-sensory model	
		Correct	Incorrect
Vision model	Correct	3,364	5
	Incorrect	148	11

Table 6.21: Contingency table used to calculate McNemar's test statistic when comparing models CV-1 and MS-2.

6.2.1.2 256 node LSTM vs. MFCC viridis

In this comparison, we compared the 256 hidden unit LSTM model from experiment CV-3 (section 6.1.1.3) to its related multi-sensory model trained using MFCC viridis spectrograms from experiment MS-4 (section 6.1.2.4).

Examining the comparisons in table 6.22 we can see that contrary to our previous model comparison, the computer vision model outperformed the multi-sensory model in both MCP and MCR. Similarly however, the computer vision model demonstrated superior performance when it comes to TPR, while the multi-sensory model boasted a higher TNR.

Model ID	**MCP**	**MCR**	**TRP**	**TNR**
CV-3	0.9929	0.9919	0.1875	0.9841
MS-4	0.9919	0.9899	0.1250	0.9937

Table 6.22: Performance comparison between models developed in experiments CV-3 and MS-4.

Once again, we calculated the McNemar's test statistic ($\chi^2 = 29.26$) using the contingency table below described in table 6.23. This corresponded to a p-value $<$ 0.001, indicating a statistically significant difference between the two models. However, since in this case, our multi-sensory only outperformed the computer vision model in only one of the 4 key performance metrics we safely failed to reject H_0.

		Multi-sensory model	
		Correct	Incorrect
Vision model	Correct	3,458	1
	Incorrect	34	35

Table 6.23: Contingency table used to calculate McNemar's test statistic when comparing models CV-3 and MS-4.

6.2.2 Viridis vs. Hot colour maps

Here we compared the performance of mutli-sensory models trained using viridis spectrograms to that of an identical model trained using hot spectrograms. To perform this comparison, we used the 128 node LSTM model from experiment CV-1 and compared the performance of 4 of the multi-sensory models trained using different audio spectrograms on top of this. These 4 models multi-sensory were trained using:

1. MFCC viridis spectrograms (MS-2/section 6.1.2.2)
2. MFCC hot spectrograms (MS-3/section 6.1.2.3)
3. Filter bank viridis spectrograms (MS-5/section 6.1.2.5)
4. Filter bank hot spectrograms (MS-6/section 6.1.2.5)

It's worth noting here that in order for us to make meaningful, and fair comparisons we only compared the MFCC and filter bank spectrograms to each other (i.e. we did not compare MFCC viridis to filter bank viridis)

Table 6.24 summarises the performance metrics for each of the 4 models analysed in this section. We can see from this table that the multi-sensory model trained using MFCC viridis spectrograms outperformed an identical model trained using MFCC hot spectrograms in all 4 of our key performance metrics.

Similarly we can see that the multi-sensory model trained using filter bank viridis spectrograms outperformed its corresponding hot spectrogram model in 3 out of 4 performance metrics (MCP, MCR, and TNR, mean class precision & recall), while the two models exhibited identical performance in terms of TPR.

MFCC comparison

Model ID	**MCP**	**MCR**	**TPR**	**TNR**
MS-2	0.9937	0.9955	0.1250	0.9994
MS-3	0.9909	0.9946	0.000	0.9914

Filter bank comparison

Model ID	**MCP**	**MCR**	**TPR**	**TNR**
MS-5	0.9921	0.9947	0.0625	0.9989
MS-6	0.9917	0.9875	0.0625	0.9977

Table 6.24: Comparison between MFCC viridis/hot multi sensory models (top) and Filter bank viridis/hot multi-sensory models (bottom).

The results described above suggest that multi-sensory models trained using audio spectrograms plotted using the viridis colour pallet outperform identical models trained using hot spectrograms. However, when we performed McNemar's test for statistical significance we found that there was no statistical difference between the model errors.

In the case of the MFCC comparison we observed a p-value $= 0.3712$ ($\chi^2 = 0.8$), while in the filter bank comparison we calculated a p-value $= 0.3865$ ($\chi^2 = 0.75$).

Since neither comparison generated a result which was deemed to be statistically significant ($\alpha = 0.05$), we cannot say for certain that the viridis colour pallet outperforms the hot pallet, however this could be an area for further research. Both contingency tables used to calculate the McNemar's test statistic are illustrated in table 6.25.

