

Learning to Recognise Visual Content from Textual Annotation

Matt Marter

Submitted for the Degree of
Doctor of Philosophy
from the
University of Surrey



Centre for Vision, Speech and Signal Processing
Faculty of Engineering and Physical Sciences
University of Surrey
Guildford, Surrey GU2 7XH, U.K.

October 2018

© Matt Marter 2018

Abstract

This thesis explores how machine learning can be applied to the task of learning to recognise visual content from different forms of textual annotation, bringing together computer vision and natural language processing. The data used in the thesis is taken from real world sources including broadcast television and photographs harvested from the internet. This leads to very few constraints on the data meaning there can be large variations in lighting, facial expression, visual properties of objects and camera angles. These sources provide the levels of data required to support modern machine learning approaches. However, annotation and or ground truth are not available and potentially expensive to obtain. This work therefore, will employ weak textual annotation in the form of subtitles, scripts, captions and tags. The use of weak textual annotation means that different techniques are also required to handle the natural language that is used to describe the visual content.

Character identification is a challenge that requires a different approach due to the similarities that will be shared between all faces. As with location recognition, the script is aligned with the video using subtitles. Faces are detected using a face detector and facial landmarks are regressed. These facial landmarks are used to create a descriptor for the face. Multiple techniques are used to assign the faces identities from the script. In the first technique, facial descriptors are clustered to the number of characters and the size of the clusters matched with the screen time of the character. In the second technique, a random forest is trained to differentiate between different faces, and the splitting criteria are used to reduce the dimensionality of the facial features. The reduced dimensionality allows a distribution of facial features to be created per scene. Then rules are created to separate scenes and identify distributions for individual characters. As well as this, data harvested from the internet is used to learn the appearance of the actors in the video and then matched to the characters. Using the various techniques give a character labelling performance of up to 82.75% accuracy using a SIFT-based descriptor and up to 96.82% using a state of the art descriptor.

Automatic caption generation for images is a relatively new and complex topic as it requires both understanding of the visual content in the image and the formation of natural language. Deep learning is powerful for object recognition and provides excellent performance on image recognition data sets. Pretrained convolutional neural networks (CNNs) were fine tuned using the parts of speech (POS) extracted from the natural language captions. A probabilistic language model can be created from the captions in the training data and be used to recreate new sentences for unseen images. To better model more complex language rules, a recurrent neural network (RNN) is used to generate sentences directly from features extracted from a CNN. An RNN that uses attention to look at different parts of an image can also utilise the final layers of a CNN to provide context for the whole image.

Location recognition, character identification and RNNs are combined to automatically generate descriptions for broadcast television using character and location names. This creates a full pipeline for automatically labelling an unseen episode of a television series. Compared with ground truth input of location and characters, only a small drop in performance occurs when using labels predicted by computer vision and machine

learning techniques. Using ground truth, a CIDEr score of 1.585 is achieved compared with 1.343 for a fully predicted system.

Data providing emotional context for words and images allow the RNN to be used to manipulate the emotional context for images. Subjective testing shows that the output captions are more emotive than captions generated without emotional context 74.85% of the time, and are almost equal to human written captions. Adjusting the emotional context is shown to generate captions that alter the content to reflect the emotion. The fusion of computer vision and natural language processing through machine learning represents an important step for both fields.

Key words: Deep Learning, Natural Language Processing, CNN, RNN, Face Recognition, Face Regression, Location Recognition, Random Forests, Weak Supervision, Caption Generation, MSCOCO, Context, Attention, Emotion

Email: M.Marter@surrey.ac.uk

WWW: <http://www.eps.surrey.ac.uk/>

Acknowledgements

I want to thank my supervisors Prof. Richard Bowden and Dr Simon Hadfield for putting up with me for so long, as well as providing much needed help and support with my research.

I'd also like to thank my office mates, and other CVSSP members for making my time in the department much more enjoyable. I'll never forget the equally bizarre and hilarious conversations that our office had on a nearly daily basis.

Thank you to everyone who filled in my survey to help me get some extra results needed to complete my research. At least it was worth the time it took to fill it in!

A special thank you to all my family for always believing in me through the many years spent on my research and thesis writing. I want to dedicate this thesis to my Grandad Peter, who sadly passed away in 2013 before I could finish my thesis.

I want to thank all my colleagues at Sen for being understanding, and working ridiculously hard through the whole year. Despite the tight deadlines and all-nighters it has been great to work with you all.

Most of all, I want to thank my girlfriend, Noemie, who has pushed me to persevere with my thesis. Without her support and encouragement, I have no doubt I'd have been unable to make it through this incredibly challenging year. I love you!

Contents

1	Introduction	9
2	Literature Review	17
2.1	Textual Annotation	17
2.2	Face Recognition	18
2.2.1	Character Identification	21
2.3	Object Recognition and Caption Generation	22
2.3.1	Caption Generation	23
2.3.2	Emotion	27
3	Character Identification in Broadcast Video	31
3.1	Facial Landmark Regression and Feature Extraction	33
3.1.1	Regression	34
3.1.2	Feature Extraction	36
3.1.3	OpenFace feature extraction	36
3.2	Unsupervised Facial Feature Clustering	37
3.2.1	Script Processing	38
3.2.2	Clustering and Labelling	39
3.2.3	Results	40
3.3	Unsupervised Gaussian Mixture Models and Random Forests	43
3.3.1	Random Forests	46
3.3.2	Character Separation	46
3.3.3	Gaussian Mixture Model	47
3.3.4	Face Classification	49
3.3.5	Results	49

3.4	Weakly Supervised Learning from Web Data	56
3.4.1	Face Tracks	58
3.4.2	Data Collection	59
3.4.3	Random Forests	60
3.4.4	Results	61
3.5	Chapter Conclusion	64
4	Caption Generation	67
4.1	Noun and Verb Prediction	70
4.1.1	Language Processing	70
4.1.2	CNN fine tuning	71
4.2	Language Model	73
4.3	Sentence Prediction	74
4.3.1	Evaluation Metrics	74
4.3.2	Probabilistic Model Results	76
4.4	Language Analysis	78
4.5	Preposition Prediction	84
4.5.1	Results	89
4.6	Alternative approaches	89
4.7	RNN	91
4.7.1	Long Short Term Memory	93
4.7.2	Attention	95
4.7.3	General Context	96
4.7.4	Beam Search	97
4.8	Experiments	97
4.8.1	CNN Features	97
4.8.2	Context Performance	99
4.8.3	Beam Size	100
4.9	Performance Comparison	101
4.10	Vocabulary Analysis	101
4.11	Chapter Conclusions	104

5	Caption Generation for Broadcast Television	107
5.1	Broadcast Television Captioning	107
5.2	Dataset	108
5.3	RNN training	108
5.4	Friends Location Recognition	109
5.5	Character Identification	112
5.6	Description Generation	114
5.7	Results	114
5.8	Friends Examples	116
5.9	Chapter Conclusions	118
6	Caption Generation with Emotion	121
6.1	Representation of Emotion	121
6.2	Adjective Noun Pairs	122
6.3	Emotion RNN	123
6.4	Results	123
6.5	Subjective Evaluation	132
6.6	Qualitative Examples	135
6.7	Failure Cases	137
6.8	Emotional Captioning for Broadcast Television	138
6.9	Chapter Conclusions	139
7	Discussion	143
7.1	Discussion	143
7.2	Future Work	149
A	Location Recognition	153
A.1	Background	154
A.1.1	Location Matching	158
A.2	Script Alignment	158
A.2.1	Subtitles	159
A.2.2	Shot Cuts	160
A.3	Mosaic Creation	162

A.3.1 Compositing	162
A.3.2 Median filter	163
A.3.3 Optical Flow	164
A.3.4 Mosaic Results	165
A.4 Location Matching	168
A.5 Results	171
A.6 Chapter Conclusions	173
References	177

Nomenclature

SB	Shot Boundary
SBD	Shot Boundary Detection
SIFT	Scale-Invariant Feature Transform
RANSAC	Random Sample Consensus
XML	Extensible Markup Language
DTW	Dynamic Time Warping
FPS	Frames Per Second
ROC	Receiver Operating Characteristic
AUC	Area Under Curve
TPR	True Positive Rate
FPR	False Positive Rate
BoW	Bag of Words
BoVW	Bag of Visual Words
SVM	Support Vector Machine
LDA	Linear Discriminant Analysis
GMM	Gaussian Mixture Model

AAM	Active Appearance Model
PCA	Principal Component Analysis
CNN	Convolutional Neural Network
ReLU	Rectified Linear Unit
SGD	Stochastic Gradient Descent
RNN	Recurrent Neural Network
LSTM	Long Short-Term Memory
NLL	Negative Log Likelihood
ANP	Adjective Noun Pair
MSCOCO	Microsoft Common Objects in Context
POS	Part of Speech
HMM	Hidden Markov Model
BLEU	Bilingual Evaluation Understudy
ROUGE	Recall-Oriented Understudy for Gisting Evaluation
METEOR	Metric for Evaluation of Translation with Explicit Ordering
CIDEr	Consensus-based Image Description Evaluation
VSO	Visual Sentiment Ontology
CRF	Conditional Random Field
MIL	Multiple Instance Learning
ME	Maximum Entropy
FFT	Fast Fourier Transform
COCA	Corpus of Contemporary American English

LFW	Labeled Faces in the Wild
LBP	Local Binary Pattern
HOG	Histogram of Oriented Gradients

Symbols

S set of shots

M set of mosaics

m mosaic

ϕ mosaic function

OF optical flow function

d mosaic distance function

P set of 2D key points

F set of feature vectors

\mathbf{f} feature vector

ψ set of inliers

I set of images

i image

h homography

s shot

SB set of shot boundaries

P set of 2D key points

F	set of feature vectors
\mathbf{f}	feature vector
ψ	set of inliers
Γ	set of text
c	set of text
t	dialogue text
A	set of scene boundaries
U^t	set of text in a scene
U^i	set of images in a scene
U^s	set of shots in a scene
Π	characters in a scene
Z	all characters
z	character
\mathbf{E}	co-occurrence of characters matrix
Υ	Set of face candidates
v	face
μ	cluster centres
M	characters in a scene
n	number of clusters
B	combination of characters
L	index of character combination
Q	frames containing a character combination

d	face image
D	set of face images
x	facial landmarks
X	set of facial landmarks
H	mapping
T	random displacements
W	classify a face
C	set of detections
p	positions of detections
Ψ	detector
o	detector threshold
r	position threshold
Y	face tracks
R	random forest
Ω	training data
Λ	label
Σ	covariance matrix
ϕ	gmm weightings
p	gmm
R	face candidate probability
W	Face classification function
Y	Set of captions

y Encoded caption

\mathbf{y} Word

C Caption length

k Vocabulary size

\mathbf{E} Word vector embedding matrix

m Word vector dimensionality

x RNN input

i input gate

f forget gate

o output gate

W Weightings

b Bias

c Cell value

h Hidden state

z Attention input

v Context input

S vector of parts of speech

T A set of words for each pos

l part of speech

q CNN training label

Chapter 1

Introduction

There is now a huge quantity of video and image content available, with only limited annotation and labels that allow this data to be retrieved from archives. The Internet Movie Database (IMDb)[1] contains details of 3 million episodes of television and over half a million films. Given the quantities, manual annotation is prohibitively expensive, which provides motivation for methods that are able to automatically annotate content. The challenges come from changes in lighting, camera position, facial position and the colour and scale of objects. Much of the annotation that is available takes the form of written descriptions of content. Learning therefore requires interpretation of natural language, which has many challenges of its own. There are many possible ways to convey the same information using natural language such as variation in word order and vocabulary. This thesis covers automatic character naming, and automatic caption generation with a focus on using natural language in the form of scripts, subtitles, and captions. To solve these different problems, a number of different techniques are used.

This introductory chapter aims to provide the motivation, aims, and constraints of the work presented in this thesis. The structure of the thesis is summarised along with details of the contributions that are made.

One of the most fundamental types of annotation needed to describe a scene is the location. Location recognition has many potential applications such as allowing people to search for videos and images of the same location. It can also be used to identify

the location of a given image when it is unknown. Automatically labelling locations in television shows and films could also be useful for people who have difficulties with their sight.

Scripted television and film contain a finite number of possible locations. Although many are shot in real world locations, a different location is often used as a proxy for practicality reasons. Hence, geographic location of sets and the movement between them may not follow the constraints of the real world. Some television shows, particularly sitcoms, use a small number of locations that appear very frequently such as the homes or places of work of the various featured characters. The rules of film making and cinematography provide some constraints to the types of shots that are used. A multi-camera setup is often used in sitcoms and soap operas, where multiple cameras record the same scene simultaneously. This allows multiple shots to be captured during a single take. Using a single-camera setup involves multiple takes and changes in the lighting for a scene. However, it provides more control of the visual style and is the prevalent technique for films and more dramatic television.

The scripts used in production contain valuable information that could be used for machine learning. The slug line at the start of each scene contains the location and time of day. It can provide information about shot transition such as fading (in or out). There is usually a short description of the actions that take place interleaved with the dialogue for each of the characters. There may also be some short prose that sets the scene. The script will be used to produce the final video but some changes often occur. The order of scenes may change, dialogue may differ, and various other changes can exist.

Annotating the characters in video is another useful type of annotation that allows easier retrieval of scenes containing a character. IMDb lists over 32 million acting credits in its database[1]. Face recognition can also be used to automatically label faces on social networking sites and in surveillance to identify people. Other security applications use it to identify a user, such as for automated passport terminals. As with location recognition, face recognition can aid people with limited vision when watching television shows or films. Figure 1.1 shows a frame from an episode of the sitcom



Figure 1.1: Example frame from the sitcom Friends showing multiple characters and their faces

Friends with multiple characters visible. It shows the wide variety of expressions and angles that are present, which exacerbate the recognition problem.

Faces all share common features so it is important to be able to identify which features can be used to differentiate between people. Features need to be robust to changes in facial expression, lighting, and camera angles. Footage taken a very long time apart can also introduce the effects of ageing, adding further complications to an already difficult task.

Locations and actors are important and useful for search, but for human uses such as audio description for the blind, these “tags” need to be converted to a meaningful description. The automatic generation of text descriptions for images is one of the most difficult tasks in computer vision, and has received a lot of attention recently. It can provide suggestions for captions when people upload their images or provide better ways to search for similar images. It is also useful for people who have difficulty seeing as it can provide a description of the image without needing annotation by a human.

Automatic caption generation combines the difficulties of recognising the contents of an image with the difficulties of generating natural language. Recognising an object is difficult when the same class can vary in colour, scale, size, camera view point, and

a very cute dog sitting by a bright monitor.
a dog, leaning against a laptop, looks gloomy.
a dog getting attention in front of a laptop
a dog that is sitting near a laptop.
dog close up in front of tv in room with candles



Figure 1.2: Example picture of a dog from MSCOCO with the provided text annotation

even shape. Images can contain people in various poses and performing various actions, which creates more challenges. The description will depend on many factors including the relative positions of objects and people as well as any other context. The intricacies of natural language and the subjectivity of annotators mean that multiple sentences could correctly describe the same image.