		Viridis model	
		Correct	Incorrect
Hot model	Correct	3,508	1
	Incorrect	4	15

		Viridis model	
		Correct	Incorrect
Hot model	Correct	3,501	4
	Incorrect	8	15

Table 6.25: Contingency tables used to calculate statistical significance for MFCC (top) and filter bank (bottom) spectrogram comparisons.

6.2.3 MFCC vs. Filter banks features

As alluded to in sections 3.4 & 4.3, we were interested in exploring the performance differences between multi sensory models trained using MFCC audio features and identical models trained using filter banks features.

Once again, to enable meaningful comparison, we decided to examine two sets of models (4 models in total) in which every component apart from the type of audio features used were identical. These models were the same models compared in the previous section (6.2.2), except rather than comparing the viridis models against the hot models for each feature type, we comapred the MFCC models to the filter bank models for each colour pallet:

1. MFCC viridis spectrograms (MS-2/section 6.1.2.2) vs. Filter bank viridis spectrograms (MS-5/section 6.1.2.5)

2. MFCC hot spectrograms (MS-3/section 6.1.2.3) vs. Filter bank hot spectrograms (MS-6/section 6.1.2.5)

When we examine the performance metrics for the two comparison detailed in table 6.26, we can see what appears to be contradicting results. In the case of the viridis comparison we see the MFCC model (MS-2) outperformed the filter banks model (MS-3) in each of the 4 key performance metrics. However, in the second comparison, we can see that the filter banks hot model out performed it's corresponding MFCC hot model in 3 out of the 4 performance metrics (MCP, TPR, & TNR).

Viridis comparison

Model ID	**MCP**	**MCR**	**TPR**	**TNR**
MS-2	0.9937	0.9955	0.1250	0.9994
MS-5	0.9921	0.9947	0.0625	0.9989

Hot comparison

Model ID	**MCP**	**MCR**	**TPR**	**TNR**
MS-3	0.9909	0.9946	0.000	0.9914
MS-6	0.9917	0.9875	0.0625	0.9977

Table 6.26: Comparison between MFCC/filter banks models using viridis spectrograms(top) and hot spectrograms (bottom).

Perhaps unsurprisingly given the contradictory results above, McNemar's test suggested that both sets of models do not significantly differ from one another. In the case of the viridis comparison, McNemar's test yielded a p-value $= 0.4497$ ($\chi^2 = 0.57$), while the hot comparison observed a p-value $= 0.3865$ ($\chi^2 = 0.75$). As a result of this, we can say that there is no statistical evidence to suggest a difference in performance based on the type of audio features used (MFCC vs. Filter banks) within the context of ***this*** study.

		MFCC model	
		Correct	Incorrect
Filter banks model	Correct	3,507	2
	Incorrect	5	14

		MFCC model	
		Correct	Incorrect
Filter banks model	Correct	3,501	4
	Incorrect	8	15

Table 6.27: Contingency tables used to calculate statistical significance for viridis (top) and hot (bottom) audio feature comparisons.

6.3 Discussion

This study set out to test the hypothesis that a model which analyses both the audio and visual information from broadcast basketball footage would yield a more effective classifier (based on MCP, MCR, TRP, % TNR) for slam dunk scenes when compared to a similar model which analyses the visual information alone. To test this hypothesis we developed a total of 6 computer vision models and 15 multi-sensory models.

Our computer vision experimentation observed differences in classification power based on the form of pooling used in the final layer of the feature extractor (maximum versus mean pooling) as well as the number of hidden units within the LSTM portion of the model. From these experiments we determined that the two most effective classifiers (and most appropriate models to compare the multi-sensory classifiers against) were then 128, and 256 unit LSTM models which utilised mean pooling in the final layer of the feature extractor. We deemed these two models to be the most effective as these models exhibited superior performance when it came to returning relevant positive samples in comparison to the other models developed.

For our multi-sensory models, we experimented with training audio models using different audio processing methods (MFCC/filter bank) and spectrogram colour pallets (viridis & hot). We also explored different forms of fusion layers (pre-classification, post-classification, and even SVM's) and pre-trained CNN-architectures (InceptionV3 & VGG-19). From these experiments it appeared that multi-sensory architectures trained using MFCC viridis spectrograms were the most effective.