The MSCOCO[100] dataset provides around 80k training images and 40k validation images with an additional 40k test images. Each of the images is labelled with 5 text descriptions written by humans. In addition, there are full segmentations for 80 classes, split into various sub categories. A framework is provided for automatic evaluation of the quality of generated captions through a variety of different metrics. Figure 1.2 shows an image of a dog and a laptop. Even the ground truth captions vary in the words they use. One mistakes the laptop for a television and another uses the word “monitor”.

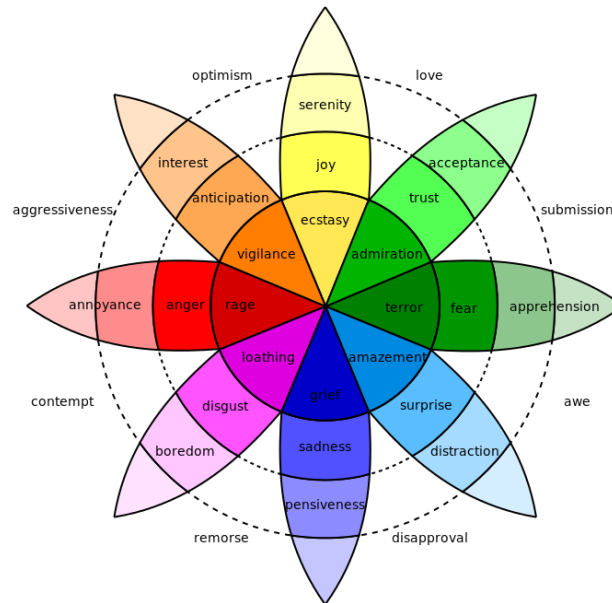


Figure 1.3: Plutchik’s wheel of emotions that provides a method to represent human emotions

One caption mentions candles that are a small feature in the back of the image, which even a human viewer may fail to notice as the dog and laptop are in the foreground. One caption uses the word “gloomy”, which relates more to how the image makes them feel. This is a good example where emotion of the annotator and subjectivity introduce further ambiguity into the interpretation and description process.

Emotion is a complex thing to represent, but Plutchik’s wheel of emotions [132] aims to provide a way to represent all the different emotions and how they are related. Plutchik’s wheel of emotions is shown in figure 1.3, it shows how the twenty four emotions can be divided into eight groups of related emotions with three levels of strength. The field of combining emotional understanding with computers is known as “affective computing”. Understanding emotion in images is a complicated task and is a relatively small area of research. It involves not only understanding what is occurring within an image, but also how it could make a person feel. Emotional annotation for images will allow people to search for images in a different way. It can also allow for AI to better interact with people by using language that reflects different emotions.

This thesis explores three core areas of content description, location, characters, and scene description. It then explores emotion as another factor in generating natural language descriptions. Chapter 2 provides a discussion of the state-of-the-art in the areas of face recognition, character naming, and automatic caption generation.

In chapter 3, various methods are explored for automatically labelling faces in broadcast video with character names. The dataset is introduced and then a method for regressing facial features and creating descriptions is presented. A state-of-the-art descriptor is also utilised to show how the descriptor influences the performance of the character identification methods. In the first method, clustering is used to divide the faces into characters. This is improved by using a random forest trained to identify faces to reduce the dimensionality of the face descriptors and represent them in a new space for separating identities. Then a Gaussian Mixture Model (GMM) is created for each combination of characters in a scene, and rules are created to automatically separate characters and classify face detections. Finally, the technique is modified to use data harvested from the web to train a random forest classifier on different actors.

Chapter 4 contains details of methods for automatically generating captions for images. Convolutional Neural Networks (CNNs) fine-tuned on MSCOCO with a probabilistic language model are used to generate captions. This is then improved by making use of an Recurrent Neural Network (RNN) to generate captions based on input from the CNNs. The RNNs can attend to different parts of an image to form the description. In addition, context provided from the final layers of a CNN are added to provide whole image context. The additional context input is shown to influence the vocabulary used by the RNN when generating captions.

Chapter 5 builds on the use of RNNs with context input in chapter 4, and combines them with the work in chapter 3 on character identification. This is used to create a method for automatically generating descriptions for broadcast television using character and location labels as an additional input. Finally, in chapter 6, emotional information is used as input to an RNN to influence the emotion of generated captions. These are compared with other automatically generated and human written captions for their emotional impact.

Chapter 7 summarises the work on the combination of computer vision, machine learning, emotion and text annotation. It also presents areas for potential future investigation to expand and improve the work.

In summary, this thesis contains work with the following contributions. In chapter 3, there is a technique for assigning character names to detected faces in broadcast video through the use of clustering without relying on speech. This was published in the paper “Friendly Faces: Weakly Supervised Character Identification” [108] . Following this is a method for reducing the dimensionality of face descriptors, and representing them in a space designed for separating identities. A random forest was trained to classify known people, and the forest output was used as a new descriptor. The descriptor is used to create a method for automatic character naming in broadcast video through the use of GMMs to model distributions and rules to separate combinations of characters. Chapter 3 also contains a method for automatically naming characters by learning the appearance of the actors using data harvested from the web.

Chapter 4 describes CNNs trained to recognise nouns and verbs from captions where the output of the noun network is used to influence the verb network. The next contribution is automatic caption generation using an RNN using attention with additional contextual input from the final hidden layers of a CNN.

Chapter 5 contains a complete description generation method for broadcast video using character and location information as additional input to the RNN.

Chapter 6 contributes the use of Adjective Noun Pairs (ANPs) and Plutchik’s wheel of emotion to create features that can be used with an RNN to change the emotional expression in generated captions.

Chapter 2

Literature Review

This chapter discusses the state-of-the-art in a variety of topics that are relevant to this thesis. They are divided into sections relating to the different topics. Script alignment is discussed as part of the background on aligning text annotation with a video. The second section discusses work relating to face regression, face recognition, other biometric features and character identification. The third section covers Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs) and caption generation. The final section covers the area of emotion in images and natural language generation.

2.1 Textual Annotation

The combination of video and text provides a lot of opportunities for computer vision and machine learning. An early example using close captions combined with videos to create automatic summaries was presented by Shahraray and Gibbon[144] in 1995. Gupta et al. [58] utilised text annotation and commentary to learn to classify images and actions in video.

Production scripts also contain annotation for a video but do not include any timing information. Everingham et al. [40][41] used the subtitles to provide timing information by matching it with the dialogue in the scripts. When there are inconsistencies with the script and subtitle information, Dynamic Time Warping (DTW) is used to find

the best alignment. When subtitle information is not available or there are scenes without dialogue, other methods need to be used for alignment. Sankar et al.[141] used a combination of location recognition, facial recognition, and speech to text to improve the alignment.

Scripts have been used in computer vision for a variety of tasks. Action recognition datasets such as “Hollywood” [91], “Hollywood 2” [107] and “Hollywood 3D” [59] used them to aid in the automatic extraction of actions. These datasets are considered “in the wild” and use data from commercially available films containing large amounts of variation in lighting, camera angles, people, style and settings. However, using scripts in this context merely extracts regions of the video in which an action is performed. The identity of who performs the action is irrelevant.

2.2 Face Recognition

For character identification, facial feature regression is useful as it allows faces to be described in ways that are invariant to changes in pose and expression. In 1998, Cootes[27] proposed a method for learning a mapping from facial image intensities to the pose of an Active Appearance Model (AAM). Matas et al. [110][78] track rigid objects using similar methods. The work of Ong and Bowden [121][122] took the sequential linear predictor proposed by Matas et al. and used it for regressing facial features. The sequential and hierarchical linear predictors were trained from Monte-Carlo samples so that facial features could be tracked in real time. Instead of using image intensities, Xiong and De la Torre [176] used Scale-Invariant Feature Transform (SIFT)[104] features. The use of SIFT features gives greater person independence when regressing. Ong treated each facial feature independently whereas Xiong used a single regressor for all feature points. Random forests have also been used for regressing facial features such as Cootes et al. [26][101] and Dantone et al. [31]. Wright and Hau [172] used trees to randomly project features and quantize them to form a histogram that can be used for face recognition.

Descriptors for face recognition are often created by extracting descriptors such as SIFT or Local Binary Pattern (LBP) [120] from facial landmarks. Five overlapping SIFT

descriptors are used by Sivic et al. [148]. Everingham et al.[40][41] also used SIFT features. In another paper, Sivic et al. [147] used Histogram of Oriented Gradients (HOG) features.

To better separate between identities, metric learning has been used to learn transformations to new spaces so that Euclidean distance is equivalent to the Mahalanobis distance. Examples of this are, Goldberger [55], Xing et al. [173], and Weinberger and Saul [168]. Wolf et al. [170] combined multiple descriptors including SIFT and LBP to create a specific similarity metric for a pair of images. Nguyen and Bai [119] used Cosine Similarity Metric Learning (CSML) to learn face verification from LBP, pixel intensity values, and Gabor filters. Huang et al. [28] combined LBP with a learned representation from a deep neural network. They used an information theory-based approach for metric learning. At the time, this achieved state-of-the-art performance on the Labeled Faces in the Wild (LFW) [69] challenge. Previously, CNN were used by Chopra et al. [23] to learn a similarity metric.

Taigman et al. [159] used a purely learned representation from a deep neural network and also provided improvements to face alignment using a 3D model. Sun et al. provided multiple improvements to the performance on the LFW challenge. This was achieved by using multiple CNNs [155], a Bayesian learning framework [156], and a different CNN architecture [157]. Schroff et al. [143] trained a CNN similar to GoogleNet [158] for directly learning an embedded representation. FaceNet achieved a 99.63% accuracy on the LFW challenge. Parkhi et al. [128] also presented a deep neural network-based approach that reaches a comparable level of performance along with a very large dataset that combined human input with automation.

In a video file, the same face can be tracked through multiple frames to create a face track. This gives more information than a single face descriptor. Various methods can be used to match face tracks. Barr et al. [8] surveyed different techniques for face recognition in video. They divided techniques into those that utilise the temporal information and those that treat the faces as an unordered set. When treating the frames as an unordered set, some techniques compare each frame in the track. In Sivic et al. [148][147], face tracks were matched by using the nearest neighbour descriptor

from one track to another. Wolf and Levy [171] used an Support Vector Machine (SVM) for comparing tracks. Apostolof and Zisserman used a random fern classifier over a track and used the maximum classification score in a track. They also summed the scores for each class along the track and used the largest sum to classify the tracks. Other techniques attempt to find a way to represent the entire track using some form of fusion. This can be done by creating a 3D model, such as Krüger et al. [86] and Zhou et al. [192]. Techniques used in other image recognition tasks can be applied to this task. Li et al. [96] created a Gaussian Mixture Model (GMM) representation, Sivic et al. [148] used a Bag of Words (BoW) feature, and Cui et al. [29] used sparse coding. Parkhi et al. [129] used a fisher vector representation for the face tracks.

Sequence-based approaches utilise the order of the frames to capture temporal information such as facial movement. Set-based approaches may be degraded by facial movement during a track, but sequence-based approaches actively use the additional information. One way of using the temporal information is with an Hidden Markov Model (HMM) such as the work of Liu and Cheng [103], Mitra et al. [115], Tistarelli et al. [163] and Eickeler et al. [38]. To extend the LBP features to include temporal information, Hadid and Pietikäinen [60] proposed the volume LBP feature that works in a 3D neighbourhood between frames. Using only 3 frames does not give enough temporal information so extended volume LBP features incorporated more frames. Some techniques attempt to unify the tracking and recognition process. When performed separately, a tracker may find images that do not work with the recognition method. Lee et al. [95] reported an improvement in recognition accuracy when combining tracking and recognition. Some techniques utilise only motion information rather than combined spatial and temporal information. Ye and Sim [183] measured how faces deformed from a neutral expression with a local deformation profile (LDP). The use of the motion information allows for robustness to changes in the face texture such as makeup.

In addition to face recognition, other methods exist to uniquely identify people. Zhang et al. [189] combined multiple methods to identify people including a full body CNN. Biometrics that can be used to separate people also include finger prints, iris, hand shape, vein patterns and measurements of body parts. It also extends to unique be-

havioural characteristics such as gait and voice. Not all of these are suitable for usage in video and may require special hardware to take measurements. Cunado et al. [30] automatically extracted gait from video and used it to recognise people. Hurley et al. [71] extracted ear features for biometric purposes.

2.2.1 Character Identification

Scripts have been used to label characters in broadcast video such as the work by Everingham et al [40][41]. They used mouth movement in the video to detect the speaker so they can be labelled using a time aligned script. They employed face descriptors and clothing colour histograms to learn character appearance and used this to label other face tracks without dialogue. Sivic et al. [147] labelled non-frontal faces exploring types of features. Cinbis et al. [25] used metric learning to separate cast members in an unsupervised manner. In Marter et al. [108] the face descriptors were clustered and then labelled using character co-occurrence information. Jin et al. [75] also used a clustering method but instead clustered tracks using a graph-based method. Clustered tracks were assigned a unique ID rather than a character name directly. Tapaswi et al. [162] clustered tracks and enforced rules that prevent clustering of tracks within a frame and linked tracks within a threading pattern of shots. Cour et al. [28] also used character co-occurrence and created bags of labels for faces, and attempted to resolve ambiguity by comparing these bags. Tapaswi et al. [161] used a Markov random field with face recognition, clothing appearance, and speaker recognition to create a probabilistic method for labelling characters. However, they used manually labelled training data rather than using script and subtitle alignment. Zhang et al. [190] used a hidden conditional random field to simultaneously cluster and label faces in videos. Bauml et al. [9] used a combination of labelled and unlabelled data to create a semi-supervised classifier. Bojanowski et al. [12] used a combination of face recognition and action recognition to label characters from the script. Ramanathan et al. [134] used natural language processing to label data where characters are referred to by something other than their name such as a pronoun.

2.3 Object Recognition and Caption Generation

Object recognition in computer vision has moved away from the use of engineered features such as SIFT and bag-of-words pipelines, and towards CNNs. LeCun et al. [94] used a CNN named LeNet for the task of hand-written character recognition in 1998. It used 32 by 32 input to a network of 6 layers with the early vision part of the network containing convolution layers followed by subsampling. The later layers were fully connected layers for classification before the output. The ImageNet [34] database uses the ontology of WordNet [45] and contains millions of images in thousands of classes. Krizhevsky et al. [85] applied CNNs to the ImageNet LSVRC-2010 contest using 1.2 million images in 1000 different classes. Their CNN design expanded on LeNet with larger convolutional layers and increased depth, and expanded to 3 channels for colour input. They also used dropout layers where neurons are randomly disabled to avoid overfitting during the training phase. Rectified Linear Units (ReLUs) are used as the non-linearities to increase training speed versus more traditional saturating non-linearities. Data augmentation is utilised in the form of random cropping, reflections and translations to increase the amount of training data as well as RGB shifting.