When it came to formally comparing the models, we found that in both cases the multi sensory model outperformed the computer vision model in terms of TNR, and even in one case both MCP and MCR 6.2.1.1. However, neither multi-sensory model demonstrated superior performance over all 4 of the key performance metrics defined in our hypothesis. As a result of this, and although there was shown to be a statistically significant difference between the errors of the multi-sensory and computer vision models, we failed to reject the null hypothesis (H_0).

In addition to this, we explored whether the colour pallet (viridis versus hot), or audio features (MFCC versus filter banks) used as part of the audio stream had an effect on the classification power of the multi-sensory model. While our experimentation showed that the MFCC viridis model always outperformed whichever model it was compared against, we could not deem these results to be statistically significant. It's worth caveating this with the fact that this study was performed on a relatively small dataset with very few positive samples, and given that the positive samples typically drove the differences between these models. It is perhaps a point for future development/research into this area to rerun these tests on a larger dataset with more positive samples to explore whether this can produce a statistically significant difference between the models.

It's also worth nothing that since the comparisons performed between the colour pallets and audio features stemmed from intrigue when performing the literature review, and to avoid HARKing[3] (Kerr, 1998), we chose to exclude these from our main research hypothesis but include them in this dissertation to inform readers of future avenues to explore (with more data).

[3] Hypothesising after results are known

Finally, despite failing to reject the null hypothesis, we believe that the preprocessing steps and models described in this dissertation provide a building block for future research into multi-sensory architectures for scene detection in basketball and other sports. In particular we would like to encourage others to explore the 128 node multi-sensory model described in experiment MS-2 (section 6.1.2.2). This model outperformed its corresponding computer vision model in all but TPR, and it would be interesting to see how this model could be improved with exposure to additional (real) positive samples in the training set.

Chapter 7

Conclusion

7.1 Problem definition & research overview

There has been a significant amount of research put into the application of computer vision and audio classification techniques for sports scene classification. While these are certainly interesting and exciting applications of these techniques, much of the research in this area, particularly relating to the sport of basketball focuses solely on building classifiers based on information from a single sensory system (audio ***or*** visual).

The primary objective of this dissertation was to determine if a multi-sensory deep learning architecture which analyses the wealth of information contained in both the audio and visual features of basketball broadcast footage could yield a more effective key event classification system when compared to a similar single sense (video) model.

The findings made in this dissertation were achieved by following the steps laid out below:

- Performing a comprehensive review of existing literature relating to the research problem.
- Identifying, acquiring, and preparing an appropriate dataset for this study.
- Developing a baseline computer vision model.

- Extending the baseline model to a multi-sensory (audio-visual) model.
- Performing a statistical comparison and evaluation of the models performance on a testing dataset.

7.2 Experimentation, evaluation & results

In order to test the research hypothesis detailed in section 1.6 we developed a total of 6 computer vision models, and 15 multi-sensory models.

Our computer vision models analysed a series of sub-sampled frames extracted from 4 second long scenes within our dataset through a simple hybrid network architecture. On the other hand, the multi-sensory models were comprised of a visual stream (identical to the computer vision model), and an audio stream which analysed the scene audio in the form of a spectrogram. Predictions from each stream were then combined in a fusion component which produced the final output for the model.

We tested each model on the same testing dataset (a 30% subset of the full dataset) and gathered the mean class precision, mean class recall, true positive rate, and true negative rate for each model. Once these performance metrics were collected, we were able to compare the performance of the multi-sensory models to their corresponding computer vision model.

From here, we used McNemear's test to analyse the differences between the errors of the two model types. We compared this test statistic to the χ^2 distribution with 1 degree of freedom and a significance level of $\alpha = 0.05$ to determine if there was a statistical difference between the results observed between the computer vision and multi-sensory models.

Although we ultimately failed to reject our H_0 since none of our multi-sensory models managed to outperform the computer vision model in all 4 metrics, the results our experimentation suggest that there is significant potential for the use of multi-sensory networks in the domain of sports video segmentation.