Simonyan and Zisserman [146] created even deeper networks with between 16 and 19 layers by using smaller (3 by 3) convolutional filters. The depth of the network meant that the network was initially trained with fewer layers and then retrained as more layers were added. The mean RGB value from the dataset is subtracted rather than the mean RGB image. Szegedy et al. [158] introduced GoogLeNet that contains 22 layers. The network introduced the Inception model that combined multiple convolution sizes with dimensionality reduction inspired by the Network in Network design by Lin et al. [99]. GoogLeNet achieved state-of-the-art performance in the 2014 ImageNet challenge that contained classes that would be hard for a human to distinguish.

For object detection, Girshick et al. [54] combined a CNN with the objectness measure from Alexe et al. [2]. The objectness score is used to find region proposals that potentially contain an object. Then a CNN is used to extract features that can be used to attempt to classify the detected object. This allowed multiple objects to be detected and classified within a single image, along with the object locations. He et al. [63]

improved performance by using a “spatial pyramid pool” that can work on input of different sizes and scales. The convolutional features were only computed once and then pooling was performed on sub-areas. Girshick [53] improved on both SPPnet and R-CNN by creating a network that takes region of interest proposals as a direct input, reducing the number of stages, and improving training for deeper networks. Ren et al.[136] further improved performance by creating a region proposal network (RPN) that was also combined with the Fast R-CNN design to create a single network that shares features. This allowed for the concept of attention, wherein the RPN instructs the network where to focus.

2.3.1 Caption Generation

Much of the early work on caption generation was in the area of headline generation for news imagery such as Feng and Lapata [46][47]. This builds upon the work of automatic summarisation where an existing sentence is extracted from the article to describe the image. They compared this with a method for generating new captions using a trigram-based language model. In both cases, a SIFT-based system was used to create a probability of words being associated with the image.

Many early approaches to image caption generation used a nearest neighbour approach where the caption from the closest image match was applied to the image. This works well with larger datasets as it increases the likelihood a matching caption will already exist in the training set. Farhadi et al. [43] defined a semantic space using a triplet of object, action and scene as an intermediate representation of images and sentences. These were then be used to match images to captions. Ordonez et al. [123] created a dataset of 1 million images each with a single caption harvested from Flickr. They demonstrated two methods for finding the most relevant caption. One used only global features for matches while the other attempted to also match the content to improve the score. Jia et al. [74] attempted to tackle the problem where the text is less directly related to the content of the image by using picture of the day content from Wikipedia. They defined a Markov random field to model similarities between the text describing the images. Hodosh et al.[65] used 8,000 images with five captions each. Their method

posed the problem as a ranking problem, and demonstrated the importance of multiple captions along with features that capture syntactic and semantic information. Socher et al. [150] used an RNN based on dependency trees to map sentences and images to a shared space for matching images and captions. Karpathy et al. [81] created a bi-directional mapping between sentences and images using image fragments from a R-CNN and sentence fragments. Devlin et al. [36] compared different approaches to nearest neighbour captioning. They found that although the approaches can perform better than methods that generate novel sentences in automatic scores, humans prefer generated sentences.

In some situations, there may be no matching caption in the training set. To avoid the difficult task of generating entirely new sentences, it is possible to combine fragments of existing sentences to create novel sentences. Li et al. [97] estimated object information triplets, visual attributes and spatial relationships from images, and used n-gram frequency data from Google to reconstruct sentences. They used a sentence fragment for each object and the spatial relationship between objects. Kuznetsova et al. [89] tried to retrieve sentence fragments of different types from the training set and then optimise with a number of constraints to create the best sentence they can. This work was built upon by Kuznetsova et al. in [90] where the hierarchical nature of natural language was used. They combined tree fragments to build complete sentences.

Images in a dataset will often fall into similar categories, where sentences following a similar structure can be used to describe them. It can be possible to use a template to create captions as this simplifies some of the difficulties of grammar and sentence structure. Yao et al. [181] parsed an image to create a hierarchical representation from a scene level down to primitives. This was combined with WordNet data to create a semantic representation of the input. Sentences were created using a simplified head-driven phrase structure grammar (HPSG). This defined rules for the formulation of sentences. Kulkarni et al. [88] discovered objects in images along with attributes and prepositions. A Conditional Random Field (CRF) was created to encode the labelling of the image that can be decoded to create sentences. They note that using only language models leads to difficulties with grammar and coherency, so they used templates to enforce constraints. Gupta and Mannem [57] tested the effects of the

quality of annotation on generated sentences without using computer vision. They used a few different templates that depend on the predicted verb. They found that predicting the verb from the annotation is the most difficult part. Yang et al. [180] used a HMM for predicting a quadruplet of noun, verb, scene and preposition before using a template and heuristics to generate an output sentence. Elliott and Keller [39] created a representation of the image with labelled regions and how the regions are related. Templates were then filled using information encoded in these representations.

The most challenging and potentially most powerful captioning techniques attempt to perform natural language generation. Mitchell et al. [114] tied the language generation and computer vision elements together more closely to filter noisy results from computer vision. Its model was data driven but with some hand written rules. Yatskar et al. [182] used densely annotated images to analyse what information is required by language generation systems. They used a generative model with a vocabulary of 2700 words that was conditioned on the densely annotated input.

The biggest recent change in image captioning has been the use of RNNs. An early use of RNNs for language processing to create a better language model than with n-gram models was by Bengio et al. [11]. Socher et al. [151] used deep RNNs for parsing images and sentences. Many methods using RNNs were developed simultaneously. Karpathy et al. [80] used a bi-directional RNN with input from VGGnet [146] and word embedding using word2vec [92]. They noted that using random initialisation for the word vectors is just as effective. Chen et al. [22] also used a bi-directional RNN design and a variant of AlexNet. Kiros et al. [84] used a Long Short-Term Memory (LSTM) [64] variant of an RNN with input from VGGNet. Vinyals et al. [167] used a similar design but used a more complicated LSTM instead, and took input from a variant of GoogLeNet. Donahue et al. [37] used an LSTM that has 2 layers, which they described as “doubly deep”. They took features from a CNN design based on AlexNet. They also allowed for video captioning using a CRF with input from multiple frames. Mao et al. [106] did not use an LSTM design but had a network with 6 layers at each time step including 2 embedding layers, a recurrent layer, a multimodal layer that took input from AlexNet, and finally, a softmax layer. Venugopalan et al. [166] extended RNN-based captioning to video sequences by using multiple input images to the LSTMs, and also by using

optical flow images to capture motion.

In contrast to the many recent RNN-based papers, Fang et al. [42] took a different approach. They trained a CNN for Multiple Instance Learning (MIL) on a vocabulary of 1000 words. They used a language model based on Maximum Entropy (ME) to rearrange words into properly structured sentences. Devlin et al. [35] combined the ME language model and RNN approaches. They found that improvements in Bilingual Evaluation Understudy (BLEU) scores are not matched by an improvement in human preference.

Xu et al. [174] used an adapted LSTM design that is capable of focusing its attention on different regions of the input image. Instead of utilising the final hidden layer of CNNs, a convolutional layer was taken as input. Different models of attention that can be used are described. Johnson et al. [77] created an approach that combined an LSTM and the R-CNN approaches of Girshick [53] and Ren et al. [136]. This created an end-to-end trainable network with attention for dense captioning. You et al. [185] combined top down and bottom up approaches to caption generation. An RNN design with input from a CNN as well as word level attributes was used. Rather than visual attention, the model is described as having semantic attention over attributes. Rennie et al. [137] combined attention and image embedding with reinforcement learning. The form of reinforcement learning used allowed the use of its own output to normalise the reward. Mun et al. [116] further expanded on attention by using a retrieved caption to guide the visual attention used to generate a detailed caption. This represents the state-of-the-art on the Microsoft Common Objects in Context (MSCOCO) dataset.

Several different datasets have been proposed for image captioning. One of the earliest was the Pascal [43] containing 1,000 images with five captions each. Flickr8k [65] contained 8,000 images, each with five captions, and was expanded upon by Flickr30k [186] to 31,783 images. The current standard is the MSCOCO [100] dataset. MSCOCO contains 80,000 training images and 40,000 validation images. All images have 5 captions and segmentation for 80 object classes. In addition, there are 40,000 test images where the captions are withheld. Ferraro et al. [48] compared the different available datasets on a variety of different metrics.

Automatic evaluation of caption generation is a difficult task in its own right. BLEU [127] was originally designed for evaluating machine translation and measures the precision of the text. Recall-Oriented Understudy for Gisting Evaluation (ROUGE) [98] is better for measuring recall using the longest common subsequence. Metric for Evaluation of Translation with Explicit ORdering (METEOR) [7] attempts to balance the measure of both precision and recall to better match human judgement. Consensus-based Image Description Evaluation (CIDEr) [165] is designed for image caption assessment and tries to capture the consensus of the reference captions.

2.3.2 Emotion

Human emotion is a topic that has been widely researched in the areas of psychology and biology. Sentiment and emotion in images, captions and annotation is a much younger field. Picard [131] first introduced the concept of “affective computing” and noted the importance of emotion for artificial intelligence. A lot of the work in the field has focused on recognition emotion of humans from facial expression and tone of voice.

Much work has been put into defining the basic set of human emotions. Ortony and Turner [124] compared the different sets of emotions that have been proposed. The basis for including emotions varies between authors. Tomkins [164] used the density of neural firing. Plutchik [132] used Darwin’s theory of evolution. There is a strong overlap between the different sets of basic emotions. Some sets used different synonyms for the same emotion. For example, Panksepp [126] used “expectancy” while Plutchik used “anticipation”. Plutchik represented the basic emotions using his wheel of emotions. This defined 8 basic emotions with 3 different levels.

Emotion is present in plain text as well as in spoken language. Detecting emotion in text is often done through sentiment analysis, which measures the positive or negative feelings conveyed by the text. Socher et al. [152] used a recursive neural network for parsing sentences to measure the sentiment. A single sentence can contain multiple positive and negative words and combine them in ways that lead to different overall sentiments. Double negatives for example, can indicate a positive sentiment. The corpus is based on the Pang and Lee [125] dataset taken from film reviews. Hutto

and Gilbert [72] applied sentiment prediction to social media content. They combine heuristics and lexical features to classify the sentiment.

It is also possible to classify the emotions in text. Tao [160] looked at predicting joy, sadness, anger, surprise, hate and fear in text. They use not only the emotional words but also the context in which the words were used. Shivhare and Khethawat [145] also detected emotion in text by using a combination of natural language processing and machine learning. Munezero et al. [117] examined the differences between affect, feeling, emotion, sentiment and opinion. They also gave suggestions on how understanding the differences can be used to improve their detection in natural language. They concluded that emotions are highly complex and nearly impossible to detect completely from text alone. The use of language to convey emotion depends on multiple factors including cultural and physiological reactions Roberts et al. [138] proposed a dataset of tweets with emotions and compared the distribution of emotions with other corpuses. Bandhakavi et al. [6] looked at learning the association of emotions and words to create a lexicon for emotion detection in social media posts.

Detection of emotion in text is one side of the relationship between text and emotion. Generating text that expresses emotion is another task that has been less well explored. Keshtkar and Inkpen [82] used heuristics and rules based on patterns of part of speech (*Is*). Ghosh et al. [52] proposed a neural model for generating conversational text with customisable affect and intensity. They allowed the affect to be set to positive, negative, anger, sadness and anxiety. They used a few seed words as context to generate a finished sentence from that input. Zhou et al. [191] also investigated emotional language in conversations and responses that relate to the emotional context of the input sentence. Asghar et al. [5] created a neural model with an affective word embedding space along with an affective loss function. Yanardag et al. [178] used an RNN to learn to write horror stories based on human written horror stories harvested from the web. This combines emotion with natural language by training on a dataset consisting only of horror stories. This is a data driven approach that relies on the data set containing text with a certain style.

Images are also very strongly related to emotion. Rahwan et al. [133] used CNNs trained

on “scary” imagery to transfer style to other images to maximise the “scariness” of the new images. This was an attempt to use a CNN to create images that can invoke an emotion in the viewer. Yanardag et al. [179] used style transfer from disaster images to images of familiar locations to induce empathy in the viewer. Here, the emotions are experienced by the viewers due to the context of the output image.

Text and images can be associated with each other to create a three way relationship that combines images, text and emotion. The Visual Sentiment Ontology (VSO) [14] proposes a dataset of images from flickr that were tagged with pairs of adjectives and nouns (Adjective Noun Pairs (ANPs)). The adjective provides a description that relates to the noun. These ANPs relate to emotional content in the images. The dataset uses Plutchik’s definition of the basic emotions. They also proposed SentiBank, detectors for sentiment in the images. With the advent of CNNs, DeepSentiBank [21] improved upon SentiBank using deep learning. Socher et al. [152] used an RNN for parsing text to recognise positive or negative sentiment using deep learning.

The combination of generating text that expresses emotion related to images is an even smaller area of research. Huber et al. [70] combined sentiment, images and facial features to create a conversational agent. Image caption generation was combined with disturbing imagery and text to create “Norman” [177], which generates disturbing captions for images. They compared captions generated by their system to captions trained on MSCOCO. Mathews et al [111] combined ANPs with MSCOCO to generate captions expressing positive or negative sentiment for images. Karayil et al. [79] detect ANPs in images and generate captions from the detected ANPs. You et al. [184] propose a method for injecting sentiment into generated captions. None of these combine caption generation for images with the ability to select the emotion of the output caption.

Chapter 3

Character Identification in Broadcast Video

A character is defined as “a being involved in a story”. In a live action broadcast video, most characters will be portrayed by a person. The identities of the characters present in a scene are a useful information source for automatically generating a description of the content. Manually labelling characters is incredibly time consuming, and automating the process is a challenging problem. The script only provides a weak annotation of which character is speaking. Information such as their position is not available. Character appearance is highly variable due to lighting, pose and expression. There is no guarantee that a speaking character is visible in the footage, although it is true in the vast majority of cases.

In order to uniquely identify characters, unique traits need to be found that can separate them from the other characters within the video footage. One of the ways to do this is by looking at the faces of the characters, as in the work of Everingham et al. [40] [41]. Another feature used by Everingham et al. is the appearance of the clothing worn by the characters. Zhang et al. [189] proposed a technique that combines multiple cues including a CNN trained on the full body in addition to face recognition. Other traits used for person identification include gait [30], and the ear [71].

The subtitle and script alignment are performed as in section A.2. The script indi-



Figure 3.1: Example faces from the opening sequence of “Friends”

cates which characters are present within each scene. By automatically detecting faces throughout the video and accurately regressing facial features, an unsupervised clustering process is used to identify individuals without user intervention. The script information is used as weak supervision in this clustering process to identify the likelihood of characters over subsets of frames.

Typically characters do not appear in isolation, instead groups appear together. Different scenes will contain different characters, and it is therefore possible to use the difference between the groups to find unique characters. Distributions of face descriptors can be used to represent the groups and be subtracted to isolate characters. A GMM is used to model the distribution of the facial features of the characters across scenes. The model of the feature space is then used to isolate and name the characters.

The script provides some useful annotation, but there are other sources of information. The internet contains a large quantity of labelled images of actors that can be associated with the characters they portray. This information is used to train classifiers to identify the characters in video footage from unlabelled episodes. However, this has its own



Figure 3.2: The regressed landmark positions on a face

challenges due to large variation in actor appearance between images of them in both real life and other roles.

The popular American sitcom “Friends” contains an ensemble cast and has scripts available for the episodes. This makes it suitable for use as a dataset. The episodes are available in high definition formats, allowing for more facial detail to be present. An example of faces detected in the opening sequence is shown in figure 3.1. The facial poses are shown in isolation above, and the method is shown to work across a variety of different faces.

3.1 Facial Landmark Regression and Feature Extraction

The first step towards identifying characters is to detect the faces in the video stream and generate descriptors. This needs to be done before the faces can be associated with a character from the script. A cascade of linear predictors is trained to regress facial feature positions from the SIFT features extracted at each of the 68 facial landmarks used. This is based on the work of Xiong and De La Torre [176], and Ong and Bowden [121][122]. The 68 points used are the MULTI-PIE [56] mark up shown in figure 3.3. This is the annotation used in the *300 Faces in-the-wild* challenge dataset

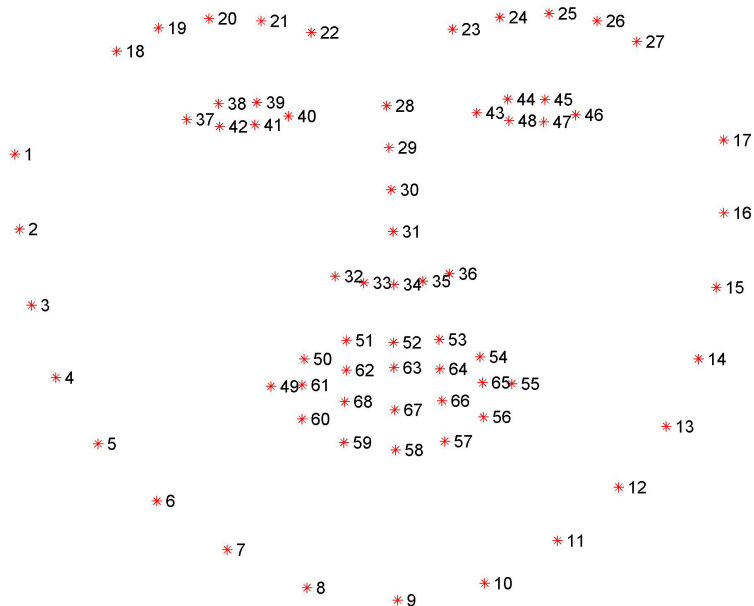


Figure 3.3: The 68 facial landmark positions

(300-W) [140] that was used for training.

3.1.1 Regression

Given a set of face images $\mathbf{d}_i \in \mathbf{D}$ with associated facial landmarks from the set of all facial landmarks (\mathbf{X}), $\mathbf{x}_i \in \mathbf{X}$. The objective is to find a mapping \mathbf{H} that can predict the displacement of the landmarks $\delta\mathbf{x}$ such that $\delta\mathbf{x} = \mathbf{H}\phi(\mathbf{x})$. The image observation, $\phi(\mathbf{x}) \in \mathbb{R}^{nm+1}$, is a 6,733-dimensional concatenated SIFT vector, where $n = 99$ is the dimension of a SIFT descriptor after projection through Principal Component Analysis (PCA) to keep the majority of the variance in the feature space. $m = |\mathbf{x}|/2$ is the number of facial landmarks and a single dimension is added as the bias term in linear regression.

To learn the mapping, \mathbf{H} , a training set of random initialisations $\mathbf{T} \in \mathbb{R}^{(|\mathbf{x}| \times r | \mathbf{X}|)}$ is extracted by randomly offsetting the model from the true face location and recording the displacements. For each displacement, the image observations are compiled into $\Phi \in \mathbb{R}^{(|\phi| \times r | \mathbf{X}|)}$ where $r = 10$ is the number of random offsets per image.

\mathbf{H} is then calculated as the least squares solution $\mathbf{H} = \mathbf{T}(\Phi\Phi^T)^{-1}$. As a single linear

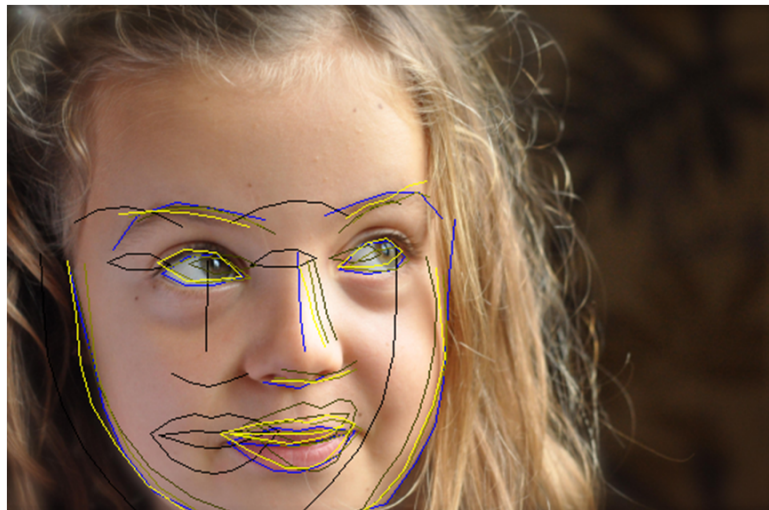


Figure 3.4: The sequence of landmark regression

mapping is insufficient to model the complexities of a highly deformable object like a face, a sequence of regressors is used where $\delta \mathbf{x}^i = H^i \phi(\mathbf{x}_0 + \delta \mathbf{x}^{i-1})$.

The facial regressor is trained on 5,691 example images taken from the 300 Faces in-the-wild challenge dataset (300-W). These images are annotated using the Multi-PIE 68 point mark up shown in Figure 3.3. The key points focus around the eyes, nose, and mouth but also include the edge of the face. The full dataset consists of consistent re-annotations for LFPW [10], AFW [193], HELEN [93] and XM2VTS [112]. The additional 135 IBUG images and the testing subsets of the aforementioned datasets are retained for internal validation.

Figure 3.2 shows the result of applying the learnt regressor to a sample static image. The regression works well on this image but with some slight error in the eye positions and some separation from the edge of the face on the left side. Figure 3.4 shows the process of regression for each regressor in the cascade starting from a mean face (dark) to final regression (yellow). The dark face is very far from the true position while the blue face is closer to the true location. The final yellow regression is very close though with some slight error on the left side of the image and in the shape of the lips. An example of regressing multiple faces from a frame taken from “Friends” is shown in Figure 3.1.

3.1.2 Feature Extraction

At runtime, to extract faces and features from a video, a Viola Jones-boosted face detector is applied to each frame. Following non maximal suppression, the regressor is independently applied at each positive detection using the mean face $\mathbf{x}_0 = \frac{1}{|x|} \sum_{\forall \mathbf{x}_i \in \mathbf{X}} \mathbf{x}_i$ as the initial estimate. This is as it was performed during training. The initial face estimate is scaled and translated to the detected face region, and the SIFT features are set at one tenth of the face scale. This translates to roughly half the size of an eye. The regressor typically converges onto the face using 4 to 5 linear predictors in the sequential cascade. The 6,733-dimensional vector made from concatenating the SIFT descriptors is then used to create a descriptor for each detected face.

3.1.3 OpenFace feature extraction

An alternative method for extracting features is using the OpenFace [3] library, which is built upon the designs of CNN-based facial features such as FaceNet [143]. The output of the network is a 128-dimensional representation of the face as a position on a unit hypersphere. The intention of this is to create a space where a larger distance between faces means the faces are less likely to be from the same person. This makes the descriptor useful for techniques such as clustering, classification, and similarity detection. The model used is the “nn4.small2.v1” model, which obtains an accuracy of 0.9292 on the LFW [69] challenge. Other CNN-based face descriptors exist such as DeepFace [159], which achieves 97.8% on LFW. However, the model does not create a representation that is as meaningful in Euclidean space. This is also the case with the VGG face network [128], which achieves 97.3% accuracy on LFW. All of these models achieve a very high accuracy, with little difference between them in terms of performance. The OpenFace descriptor was used with the methods described in this chapter to provide a comparison with the performance of the SIFT-based descriptor.

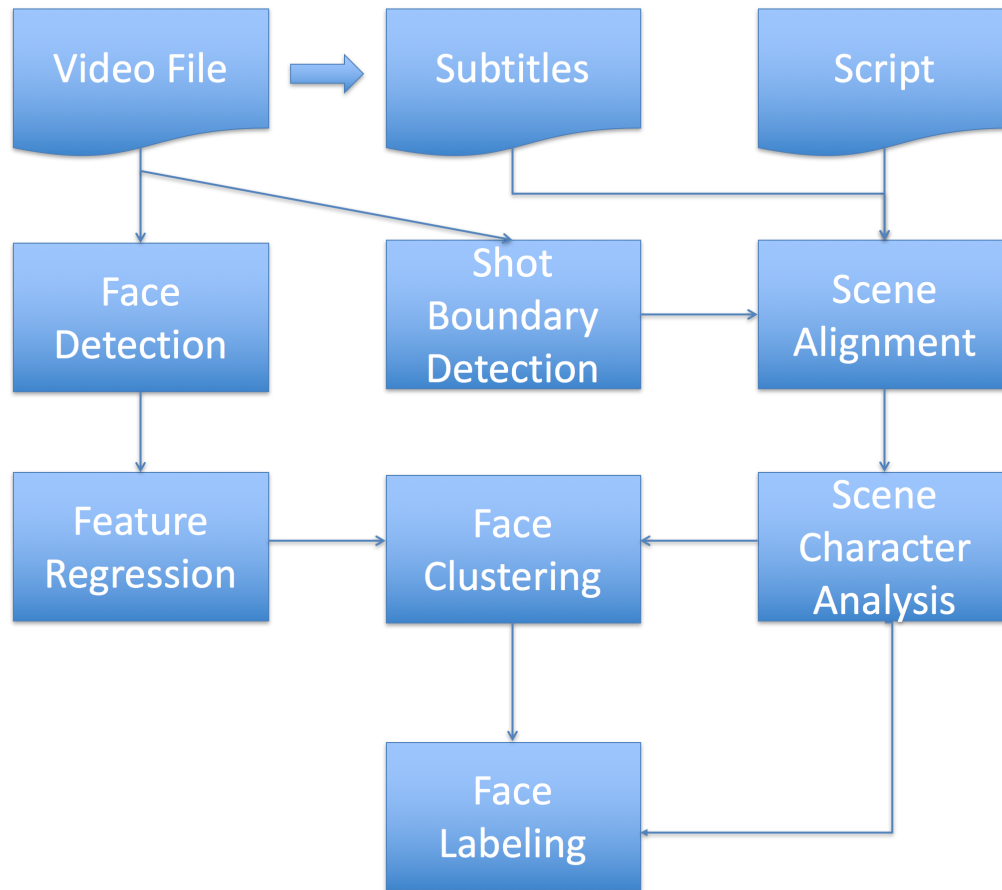


Figure 3.5: Character identification framework.

3.2 Unsupervised Facial Feature Clustering

Assigning character names to the faces requires several steps; an overview of the approach is shown in figure 3.5. Firstly, the subtitles are used to align the video with the script as in section A.2. Using the time aligned script, every scene that contains a particular character is identified. This is passed to the clustering process along with the face descriptors extracted from these frames. Unsupervised clustering provides sets of visually similar face descriptors, which are assigned labels from the script.

3.2.1 Script Processing

To name the faces that have been extracted, the face descriptors are clustered and labelled with character names from the script. Clustering all of the data from a whole episode at once results in less distinctive cluster sizes. This is because each character is more likely to appear an equal amount across the whole episode, while scenes are more likely to focus on a subset of characters. The scenes are divided into subsets that include every scene that contains a particular character. The aim is to make that character the most frequently appearing face in that subset and therefore, the largest cluster. The size of the remaining clusters should correspond to the frequency of each other character appearing in those scenes. It is also unlikely each character will appear with every other character. This means there will be fewer characters in the clustering process, lowering the chance of confusion between characters. For each scene k , the character list Π_k is defined as containing the characters z present in U^s_k . These are characters that have dialogue or stage instructions in the script for a particular scene. The assumption is that each character present will have some dialogue or stage instructions. The set of all the characters in the episode, Z , is defined by $Z = \bigcup_k \Pi_k$. With the script aligned to the video, the scene boundaries are used to extract the frames and associated face descriptors from each scene. The result is a matrix (\mathbf{E}) of frames (i) from the video where characters occur together,

$$\mathbf{E}_{xy} = \{i | i \in U^s_z \text{ and } z_x \in \Pi_z \text{ and } z_y \in \Pi_z\}. \quad (3.1)$$

\mathbf{E}_{xy} contains the set of all frames from all scenes including character z_x and character z_y . This can then be normalised by the number of frames in which each character is present,

$$\bar{\mathbf{E}}_{xy} = \frac{|\mathbf{E}_{xy}|}{\sum_l |\mathbf{E}_{xl}|}. \quad (3.2)$$

For each character, (z_k), k-means clustering is performed on the descriptors for all face candidates Υ_k . The face candidates are the set of feature descriptors (v) extracted from all the frames in the scenes containing that character. By only selecting face candidates from all the scenes containing a particular character (z_k), the input to the clustering is

reduced. The face candidates are obtained by,

$$\Upsilon_k = \{v^F | v^F \in U_j^s \text{ and } z_k \in \Pi_j\}. \quad (3.3)$$

The number of clusters n is the cardinality of the set of characters that co-occur with that character,

$$n_k = |\{\mathbf{E}_{jk} \neq 0\}|. \quad (3.4)$$

3.2.2 Clustering and Labelling

The expectation when clustering the face descriptors is that each individual will be partitioned into separate clusters in the feature space. The cluster centres μ are found using the k-means algorithm,

$$\mu_k = \text{kmeans}(\Upsilon_k; n_k). \quad (3.5)$$

The k-means is initialised with k randomly selected cluster centres. Each of the points is assigned to its nearest cluster, and then the cluster centre is shifted to the mean of that cluster. This is performed iteratively until there is very little change or the maximum number of iterations is reached. This partitions the data into clusters of face descriptors but they have yet to be assigned character labels. To overcome this problem, a histogram of the number of face descriptors belonging to each cluster, with index l , is created. The histogram M_{jk} is calculated by,

$$M_{jk} = \frac{\left| \left\{ v | v \in \Upsilon_j \text{ where } \arg \min_l (|v - \mu_l|^2) = k \right\} \right|}{|\Upsilon_j|}. \quad (3.6)$$

To match the labels from the co-occurrence histogram to the membership histogram, it is re-ordered to find the minimum χ^2 distance between the two histograms. The size of each bin in the membership histogram should correspond with the size of the bins in the co-occurrence histogram. This relies on characters appearing separately from other characters in different scenes. This allows the labels from the co-occurrence histogram to be transferred to the membership histogram. Each cluster now has a label corresponding to a character name.