Our experimentation results demonstrated that multi-sensory networks can improve the mean class precision, mean class recall, and true negative rate of a slam dunk scene classifier when compared to a single sense model. Although not outperforming the computer vision model in true positive rate, the multi sensory model retrieved a much higher proportion of relative positive (slam dunk) samples. It is also not unreasonable to muse over how the multi-sensory models performance could be improved by the introduction of additional positive samples to the dataset.

In addition to testing our research hypothesis, we also explored the impact on the performance of multi sensory networks trained using different representations of the audio features. These experiments explored differences in MFCC vs. Filter bank approaches as well as determining if the colour map used when plotting the audio spectrograms had an effect on classification power. While these comparisons suggested that MFCC viridis representations of the audio generated the best results, the differences between the models were not found to be statistically significant.

The experiments designed and results gathered as part of this dissertation were limited by the following key factors:

- The quality of audio and video footage within the dataset
- The relatively small sample size of 10 games analysed within the study
- Class imbalance and the lack of a large amount of positive samples
- Hardware and time restrictions limited the number of epochs our models could be trained over and the extent to which we could retrain the InceptionV3 and VGG19 architectures

7.3 Contributions and impact

The work performed in this dissertation has contributed to the existing body of knowledge by demonstrating that even on a small sample size of 10 games, a multi-sensory deep learning architecture can outperform a similar computer vision model in terms of mean class precision/recall and true negative rate when classifying the occurrence of slam dunks within broadcast basketball footage.

While this dissertation focused only on the classification of slam dunk scenes within the sport of basketball, the methodology outlined in this dissertation provides a scalable building block for further development of these models which could include identifying more event types within the game of basketball, or indeed identifying events in other sports. This further development could go a long way in revolutionising how sports teams at the highest level collect, annotate, and analyse footage of their opponents.

7.4 Future work & recommendations

While conducting research and experimentation as part of this dissertation there were a number of avenues for potential future research that became obvious to us. Unfortunately we were not able to explore these avenues due to time constraints, hardware limitations, and the ideas relevance to the research question. These ideas and recommendations are outlined below:

Improving the existing models:
Upon reviewing the research performed as part of this dissertation and the limitations of that research, we have identified 4 key actions which could improve the performance of the multi-sensory architectures developed.

The first of these actions is to acquire more data to train the models on. This is perhaps a simple task if one's hardware permits since this study focused on a subset of data collected and annotated by Ramanathan et al. (2016). This dataset contains annotations for 257 games and future researchers could look to retrain the models described in this dissertation on the full dataset.

Secondly, a major limitation of this study mentioned in section 7.2 is the quality of the audio/video footage which these models are trained on. For this reason we would advise that anyone looking to perform future work in this area attempts to collect/annotate footage of a higher recording quality than that used in this study. Obviously this is a very time consuming and difficult task, however we feel higher quality data could significantly improve the performance of these models.

In addition to this, we advise that anybody looking to perform future work on the models discussed in this dissertation should look to overcome the heavy class imbalance within the dataset by introducing more real positive samples before applying techniques such as oversampling and SMOTE. We feel that access to more real life examples of slam dunks would help the models generalise better to unseen data. Perhaps one could start by using footage from one of the many slam dunk highlight packages available online.

Finally, we would recommend that anybody who is interested in improving on the models developed in this dissertation should look at training these models over a larger number of epochs and retraining some of the later convolution layers in the InceptionV3 and VGG19 models. This of course would require additional time/processing power but could significantly improve the performance of the models, particularly the audio component of the multi-sensory architecture.

Identifying additional key events:

While this study focused on classifying the occurrence of slam dunk scenes exclusively, a logical next step would be to expand and retrain the models described in this dissertation to classify a wider range of key events. Considering the fact that the dataset created by Ramanathan et al. (2016) contains annotations for a number of events, one could use this dataset and retrain the current models to identify the occurance of successful jump shots and layups.

Expanding the current models to different sports:

Although this study focused solely on the sport of basketball, a potential avenue for future research would be assessing the performance of multi-sensory networks at key event classification in other sports. While the possibilities here are countless, some ideas include:

- Goal scene classification in Gaelic football, hurling, and soccer

- Touchdown scene classification in American football
- Highlight detection in golf

Applying multi-sensory architectures to different forms of data:
The multi-sensory models developed as part of this dissertation analyse information through 2 different sensory streams (audio and visual) before producing a final classification. While the audio and visual information certainly appears different to us as humans, they can simply be described as two sets of features which describe the same event.