3.2.3 Results

The clustering technique was evaluated on a dataset of approximately 35,000 images, obtained from the TV sitcom “Friends”. This program is ideal for examining the approach, as the large cast of “main” (i.e. re-occurring and named) characters, makes the task especially challenging. In addition, scripts are easily obtainable for the series, and the large amount of dialogue facilitates accurate subtitle-to-script alignment. The total number of named characters present in the relevant script is 9 spread across 14 different scenes.

From the dataset, around 20,000 face descriptors are extracted. After partitioning based on character co-occurrence as described in section 3.2.2, each character has an average of 6,830 face candidates. The experiment was performed with both the SIFT and OpenFace descriptors to see how each of them performs with the method.

In order to assign identities, it is assumed that the density of the clusters relates to the frequency of character co-occurrences. Background faces may also be detected and mixed into the set of face candidates. These will generally occur much less frequently and will hopefully be subsumed by another, larger cluster. In order to evaluate this assumption, figure 3.6 compares histograms from the three most commonly occurring characters in the dataset. The cluster size histograms on the bottom rows have been re-ordered so that their χ^2 distance is as low as possible to the co-occurrence histogram above. The size of each cluster should correspond to how frequently that character appears. This is not necessarily the same as re-ordering, despite the examples in figure 3.6.

It can be seen that, in general, cluster density correlates well with character co-occurrence frequency, meaning that the assumption used for identity assignment is likely a valid one. There are some inconsistencies, particularly with minor characters, as they tend to talk (and be visible) less often. Thus, the cluster density of the smallest clusters is lower than what would be expected if all characters appeared equally. The cluster sizes for the SIFT clusters and OpenFace clusters are similar. However, the smaller clusters have increased in size with the OpenFace descriptors. The larger clusters are smaller to compensate. This shows that the less frequent characters are

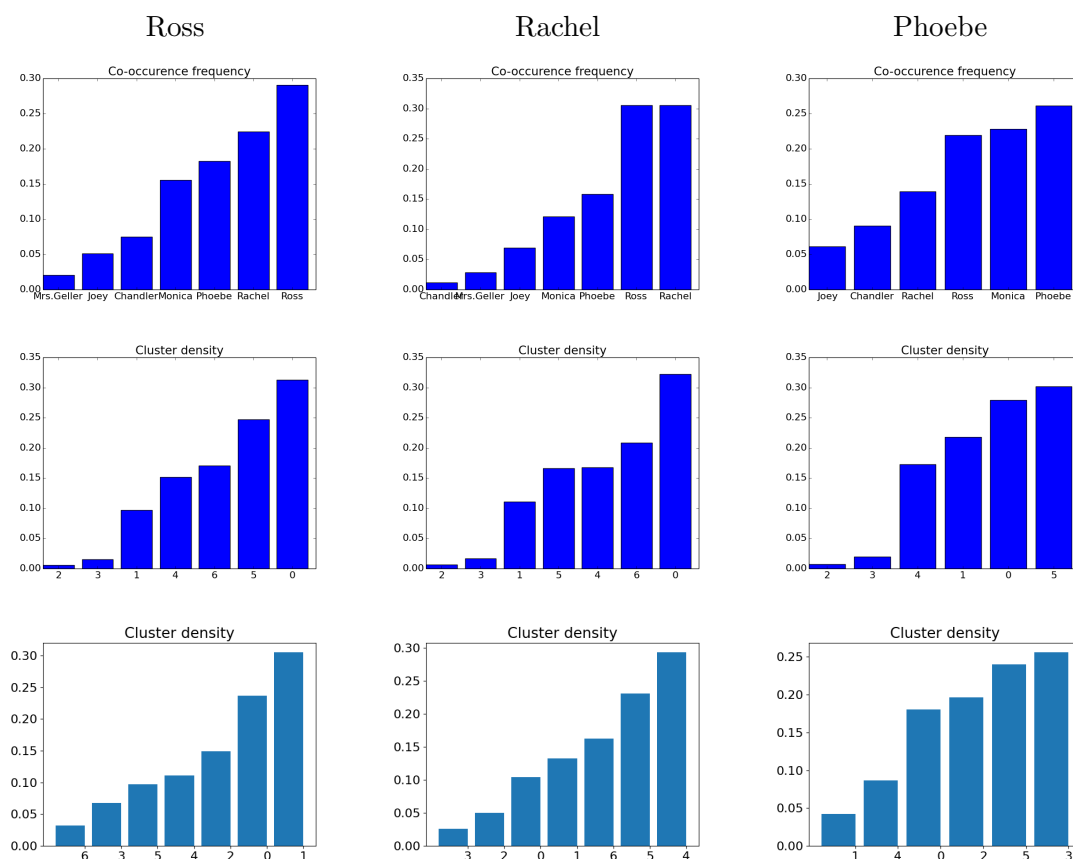


Figure 3.6: Comparison of co-occurrence frequency (top) and cluster size for SIFT (middle) and OpenFace descriptors (bottom) after identity assignment, for the 3 most common characters.

being formed into better clusters rather than being confused other characters or clustering based on other information such as facial pose. The OpenFace representation is designed to better separate identities with a Euclidean distance and should cluster based on identity.

It is extremely time-consuming to manually annotate the identity of tens of thousands of face images. Avoiding this task is one of the primary motivations for the automatic data-driven approach. As such, the technique is evaluated in terms of “character exemplars”. For each cluster with an assigned identity, the face candidate closest to the cluster centre is extracted, and whether this exemplar belongs to the assigned character is recorded. The exemplars when using the SIFT-based descriptor are shown in fig-


















































Exemplar of character	Rachel									
	Ross									
	Chandler									
	Janice									
	Joey									
	Monica									
	Mr. Geller									
	Mrs. Geller									
	Phoebe									
		when co-occurring with character								
		Chandler	Janice	Joey	Monica	Mr. Geller	Mrs. Geller	Phoebe	Rachel	Ross

Figure 3.7: The character co-occurrence matrix \mathbf{E} , represented using the assigned exemplars. Exemplars with a blue border are correctly identified. Gaps indicate pairs of characters which never co-occur within the dataset. Roughly one third of cluster exemplars have had the correct identity assigned.

ure 3.7 for the full co-occurrence matrix \mathbf{E} , as calculated by equation 3.1. Blue borders indicate exemplars with correctly assigned identities.

Overall 29% of the exemplars are assigned the correct identity when using the SIFT-based descriptor and 31% when using the OpenFace descriptor. This is around 2.5 times the accuracy achievable by random assignment, indicating that even the weak supervision can be hugely beneficial. However, overall this is still a poor overall performance that needs to be improved before it would be suitable for use in a caption generation system for broadcast video.

This evaluation can be extended to look at the accuracy of the K-nearest exemplars,

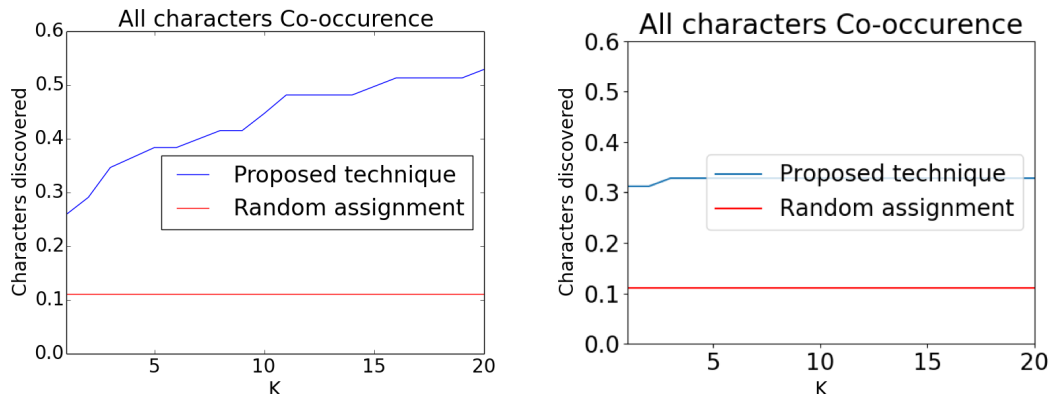


Figure 3.8: Accuracy of identity assignment for the K-nearest exemplars using the SIFT-based descriptor (left) and OpenFace descriptor (right).

as shown in figure 3.8. The SIFT-based descriptor results are on the left and the OpenFace descriptor results are on the right. It shows that when examining up to the top 20 exemplars, it is possible to identify more than 50% of the characters correctly. The OpenFace results start with the first exemplar being correct more often than when using the SIFT-based descriptor. However, this does not improve when examining more exemplars. This is because the clusters contain fewer faces from other characters. This makes it less likely any faces belonging to the character name will appear if the label is different.

This technique provides a simple method for identifying characters in footage automatically using weak supervision from the script but has limited performance. The clustering only provides a simple, spherical model of the space as the membership to a cluster is defined by the closest cluster centre (Euclidean distance). The use of GMMs should allow for better modelling of the distribution.

3.3 Unsupervised Gaussian Mixture Models and Random Forests

To improve on the clustering used in the previous section, an improved character distribution model is required. A GMM is used to replace the clustering. A method to improve the separation of facial features between identities, rather than pose and



Figure 3.9: Facial features regressed and characters labelled on a frame from “Friends”

lighting is also required. This allows for the use of more advanced character name assignment.

To create a GMM, the dimensionality of the features must be reduced as it is greater than the number of samples in many cases. Linear Discriminant Analysis (LDA) is one technique that attempts to find features that can discriminate classes. However, this technique also requires full rank covariance matrices. Instead, a random forest is trained to discriminate between identities and used to create a novel method for encoding the facial feature vectors. The forest is trained on 20 identities not present in the video. This allows faces to be represented as a probability of belonging to each of the training classes. Using the random forest output for each face descriptor, a GMM is built for each unique combination of characters. The GMM provides better model of distribution compared with cluster membership.

Improvements can also be made to the label assignment methodology. Rather than using character appearance frequency, information is combined from multiple co-occurrence sets. It is highly unlikely that characters will appear in isolation and multiple individuals are intermixed within a set. The power set defines all possible combinations of characters that could appear together in an episode. In the data, there is only a small portion of this power set. The approach adopted in this thesis is to find rules that allow the differencing of sets of characters to isolate individuals e.g. $\{Ross, Rachel, Phoebe\} \setminus \{Ross, Rachel\} = Phoebe$. These rules are then used with

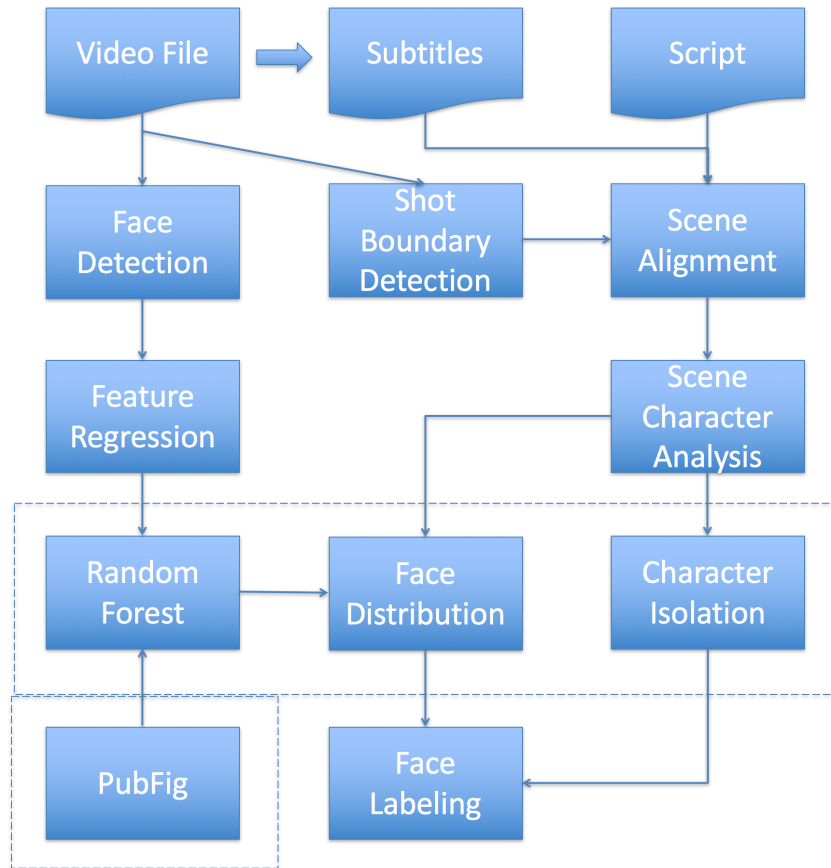


Figure 3.10: An overview of the random forest and Gaussian mixture model character labelling pipeline

the GMMs to find the probability distribution of each character in feature space. These rules combine multiple sets of characters until an individual is isolated. This allows for the classification of the extracted faces as one of the characters present in the script. An overview of this method is shown in figure 3.10. The new and changed sections are indicated by dotted lines. An example frame from “Friends” is shown in figure 3.9 with the regressed facial poses, and correct automatically generated character labels. Some of Rachel’s hair is covering her face and this could effect the accuracy of the character labelling.

3.3.1 Random Forests

The SIFT-based face regressor is applied to videos to create a face descriptor for each detected face in the video. This creates very high-dimensional feature vectors containing redundant data, which is not helpful for discriminating between characters. To find which data is useful to separate between characters, a random forest is trained. From the PubFig dataset, 10 males and 10 females with a large number of training examples were selected. This creates a balance between genders. The face regressor is applied to these faces, and the extracted features are used to train a random forest to distinguish between the classes. This uses the random forest to learn how to distinguish between identities to create a new feature space that can be used to better separate characters. The output of the forest will be a probability distribution across identities in the training set. A new face that does not belong to one of the identities in the training set should be represented as a combination of the identities, and occupy a new position in this feature space.

Eigen vectors from the faces regressed from the PubFig data are used to project the facial features extracted from “Friends”, which are then input to the random forest. The output probabilities at each leaf node in the random forest then generate a new feature vector of only 20 dimensions for each input. This process encodes the face descriptors based on their similarity to the data used to train the tree. The lower-dimensional feature vector is easier to process and requires fewer samples to create a Gaussian distribution.

3.3.2 Character Separation

The script and video alignment from section A.2 is used as before. The characters in a scene (Π_k) are taken as the characters present in the corresponding script scene (U_k^t). Given the set of characters Π_k in a scene k , the set of all characters in an episode, Z , is defined as,

$$Z = \{z | z \in \Pi_k, \forall k\}. \quad (3.7)$$

The power set of all characters, $\mathbb{P}(Z)$, with cardinality $|\mathbb{P}(Z)| = 2^{|Z|}$ contains all possible combinations of the characters within an episode. However, in practice only a



Figure 3.11: Example character combinations where the relative complement can be used to find an individual character

subset of all these character combinations will be seen i.e. $\bigcup_{\forall k} \Pi_k \subset \mathbb{P}(Z)$. In particular, isolated occurrences of a character $z \in Z$ are unlikely but are specifically of interest. A probabilistic model $p(z)$ can be learnt and used to label examples.

The aim is to find which subsets of the power set can be combined to isolate individual characters. Figure 3.11 shows 2 possible combinations of characters and how they can be differenced to isolate one of the characters.

All of the different character combinations (Π_k) occurring in the script are stored in a set (B), initially $B^0 = \Pi_{0...K}$ for k scenes. The set of available character combinations can be expanded by finding the difference between character combinations in the set where one is a subset of the other,

$$B^{i+1} = \{B_k^i \setminus B_n^i | B_n^i \subset B_k^i\}. \quad (3.8)$$

This is iterated until $\|B^{i+1}\| = \|B^i\|$ meaning that no new subsets can be found. The set (L) contains the indices of the character combinations, which are then used to find the character combinations to subtract.

$$L^{i+1} = \{k, n | (B_k^i \setminus B_n^i) \notin B^i \text{ and } B_n^i \subset B_k^i\} \quad (3.9)$$

This generates the rules that are needed to perform character isolation.

3.3.3 Gaussian Mixture Model

The encoded feature vectors are created for all of the faces extracted from the “Friends” data by inserting the original SIFT features to the random forests described in sec-

tion 3.3.1. The scene boundaries are combined with the new face descriptors to model the distribution of faces for all scenes. All the frames containing a particular character combination (Q_m) are found from a scene (U^s_k) where the characters, Π_k , are the same as character combination B_m ,

$$Q_m = \{i | i \in U^s_k \text{ and } \Pi_k = B_m\}. \quad (3.10)$$

Each face candidate (v^F) is extracted from a frame in the video that may also contain other face candidates. For each character combination, all the face candidates (Υ_m) are extracted from the corresponding frames,

$$\Upsilon_m = \{v_j^F | i_j \in Q_m\}. \quad (3.11)$$

Once the face candidates are separated, their distributions are modelled so that the previously generated rules can be used to isolate the characters. For all of the face candidates (Υ_m) the k-means algorithm is used to find the cluster centres (μ_m where n_m is the number of clusters,

$$\mu_m = \text{kmeans}(\Upsilon_m; n_m). \quad (3.12)$$

For each cluster the covariance matrix is calculated so the cluster can be modelled with a Gaussian distribution. Face candidates (v^F) are members (M_j) of their closest cluster centre,

$$M_{mj} = \{v | v \in \Upsilon_m \text{ where } \arg \min_l (|v - \mu_l|^2) = j\} \quad (3.13)$$

The covariance matrix (Σ_{mj}) is found by subtracting the mean μ_m from each face candidate (v^F) to create a vector and then multiplying that by its transpose,

$$\Sigma_{mj} = (M_{mj} - \mu_{mj})(M_{mj} - \mu_{mj})^T \quad (3.14)$$

A covariance matrix (Σ_{mj}) is created and the means from the clustering are used to create Gaussian distributions. A weighting (ϕ_{mj}) is required for each cluster to be able to create a gaussian mixture model. This is the number of members of each cluster divided by the total number of face candidates,

$$\phi_{mj} = \frac{\|M_{mj}\|}{\|M\|} \quad (3.15)$$

With the means (μ_m), covariance matrices (Σ_m) and weightings (ϕ_m), a GMM is created for each character combination (B_m),

$$p_m(v) = \sum_{j=0}^n \phi_{mj} \mathcal{N}(\mu_{mj}, \Sigma_{mj}) \quad (3.16)$$

This creates the GMMs that can be used to facilitate the character naming.

3.3.4 Face Classification

To use the GMMs for classification, they are combined using the separation rules that were created in section 3.3.2. Equation 3.9 gives the pairs of GMMs from equation 3.16 that are combined to get the character combination (B_m). The probability of a face candidate (v) belonging to each combination of characters is found using,

$$R_m(v) = \{p_a(v) - p_b(v) | (a, b) = L_m\}. \quad (3.17)$$

This provides the probability of a face belonging to the combinations of characters that is found by combining GMMs. The function $W(v)$ is used to classify faces as characters,

$$W(v) = \left\{ B_m \left| \arg \max_y (R_y(v)) = m \text{ where } |B_m| = 1 \right. \right\} \quad (3.18)$$

This finds the highest character probability for each face extracted from the video and allows them to be labelled with a character name.

3.3.5 Results

As before, the method is evaluated using data obtained from the American sitcom “Friends”.

For each character combination, the face candidates from all of their scenes is used to create a GMM. These GMMs exist in high-dimensional spaces but are visualised in 2D using PCA to project onto the top 2 eigenvectors. These 2D heat map images visualise the distribution of each GMM. Examples of these heat maps are shown in figure 3.12 and figure 3.13. Although multiple characters are represented, only a single obvious peak is visible in most. The shape of the distributions is irregular. Figures 3.13(c) and

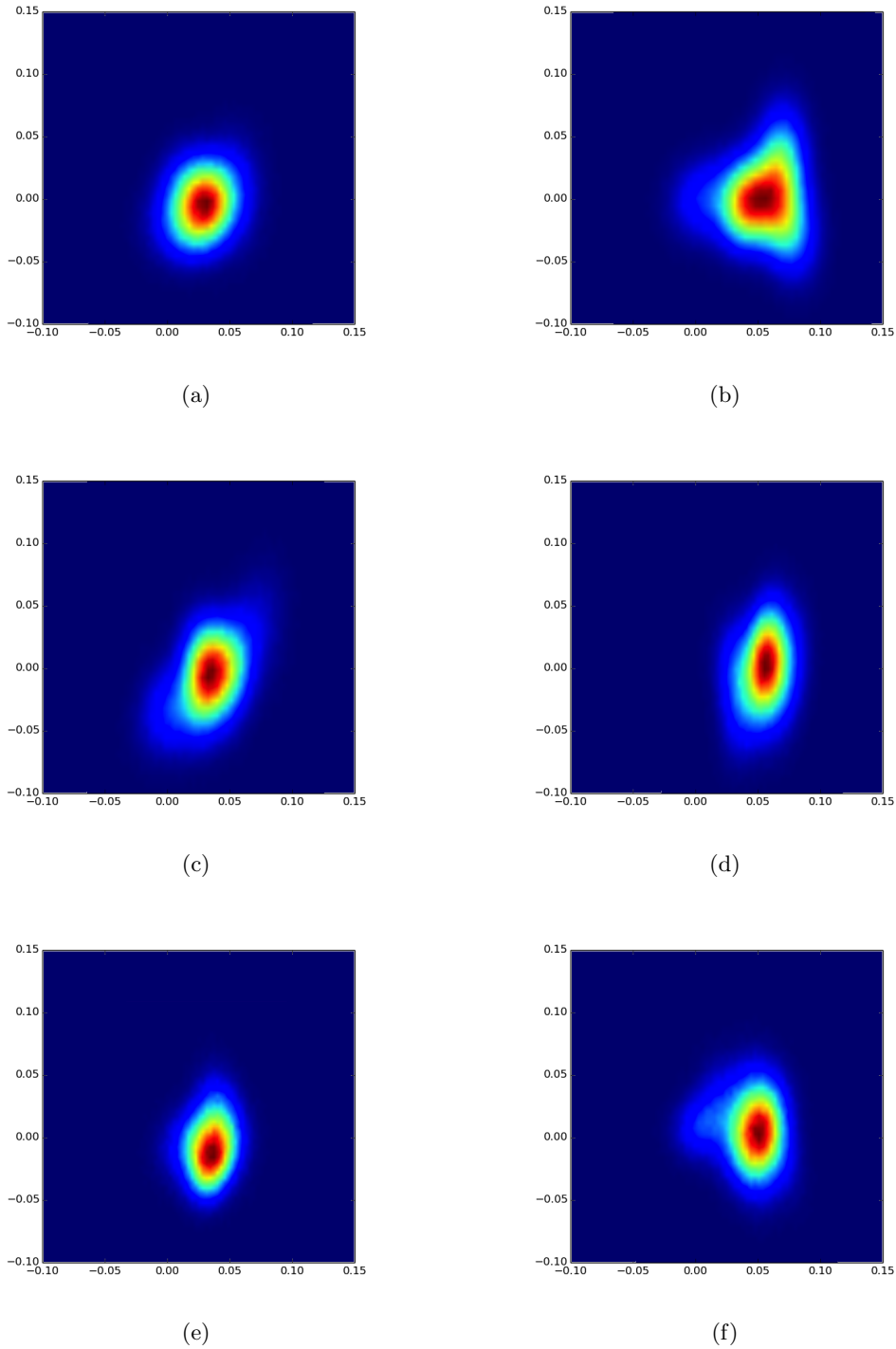


Figure 3.12: Example heatmaps for different character combinations. (a) Chandler and Joey (b) Rachel and Ross (c) Janice, Chandler and Joey (d) Monica, Ross, Chandler and Phoebe (e) Mrs. Geller, Rachel and Ross (f) Monica, Rachel, Ross, Phoebe and Joey

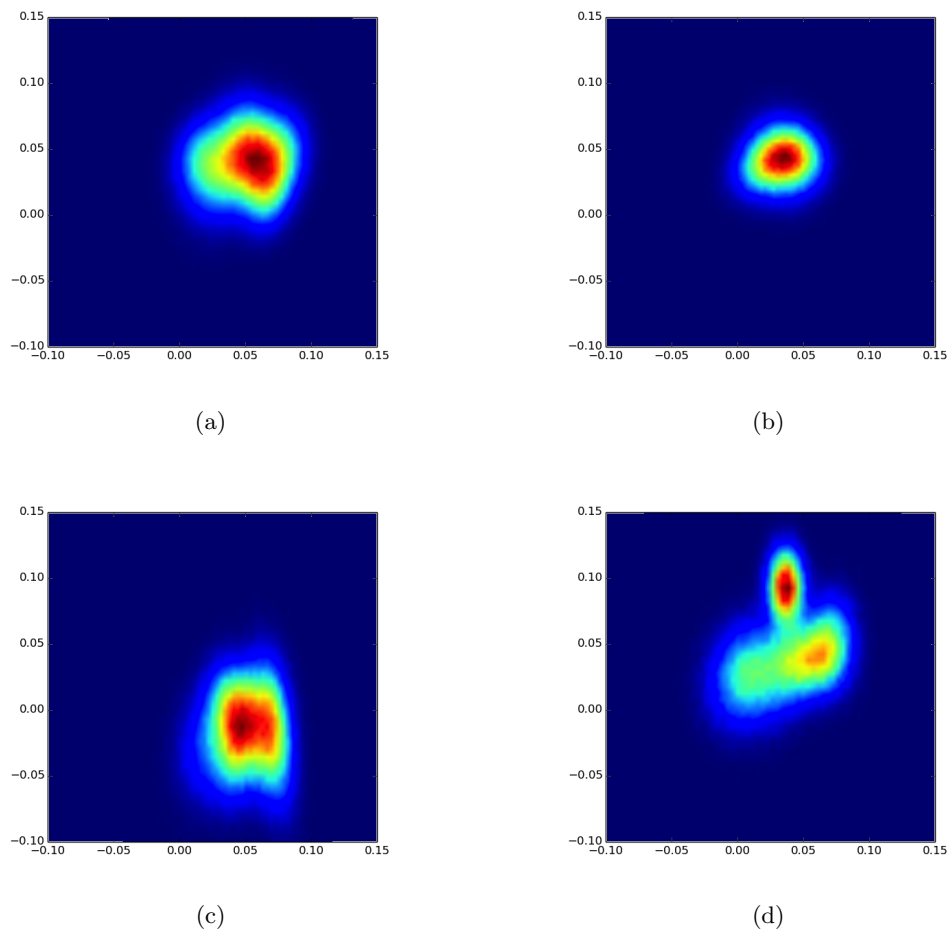


Figure 3.13: Example heatmaps for different character combinations. (a) Monica, Rachel, Phoebe and Ross (b) Monica and Phoebe (c) Mr. Geller and Monica (d) Janice and Joey

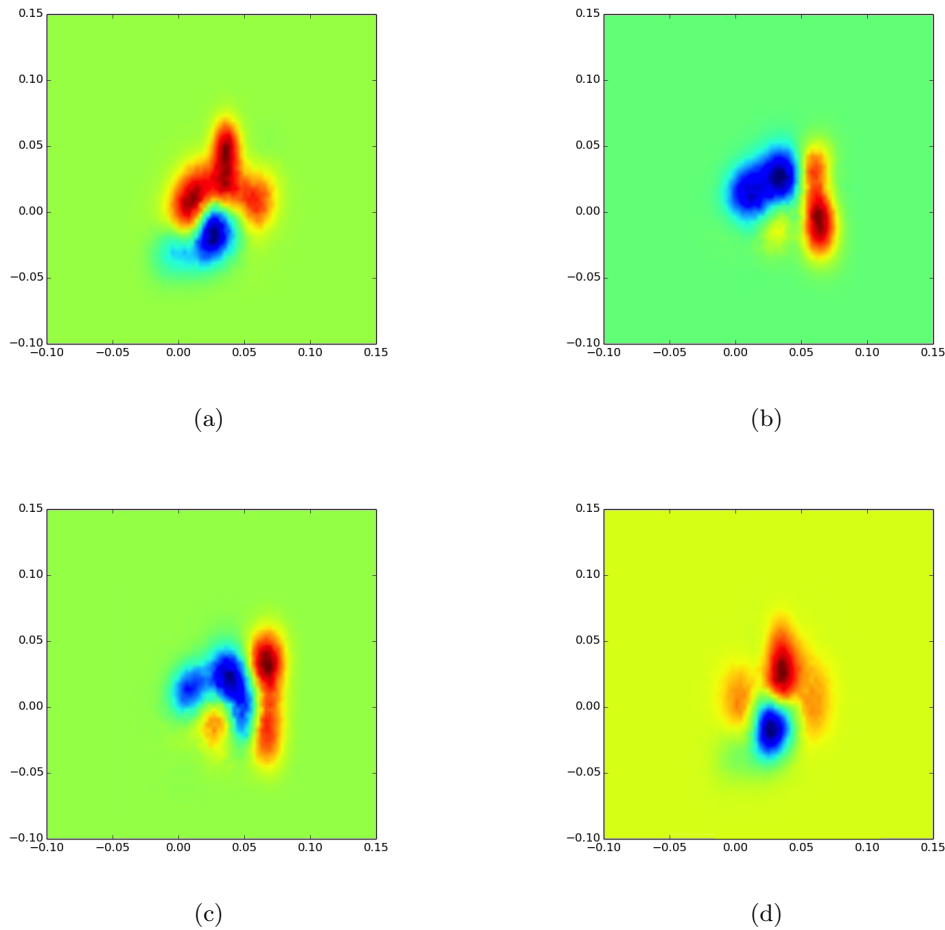


Figure 3.14: Example heatmaps for individual characters made by subtracting the heat maps for groups of characters. (a) Joey (b) Mr. Geller (c) Phoebe (d) Ross

(d) have more obvious separation in the space. These distributions show that some face distributions have clear and distinct modes whereas some are less well defined in feature space. The projection used for visualisation potentially masks subtle differences.