For this reason we recommend that some future research should explore the application of multi-sensory networks across different forms of data. A simple experiment could be to develop a multi-sensory network for predicting house prices in which stream A of the network makes a prediction based on the size of the house, number of beds, etc. While stream B analyses features to do with the area in which the house is located (number of schools nearby, crime rate, etc.). Outputs from these models could be combined in a fusion layer before a final prediction is made. The results from this model could be compared to a single sense network which analyses the entire feature set.

7.5 Closing thoughts

Multi-sensory deep learning architectures for slam dunk scene classification are an incredibly exciting application of deep learning, image/video classification and audio processing. The results observed in this study suggest that multi-sensory architectures demonstrate superior performance in terms of mean class precision/recall and true negative rate when compared to a single sense model.

While this study was unable to show that multi-sensory models could yield a greater true positive rate than a similar single sense model, our multi sensory model was capable of returning a greater proportion of relevant positive samples. In addition to this, we have identified a number of ways in which the current models could be improved for future research. We have also made recommendations on how these models could be expanded to other domains.

The models developed for this dissertation provide a scalable, dynamic building block for future work this area. Further refinement of these multi-sensory models could revolutionise sports video segmentation and change how sports teams approach video analysis.

References

Afonso, V. X., Tompkins, W. J., Nguyen, T. Q., & Luo, S. (1999). Ecg beat detection using filter banks. *IEEE transactions on biomedical engineering*, *46*(2), 192-202. doi: 10.1109/10.740882

Alsheikh, M. A., Niyato, D., Lin, S., Tan, H.-P., & Han, Z. (2016). Mobile big data analytics using deep learning and apache spark. *IEEE network*, *30*(3), 22-29. doi: 10.1109/MNET.2016.7474340

Amiriparian, S., Gerczuk, M., Ottl, S., Cummins, N., Freitag, M., Pugachevskiy, S., & Schuller, B. (2017). Snore sound classification using image-based deep spectrum. *In Interspeech 2017*, 3512-3516. doi: 10.21437/Interspeech.2017-434

Aytar, Y., Vondrick, C., & Torralba, A. (2016). Soundnet: Learning sound representations from unlabeled video. *In Advances in Neural Information Processing Systems*, 892-900. Retrieved from http://papers.nips.cc/paper/6146-soundnet-learning-sound-representations-from-unlabeled-video.pdf

Baccouche, M., Mamalet, F., Wolf, C., Garcia, C., & Baskurt, A. (2010). Action classification in soccer videos with long short-term memory recurrent neural networks. *In proceedings of the International Conference on Artificial Neural Networks*, 154-159. doi: 10.1007/978-3-642-15822-3_20

Ballan, L., Bazzica, A., Bertini, M., Del Bimbo, A., & Serra, G. (2009). Deep networks for audio event classification in soccer videos. *In Proceedings of the 2009 IEEE International Conference on Multimedia and Expo*, 474-477. doi: 10.1109/ICME.2009.5202537

Batista, G., Prati, R. C., & Monard, M. C. (2004). A study of the behavior of several methods for balancing machine learning training data. *ACM SIGKDD explorations newsletter*, *6*(1), 20-29. doi: 10.1145/1007730.1007735

Boddapati, V., Petef, A., Rasmusson, J., & Lundberg, L. (2017). Classifying environmental sounds using image recognition networks. *Procedia computer science*, *112*, 2048-2056. doi: 10.1016/j.procs.2017.08.250

Buda, M., Maki, A., & Mazurowski, M. A. (2018). A systematic study of the class imbalance problem in convolutional neural networks. *Neural Networks*, *106*, 249-259. doi: 10.1016/j.neunet.2018.07.011

Burns, B. D. (2001). The hot hand in basketball: Fallacy or adaptive thinking. *In Proceedings of the annual meeting of the Cognitive Science Society*, *23*, 152-157.