To visualise the distributions of individual characters, the 2D projections of the GMMs for groups of characters can be separated using the rules for character isolation described in section 3.3.2. Some example heat maps for isolated characters are shown in figure 3.14. The plots show the probability distribution of an individual character in the visualised space. The images show more obvious peaks and troughs in the space. There are now very low probability areas in the space where the other characters are located.

To evaluate the performance of the GMMs for classification, 8,000 faces from the 20,000 that have been automatically extracted were labelled. The methodology described in section 3.3.4 was used to find the most likely character for the test face candidates.

Table 3.1 shows the confusion matrix for the results of the classification on the episode of “Friends” using the random forest probability space. The average performance is around 63% compared to 29% using simple k-means clustering. Table 3.2 shows the performance for each character and the number of samples used. The number of samples is roughly proportional to how often the characters appear in the episode. The performance for characters with a low number of appearances in the episode is higher. This is likely to be because they have a smaller variation in their appearance, especially if they only appear in one scene such as Mr. and Mrs. Geller. The lowest performance on a main character is for Monica, who is often confused with another main female character, Phoebe. Monica appears with many other characters in the episode, this means the area her feature descriptors occupy could be less well defined by the GMMs. Ross is most often confused with Rachel, even though they are of the opposite gender. Ross and Rachel often appear together so the distributions are harder to separate, but it also shows the descriptor does not effectively separate characters based on gender. Janice and Joey are often confused with each other; Janice is more often classified as Joey than herself. These characters both appear together frequently and it shows the importance of hair appearance to the descriptor. Both characters have a similar fringe, as can be seen in figure 3.15. Interestingly, the confusion is in the random forest output space.

The GMM-based character identification was also performed using the OpenFace embedded representation, with the confusion matrix shown in table 3.3, and the table of results in table 3.4. These results show a higher performance than the random forest output descriptor, with an average performance of 85.22%, up from 63.49%. This is an increase of over 20%. Performance on Monica is the most improved, with a higher performance than Rachel. She is confused less often with other characters with her main confusion being with Phoebe, and no confusion with any male characters. This shows that the OpenFace descriptor is more suited to dividing people by their gender. The performance of Janice is also greatly improved, with her now being classified as

	Monica	Rachel	Mr. Geller	Phoebe	Mrs. Geller	Janice	Ross	Chandler	Joey
Monica	0.28	0.13	0.14	0.26	0.00	0.00	0.14	0.00	0.04
Rachel	0.20	0.58	0.00	0.16	0.05	0.00	0.01	0.00	0.00
Mr. Geller	0.01	0.00	0.99	0.00	0.00	0.00	0.00	0.00	0.00
Phoebe	0.14	0.03	0.00	0.78	0.00	0.00	0.02	0.00	0.03
Mrs. Geller	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00
Janice	0.00	0.00	0.00	0.00	0.00	0.38	0.00	0.04	0.58
Ross	0.03	0.32	0.00	0.06	0.06	0.00	0.53	0.00	0.00
Chandler	0.00	0.00	0.00	0.00	0.00	0.00	0.20	0.51	0.28
Joey	0.00	0.00	0.00	0.00	0.00	0.17	0.00	0.15	0.68

Table 3.1: Confusion matrix for the results of the automatic classification using a GMM-based character separation method with the random forest probability descriptor



Figure 3.15: Pictures of the “Friends” characters Joey (left) and Janice (right), showing their fringes which cause their characters to be confused.

herself the majority of the time. She is still confused with Joey but at the same rate as with Chandler. The differences between the characters are better represented by the embedding space, with less interference from their hair styles.

Using a GMMs provides an improvement over the use of k-means clustering and assigning labels based on cluster size and character appearance frequency as in section 3.2. The best performance using the cluster labelling is 31% for the OpenFace descriptor for the clusters. The character average performance using the GMM-based method is 63.49% with the random forest output-based descriptor, and 85.22% using the OpenFace descriptor. This is a large performance increase over the clustering based method. The 6,733-dimensional SIFT-based descriptor directly does not work directly with the GMM-based method as its dimensionality is too high. The OpenFace descriptor allows for an even higher performance with this method. This method only uses very weakly supervised data that is available within the episode itself. By bringing in external weakly supervised data, it is possible to further improve the classification accuracy.

Character	Correct Samples out of Total	Percentage
Monica	461/1661	27.75
Rachel	573/995	57.59
Mr. Geller	276/278	99.28
Phoebe	1031/1324	77.87
Mrs. Geller	171/171	100.00
Janice	81/215	37.67
Ross	950/1808	52.54
Chandler	655/1281	51.13
Joey	435/644	67.55
Character Average		63.49

Table 3.2: Table of results for the classification and the performance average for all characters when using the random forest embedded descriptor

3.4 Weakly Supervised Learning from Web Data

So far the methods have only directly used the existing data within the episode. The internet provides a large source of data that has already been weakly labelled. The web data can be automatically harvested to keep the system fully automatic. The script is still used to find information about which characters are in a scene and find the character names so that they can be related to the actor's names. Random forests are trained for each actor with positive data from the web and negative data from other characters in the video itself. The individual face detections in the video are combined into face tracks to provide temporal consistency. An overview of the method is shown in figure 3.16. This shows the added web harvesting and face tracking sections. The PubFig data is no longer used, as the random forest is now trained directly to identify the characters, meaning the character isolation rules are also no longer required.

	Monica	Rachel	Mr. Geller	Phoebe	Mrs. Geller	Janice	Ross	Chandler	Joey
Monica	0.76	0.07	0.00	0.17	0.00	0.00	0.00	0.00	0.00
Rachel	0.09	0.71	0.00	0.00	0.10	0.10	0.00	0.00	0.00
Mr. Geller	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00
Phoebe	0.05	0.02	0.00	0.93	0.00	0.00	0.00	0.00	0.00
Mrs. Geller	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00
Janice	0.00	0.00	0.00	0.00	0.00	0.85	0.00	0.07	0.08
Ross	0.01	0.09	0.00	0.03	0.03	0.00	0.84	0.00	0.00
Chandler	0.00	0.00	0.00	0.00	0.00	0.00	0.08	0.80	0.12
Joey	0.00	0.00	0.00	0.00	0.00	0.12	0.00	0.10	0.78

Table 3.3: Confusion matrix for the results of the automatic classification using a GMM-based character separation method with the OpenFace descriptor

Character	Correct Samples out of Total	Percentage
Monica	1262/1661	75.98
Rachel	706/995	70.95
Mr. Geller	278/278	100.00
Phoebe	1231/1324	92.98
Mrs. Geller	171/171	100.00
Janice	183/215	85.12
Ross	1518/1808	83.96
Chandler	1025/1281	80.02
Joey	502/644	77.95
Character Average		85.22

Table 3.4: Table of results for the classification and the performance average for all characters when using the OpenFace descriptor

3.4.1 Face Tracks

The frontal face detector and regressor from section 3.1.1 are applied to each frame in the video to find descriptors for each face. This creates descriptors for individual face detections but as it is a video, face detections in consecutive frames will be very similar and change slowly over time. The facial regressor performs best on frontal faces but the videos contain examples of non frontal faces. To provide temporally consistent labels to faces through the video and provide labels to faces that move to a profile position during the video, detections of faces are combined into tracks.

As well as the frontal face detector, a left and right profile face detector is also applied to the video. For each frame in the video, i_j , a set of face detections C are obtained at positions p by using the face detectors Ψ ,

$$C_j = \{p | \Psi(p) > o \text{ and } p \in i_j\}. \quad (3.19)$$

Each face detection position is combined with the nearest neighbour in the subsequent frame i_{j+1} within a threshold $\|p_j - p\|^2 < r$ unless the next frame is a shot boundary

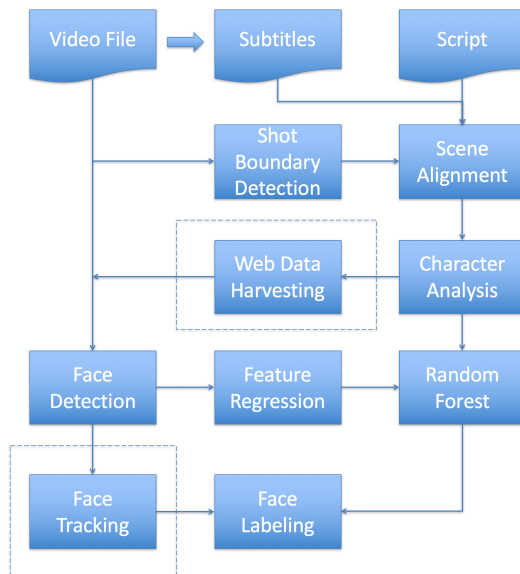


Figure 3.16: An overview of the automatic character naming methodology using automatically harvested web data

$(i_{j+1} \in SB)$,

$$p_{j+1}^k = \arg \min_{p \in C_{j+1}} (\|p_j^k - p\|^2). \quad (3.20)$$

The individual detections can now be combined to create tracks of faces within the video, Y^k is the set of all p^k . This allows for the combination of predictions within a track to get the best label for the track.

3.4.2 Data Collection

The video contains several main characters named in the script. To learn their identities, information that is easily obtainable from the internet is a useful resource. The main characters in an episode, Z , can be found using equation 3.7. Each of the characters (z) is portrayed by an actor that can be found by using the character names and episode number from internet resources. Figure 3.17 shows the actors names with the names of the characters. An image search engine is then used to search for the names of the actors and the top results are automatically downloaded. The face detector and regressor from section 3.1.1 are applied to each of these images to obtain face descriptors for the faces. Images with multiple faces are discarded as it is unclear which face

	Jennifer Aniston	...	Rachel Green
	Courteney Cox	...	Monica Geller
	Lisa Kudrow	...	Phoebe Buffay
	Matt LeBlanc	...	Joey Tribbiani
	Matthew Perry	...	Chandler Bing
	David Schwimmer	...	Dr. Ross Geller
	Maggie Wheeler	...	Janice
	Christina Pickles	...	Judy Geller
	Elliott Gould	...	Jack Geller

Figure 3.17: Images of the actors from “Friends” next their names and their character names

is correct. There are some incorrectly labelled examples in the results but it can be assumed that the majority are correct. This provides positive training examples that can be used to train the classifier.

3.4.3 Random Forests

A random forest classifier is trained that is capable of classifying a face as one of the characters in the scene. The web data is used to give positive and negative examples of each character. The web images vary in appearance as they may be taken many years apart, with different hair styles and different makeup, whereas within in an episode they are likely to be more consistent. An example image of one of the actors, Jennifer

Aniston, with the facial features regressed is shown in figure 3.18. The regression has worked very well in this case with no obvious errors. Negative examples are taken from other scenes without that character so the classifier is better adapted to the data contained in the episode.

All the face candidates for a group of characters (Υ_m) are extracted from all the corresponding frames as in equation 3.11. There are also face descriptors v^w extracted from the web images, I^w , which belong to the actor for a known character (z). The set of face candidate Υ_z^w taken from the web images for a character are, $\Upsilon_z^w = v_{0\dots N}^w$.

Random forests ($R(v; \Omega)$) are trained where v is the face candidate to be classified and Ω is the data used for training. Positive training data for a character is taken from the web data, and the negative training data is taken from the video and the web. The labels of the training data are defined using $\Lambda(v)$, where 0 is negative and 1 is positive,

$$\Lambda(v) = \begin{cases} 1 & \text{if } v \in \Upsilon_z^w \\ 0 & \text{if } v \notin \Upsilon_z^w \end{cases} \quad (3.21)$$

The training data Ω_z^m , for a particular character in a character combination m is given as,

$$\Omega_z^m = \{v | v \in \Upsilon_z^w \forall z \in B_m \text{ or } v \in \Upsilon_m \forall p \text{ where } z \notin B_p \text{ and } |B_m \cap B_p| \geq 1\}. \quad (3.22)$$

To avoid the problem of bias in the training data, the negative training data is randomly sampled to balance it against the number of positive examples. The character label is found by taking the argmax over each of the random forests. To provide temporal consistency, the largest number of predicted face labels within a face track Y is used to label the rest of the track. Training a random forest with 1,000 trees and a maximum depth of 10 can be completed in minutes on a modern Intel Core i7-based machine.

3.4.4 Results

The use of labelled data from the internet allows for a large increase in the performance of the character labelling. The confusion matrix when using the SIFT-based descriptor is shown in table 3.5, and the corresponding results table is shown in table 3.6.

	Monica	Rachel	Mr. Geller	Phoebe	Mrs. Geller	Janice	Ross	Chandler	Joey
Monica	0.64	0.00	0.04	0.08	0.00	0.00	0.07	0.00	0.17
Rachel	0.11	0.43	0.00	0.00	0.02	0.00	0.44	0.00	0.00
Mr. Geller	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00
Phoebe	0.19	0.00	0.00	0.65	0.00	0.00	0.13	0.00	0.03
Mrs. Geller	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00
Janice	0.00	0.00	0.00	0.00	0.00	0.79	0.00	0.00	0.21
Ross	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00
Chandler	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.99	0.01
Joey	0.00	0.00	0.00	0.00	0.00	0.03	0.00	0.01	0.96

Table 3.5: Confusion matrix for the results of the automatic classification using a random forest with the SIFT-based descriptor



Figure 3.18: An example image of Jennifer Aniston with the facial landmarks and estimated pose highlighted

Many characters now have near 100% accuracy with the average accuracy over all 9 characters being nearly 83% compared with 69% when using the GMMs. However, the classification performance for Rachel is reduced and the performance is lower for females in general. This may be due to biases in the regressor training set. It could also be how the regressor is affected by the higher variability of females due to makeup or change in hair styles. Rachel is classified as herself slightly less often than she is classified as Ross, despite them being different genders. This again shows a poor ability for the system to discriminate between genders. Monica is often confused with Joey as is Janice. Though the confusion between Janice and Joey is much lower than when using the GMM-based method.