Campos, J. L., Butler, J. S., & Bulthoff, H. H. (2012). Multisensory integration in the estimation of walked distances. *Experimental brain research*, *218*(4), 551-565. doi: 10.1007/s00221-012-3048-1

Carvalho, T., De Rezende, E. R., Alves, M. T., Balieiro, F. K., & Sovat, R. B. (2017). Exposing computer generated images by eye's region classification via transfer learning of vgg19 cnn. *In 2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*, 866-870. doi: 10.1109/CVPR.2016.308

Chalapathy, R., Menon, A. K., & Chawla, S. (2019). Anomaly detection using one-class neural networks. *arXiv preprint arXiv:1802.06360*.

Cho, K., Van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using rnn encoder-decoder for statistical machine translation. *In proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 1724-1734. doi: 10.3115/v1/D14-1179

Chow, V., & Dibua, O. (2018). Action recognition in tennis using deep neural networks. *Stanford University CS230*, 1-6. Retrieved from http://cs230.stanford.edu/files_winter_2018/projects/6945761.pdf

Costa, Y. M., Oliveira, L. S., Koericb, A. L., & Gouyon, F. (2011). Music genre recognition using spectrograms. *In Proceedings of the 18th International Conference on Systems, Signals and Image Processing*, 1-4.

Davis, S., & Mermelstein, P. (1980). Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE transactions on acoustics, speech, and signal processing*, *28*(4), 357-366. doi: 10.1109/TASSP.1980.1163420

Dietterich, T. G. (1998). Approximate statistical tests for comparing supervised classification learning algorithms. *Neural computation*, *10*(7), 1895-1923. doi: 10.1162/089976698300017197

D'Orazio, T., & Leo, M. (2010). A review of vision-based systems for soccer video analysis. *Pattern Recognition*, *43*(8), 2911-2926. doi: 10.1016/j.patcog.2010.03.009

Douzas, G., & Bacao, F. (2018). Effective data generation for imbalanced learning using conditional generative adversarial networks. *Expert Systems with applications*, *91*, 464–471. doi: 10.1016/j.eswa.2017.09.030

Edwards, A. L. (1948). Note on the "correction for continuity" in testing the significance of the difference between correlated proportions. *Psychometrika*, *13*(3), 185-187. doi: 10.1007/BF02289261

Ernst, M., & Bulthoff, H. H. (2004). Merging the senses into a robust percept. *Trends in cognitive sciences*, *8*(4), 162-169. doi: 10.1016/j.tics.2004.02.002

Esteva, A., Kuprel, B., Novoa, R. A., Ko, J., Swetter, S. M., Blau, H. M., & Thrun, S. (2017). Dermatologist-level classification of skin cancer with deep neural networks. *Nature*, *542*(7639), 115-127. doi: 10.1038/nature21056

Fan, Y., Lu, X., Li, D., & Liu, Y. (2016). Video-based emotion recognition using cnn-rnn and c3d hybrid networks. *In Proceedings of the 18th ACM International Conference on Multimodal Interaction - ICMI 2016*, 445-450. doi: 0.1145/2993148.2997632

Gers, F. A., Schmidhuber, J., & Cummins, F. (1999). Learning to forget: Continual prediction with lstm. *9th International Conference on Artificial Neural Networks (ICANN '99)*, 850-855. doi: 10.1049/cp:19991218

Gilovich, T., Vallone, R., & Tversky, A. (1985). The hot hand in basketball: On the misperception of random sequences. *Cognitive Psychology*, *17*(3), 295- 314. doi: 10.1016/0010-0285(85)90010-6

Goldsberry, K. (2012). Courtvision: New visual and spatial analytics for the nba. *In Proceedings of the 2012 MIT Sloan sports analytics conference*, *9*, 12-15. Retrieved from http://www.sloansportsconference.com/wp-content/uploads/2012/02/Goldsberry_Sloan_Submission.pdf

Goldsberry, K., & Weiss, E. (2013). The dwight effect: A new ensemble of interior defense analytics for the nba. *In Proceedings of the 2013 MIT Sloan sports analytics conference*, *10*, 1-11. Retrieved from https://pdfs.semanticscholar.org/2928/057a2c680906c3333fc0a069fcc547a5f85c.pdf

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... Bengio, Y. (2014). Generative adversarial nets. *Proceedings of the International Conference on Neural Information Processing Systems (NIPS 2014)*, 2672-2680. Retrieved from http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf

Henz, M. (2001). Scheduling a major college basketball conference—revisited. *Operations research*, *49*(1), 163-168. doi: 10.1287/opre.49.1.163.1119

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, *9*(8), 1735-1780. doi: 10.1162/neco.1997.9.8.1735)

Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., & Fei-Fei, L. (2014). Large-scale video classification with convolutional neural networks. *In 2014 IEEE Conference on Computer Vision and Pattern Recognition*, 1725-1732. doi: doi.org/10.1109/CVPR.2014.223

Kerr, N. L. (1998). Harking: Hypothesizing after the results are known. *Personality and Social Psychology Review*, *2*(3), 196-217. doi: 10.1207/s15327957pspr0203_4

Kristan, M., Perš, J., Perše, M., & Kovačič, S. (2009). Closed-world tracking of multiple interacting targets for indoor-sports applications. *Computer Vision and Image Understanding*, *113*(5), 598-611. doi: 10.1016/j.cviu.2008.01.009

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, *60*(6), 84-90. doi: 10.1145/3065386

Langenkamper, D., van Kevelaer, R., & Nattkemper, T. W. (2018). Strategies for tackling the class imbalance problem in marine image classification. *International Conference on Pattern Recognition*, 20-36. doi: 10.1007/978-3-030-05792-3_3

Manevitz, L., & Yousef, M. (2007). One-class document classification via neural networks. *Neurocomputing*, *70*(7-9), 1466-1481. doi: 10.1016/j.neucom.2006.05.013

Manevitz, L. M., & Yousef, M. (2001). One-class svms for document classification. *Journal of machine Learning research*, *2*, 139-154.

Mi, Q., & Xue, D. (2018). A sound-based video clipping framework toward sports scenes. *Journal of Visual Communication and Image Representation*, *55*, 648-653. doi: 10.1016/j.jvcir.2018.07.008

Murray, G., Renals, S., Carletta, J., & Moore, J. (2006). Incorporating speaker and discourse features into speech summarization. *In proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, 367-374. doi: 10.3115/1220835.1220882

Nemhauser, G. L., & Trick, M. A. (1998). Scheduling a major college basketball conference. *Operations Research*, *46*(1), 1-8. doi: 10.1287/opre.46.1.1

Newell, F. N., Bulthoff, H. H., & Ernst, M. O. (2003). Cross-modal perception of actively explored objects. *In Proceedings of the Eurohaptics 2003 Conference*, 291-299. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.10.8186&rep=rep1&type=pdf

Niu, Z., Gao, X., & Tian, Q. (2012). Tactic analysis based on real-world ball trajectory in soccer video. *Pattern Recognition*, *45*(5), 1937-1947. doi: 10.1016/j.patcog.2011.10.023

Pauwels, E. J., & Ambekar, O. (2011). One class classification for anomaly detection: Support vector data description revisited. *In Proceedings of the 2011 Industrial Conference on Data Mining*, 25-39. doi: 10.1007/978-3-642-23184-1_3

Pawara, P., Okafor, E., Schomaker, L., & Wiering, M. (2017). Data augmentation for plant classification. *International Conference on Advanced Concepts for Intelligent Vision Systems*, 615-626. doi: 10.1007/978-3-319-70353-4_52

Piczak, K. J. (2015). Environmental sound classification with convolutional neural networks. *In 2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP)*, 1-6. doi: 10.1109/MLSP.2015.7324337

Ramanathan, V., Huang, J., Abu-El-Haija, S., Gorban, A., Murphy, K., & Fei-Fei, L. (2016). Detecting events and key actors in multi-person videos. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3043-3053. doi: 10.1109/CVPR.2016.332

Shams, L., Kamitani, Y., & Shimojo, S. (2000). Illusions: What you see is what you hear. *Nature*, *408*(6814), 788. doi: 10.1038/35048669

Shimojo, S., & Shams, L. (2001). Sensory modalities are not separate modalities: plasticity and interactions. *Current opinion in neurobiology*, *11*(4), 505–509. doi: 10.1016/S0959-4388(00)00241-5

Simonyan, K., & Zisserman, A. (2014). Two-stream convolutional networks for action recognition in videos. *In Advances in neural information processing systems*, 568-576. doi: 10.1016/j.cviu.2003.06.004

Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. *CoRR*, *abs/1409.1556*.

Spanhol, F. A., Oliveira, L. S., Petitjean, C., & Heutte, L. (2016). Breast cancer histopathological image classification using convolutional neural networks. *International Joint Conference on Neural Networks (IJCNN)*, 2560-2567. doi: 10.1109/IJCNN.2016.7727519

Staw, B. M., & Hoang, H. (1995). Sunk costs in the nba: Why draft order affects playing time and survival in professional basketball. *Administrative Science Quarterly*, 474-494. doi: 10.2307/2393794

Sudre, C. H., Li, W., Vercauteren, T., Ourselin, S., & Cardoso, M. J. (2017). Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations. *Deep learning in medical image analysis and multimodal learning for clinical decision support*, 240-248. doi: 10.1007/978-3-319-67558-9_28

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., . . . Rabinovich, A. (2015). Going deeper with convolutions. *In Proceedings of the IEEE conference on computer vision and pattern recognition*, 1-9. doi: 10.1109/CVPR.2015.7298594

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the inception architecture for computer vision. *In Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, 2818-2826. doi: 10.1109/CVPR.2016.308

Tzirakis, P., Trigeorgis, G., Nicolaou, M. A., Schuller, B. W., & Zafeiriou, S. (2017). End-to-end multimodal emotion recognition using deep neural networks. *IEEE Journal of Selected Topics in Signal*, *11*(8), 1301-1309. doi: 10.1109/JSTSP.2017.2764438

Urbano, J., Bogdanov, D., Herrera Boyer, P., Gomez Gutierrez, E., & Serra, X. (2014). What is the effect of audio quality on the robustness of mfccs and chroma features? *In proceedings of the 15th Conference of the International Society for Music Information Retrieval (ISMIR 2014); 2014 Oct 27-31; Taipei*, 573-578. doi: 10.1145/1873951.1874137

Weisstein, E. W. (2003). Convolution.

Yu, D., & Seltzer, M. L. (2011). Improved bottleneck features using pretrained deep neural networks. *In proceedings of the twelfth annual conference of the international speech communication association*, 237-240. Retrieved from https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/Bottleneck-Interspeech2011-pub.pdf

Yue-Hei Ng, J., Hausknecht, M., Vijayanarasimhan, S., Vinyals, O., Monga, R., & Toderici, G. (2015). Beyond short snippets: Deep networks for video classification. *In 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 4694-4702. doi: 10.1109/CVPR.2015.7299101

Zhang, X., Zou, Y., & Shi, W. (2017). Dilated convolution neural network with leakyrelu for environmental sound classification. *In proceedings of the 22nd International Conference on Digital Signal Processing (DSP)*, 1-5. doi: 10.1109/ICDSP.2017.8096153

Zheng, Y., Yang, C., & Merkulov, A. (2018). Breast cancer screening using convolutional neural network and follow-up digital mammography. *In proceedings at Computational Imaging III*, 4-18. doi: 10.1117/12.2304564

Zhong, Y., Sullivan, J., & Li, H. (2016). Face attribute prediction using off-the-shelf cnn features. *International Conference on Biometrics (ICB)*, 1-7. doi: 10.1109/ICB.2016.7550092

Appendix A

A summary of games analysed

Game ID	Year	Team A	Team B	YouTube ID
0	1996	Kentucky	UMASS	TdFsLf6NdA4
1	2005	West Virginia	Wake Forest	63ce3pgTihA
2	2009	Arizona	Louisville	DgsbjLL1ZwU
3	1996	Syracuse	Mississippi State	ahEnvgLvRI4
4	2000	Gonzaga	Purdue	nRfZM1def1k
5	2007	North Carolina	Georgetown	tjzSVn2WE6o
6	2003	Syracuse	Auburn	OVRp7_wN3Ys
7	2007	Ohio State	Florida	3gtm0aaBkxM
8	1992	Michigan	Duke	rBQ4-zr22Nc
9	1994	Florida	Duke	kJFBSK-qX58

Table A.1: Information about each game analysed as part of this dissertation. Games can be viewed by prefixing the YouTube ID with http://youtube.com/watch?v=