The performance when using the OpenFace descriptor is shown in the confusion matrix in table 3.7 and the table of results 3.8. Performance approaches 100% for all of the characters in the episode with the lowest performance being 87%, which is still better than the average performance of the method when utilising the SIFT-based descriptor. Rachel is the lowest performing character with both descriptors, though the OpenFace descriptor provides better performance. In both cases, the confusion is mostly with Ross. Likewise, there is still confusion between Joey and Janice, meaning their similar fringes may still be a factor with the OpenFace descriptor. The performance when

Character	Correct Samples out of Total	Percentage
Monica	1062/1661	63.94
Rachel	429/995	43.11
Mr. Geller	278/278	100.00
Phoebe	855/1324	64.58
Mrs. Geller	171/171	100.00
Janice	170/215	79.07
Ross	1803/1808	99.72
Chandler	1264/1281	98.67
Joey	616/644	95.65
Character Average		82.75

Table 3.6: Table of results for the classification and the performance average for all characters

using OpenFace with the web data should be acceptable for providing character labels to a caption generation method.

3.5 Chapter Conclusion

In this chapter there are three methods for the automatic classification of characters in broadcast video. Each of the methods was tested using a SIFT-based face descriptor and a CNN-based face descriptor with an identity-based embedded representation. The first method uses clustering to group the face descriptors together, then labels them using the cluster size combined with the histogram of character appearances. This achieves only 29% accuracy when using the SIFT-based descriptor and a slightly improved performance of 31% with the CNN, though more pure clusters are created. Simple clustering has limitations in the modelling of the face distributions and how the labels are assigned. To improve on this, a method that uses GMMs to represent the distribution of faces was also investigated. To ensure that the data is clustered by difference in identity, random forests were trained to separate different people, and

	Monica	Rachel	Mr. Geller	Phoebe	Mrs. Geller	Janice	Ross	Chandler	Joey
Monica	0.96	0.01	0.00	0.03	0.00	0.00	0.00	0.00	0.00
Rachel	0.00	0.87	0.00	0.00	0.02	0.00	0.11	0.00	0.00
Mr. Geller	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00
Phoebe	0.02	0.00	0.00	0.98	0.00	0.00	0.00	0.00	0.00
Mrs. Geller	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00
Janice	0.00	0.00	0.00	0.00	0.00	0.93	0.00	0.00	0.07
Ross	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00
Chandler	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00
Joey	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.01	0.98

Table 3.7: Confusion matrix for the results of the automatic classification using a random forest with OpenFace descriptors

Character	Correct Samples out of Total	Percentage
Monica	1588/1661	95.61
Rachel	867/995	87.14
Mr. Geller	278/278	100.00
Phoebe	1300/1324	98.19
Mrs. Geller	171/171	100.00
Janice	200/215	93.02
Ross	1807/1808	99.94
Chandler	1275/1281	99.53
Joey	631/644	97.98
Character Average		96.82

Table 3.8: Table of results for the classification and the performance average for all characters

these forests were used to represent the face descriptors with fewer dimensions in an identity-based space. To separate sets of characters, rules were created to combine the distributions and isolate characters. A performance of over 60% was achieved using this method. Some confusion between characters occurred due to the limitations of the descriptor, notably when confusing two characters with similar fringes. When using the OpenFace descriptor with the GMMs and separation rules, this performance increased to 85%. This is due to the better representation of identities by the OpenFace descriptor when compared with the random forest output.

Another method that utilises further information harvested from the internet was demonstrated. This uses a combination of web data and data from the video to train multiple detectors for the characters in the video. Using this method, a performance of over 80% was achieved with the SIFT-based descriptor and 97% when using the OpenFace descriptor. This is a level of performance that is suitable for use as input to an automatic description generation pipeline for broadcast video.

Chapter 4

Caption Generation with Contextual Input

Generating natural language descriptions from image inputs is a difficult problem. It requires not only recognising the objects, actions and people in the scene but also how they fit together to form a complete sentence. The current state-of-the-art techniques for object recognition involve using Convolutional Neural Networks (CNNs) such as GoogleNet [158]. CNNs are examples of deep learning, where the features in the image are themselves learnt rather than relying on engineered features. The generation of complete sentences requires knowing how these components fit together i.e. grammar. The use of an existing caption generation technique (Im2Text [123]) was tested on data from “Friends”. The system was based on retrieving captions from the nearest neighbour image in the training set. This was chosen because of the size of the dataset with 1 million captions. Due to the differences between the content in the show and Flickr images in the dataset, the resulting captions were not very relevant. An example of the poorly fitting caption generated is shown in figure 4.1. In addition, a CNN [85] trained on ImageNet was used on objectness [2] windows detected in frames from “Friends”. The objectness score is a technique to pre-filter parts of an image based on how likely they are to contain an object of any sort, compared with background image. The CNN was used to classify objects into 1,000 categories from the ImageNet challenge. While some good results were achieved, the overlap of objects in the TV show and the cate-



Figure 4.1: Image from “Friends” given the caption “This little pink flower was so alone in the sea of green” by a pre-trained text description caption generation system that uses the nearest neighbour from a database of images and captions

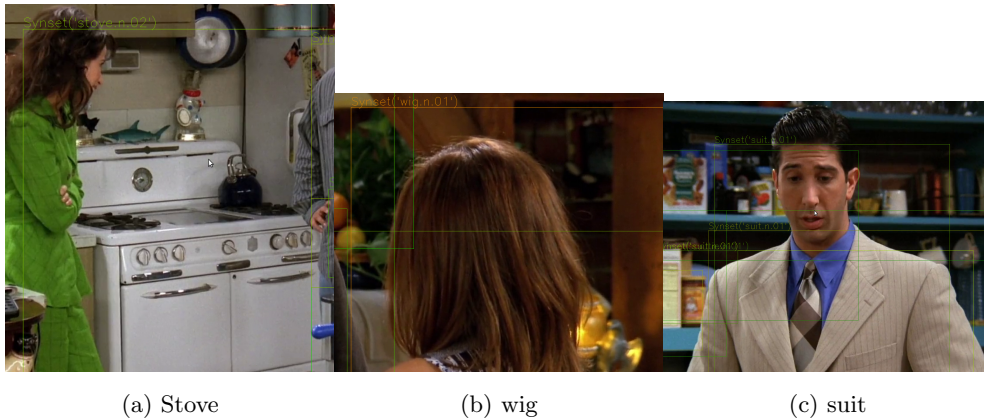


Figure 4.2: Example CNN classifications on the object windows

gories in the ImageNet challenge was low. Some example objects detected are shown in figure 4.2. The classification of “stove” on the window in figure 4.2(a) is correct. The labels of “wig” and “suit” in figures 4.2(b) and figure 4.2(c), respectively, are only partly correct although it is possible to see why the classification decisions were made.

Analysis of the scripts available for “Friends” showed that there are few descriptions available therefore it was decided that it is not a suitable dataset for training a caption generation system. The Microsoft Common Objects in Context (MSCOCO) dataset provides a large collection of images containing a variety of objects and people within common settings. It consists of 80,000 training images, 40,000 validation images and a test set of 40,000 images with hidden labels. Each of the images contains a subset

an orange and white cat using a lap top computer on a desk
a cat sits with paws on a laptops keyboard.
a cat with his hands on a lap top computer.
a cat looking at computer while sitting on the keys.
a cat is sitting on a laptop computer.

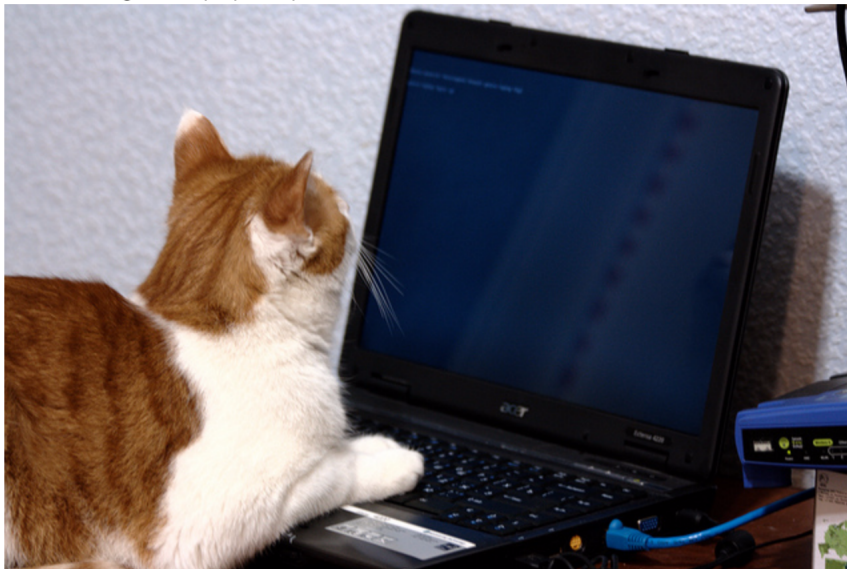


Figure 4.3: Example image from the MSCOCO dataset with its 5 captions

of the 80 object categories and is annotated with a full segmentation for these objects along with five natural language descriptions written by humans. Figure 4.3 shows an example of one image taken from MSCOCO with the five accompanying captions. The variation in the captions can be seen including spelling, and grammar mistakes. The 80 object categories were chosen to reflect the most frequent objects. The objects include animals, furniture and vehicles.

One method for generating descriptions using CNNs is to predict the words present in an image and using a probability-based language model to assemble them into a sentence. A different CNN can be used for different parts of speech as they can reflect different visual elements in an image. The language model needs to be trained on a large training set and needs to be able to use multiple previous words to predict the next word. Many existing CNN models are trained to recognise single classes in an image. Multiple objects appear in the majority of the images in the MSCOCO, meaning a method to predict multiple classes is required.



Figure 4.4: Image from the MSCOCO dataset with the highest probability nouns and verbs that have been predicted

4.1 Noun and Verb Prediction

Language contains many parts of speech, relating to different types of words used for different purposes. Nouns, for example, can loosely be defined as relating to things such as objects, people or ideas. Verbs are words used to describe actions. More abstract concepts may be harder to predict directly from the pixels in an image, and may require pre-existing knowledge. This is something that is easy for humans but much more difficult for computers. Figure 4.4 shows an example image from the MSCOCO dataset with the highest probability nouns and verbs predicted by the technique described in this section. The words that have been predicted are correct, but it is lacking the structure and grammar required for a natural language caption.

4.1.1 Language Processing

To predict the words in an image, a CNN can be used, but this requires training labels for each of the input images. To be able to train a different CNN for different parts of speech, first, the different parts of speech need to be separated. In different

contexts the same words can belong to different parts of speech. For example, “book” can be both a noun and a verb. In the same way, the same word can have multiple senses. Many parts of speech taggers exist that use different methods and are trained on different datasets. The first process is to tokenise a sentence so that each word is a token. Hyphenated words, contractions and possessives may also be separate tokens. A Part of Speech (POS) tagger can use a Hidden Markov Model (HMM), sets of rules or machine learning.

To train a machine learning algorithm to predict words describing an image, a ground truth training label needs to be created. One way to do this is by storing a sparse vector with a different element for the each word. A POS tagger is used to tokenise the sentence and the most common words for each part of speech are found by creating a histogram. Limiting to the most common words is required, as less frequent words will have too few examples to be learnt during training. Where there are multiple sentences, the words from each sentence can simply be combined into a single vector.

Given a set of captions $Y = y_{0...n}$, the set of all words (T_l) belonging to a particular part of speech l are found. Each word \mathbf{y} within a caption (y) is separated by its associated part of speech (l) to obtain a set of words for each part of speech, l ,

$$T_l = \{\mathbf{y}_i | \text{POS}(\mathbf{y}|y_i) = l\}. \quad (4.1)$$

This defines a set of classes for the dataset but the training label for each image still needs to be created. The binary vector q_l^i for each image is,

$$q_l^i[n] = \begin{cases} 1 & \text{if } T_l[n] \in y_i \\ 0 & \text{otherwise.} \end{cases} \quad (4.2)$$

4.1.2 CNN fine tuning

A different CNN can be used for different parts of speech because different parts of speech will relate to different types of content in the image. Using a single CNN to learn them all will be more complicated. The 2 main parts of speech are nouns and verbs. The verbs used are dependent on the nouns that are used. This means

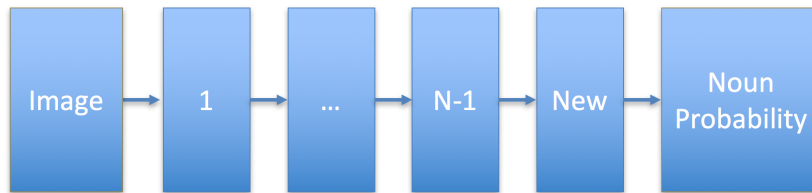


Figure 4.5: Arrangement of the layers of the CNN when fine-tuning for noun prediction

that the output of the noun CNN can be used as an extra input to the verb CNN to provide context. Training a CNN requires a very large amount of training data, due to the number of free parameters that need to be optimised. The solution is to use a CNN that has already been trained for one image recognition task and to adjust the parameters of the CNN to perform the new task. This is called fine-tuning and allows a smaller training set to be used for training a CNN.

When fine-tuning, the final hidden layer of the network is randomly initialised, then trained with a high learning rate. The earlier layers have a learning rate set either to 0 or reduced from their original values. This is to make sure the final layer trains to fit to the new training data, whereas the earlier layers are assumed to be less specialised and require less adjustment. A sigmoid cross-entropy loss layer,

$$E = \frac{-1}{n} \sum_{n=1}^N [p_n \log \hat{p}_n + (1 - p_n) \log(1 - \hat{p}_n)], \quad (4.3)$$

allows the network to be trained to regress the sparse vectors. The sigmoid function constrains the values between 0 and 1 but allows multiple classes to have a high probability. The softmax function,

$$\hat{p}_{nk} = \exp(x_{nk}) / \left[\sum_{k'} \exp(x_{nk'}) \right], \quad (4.4)$$

that is commonly used is designed to give a single class a high probability and have the elements sum to 1, and therefore it is not suitable when classes are not mutually exclusive.

Figure 4.5 shows the arrangement used for fine-tuning an existing CNN to predict the probabilities of nouns. The verbs that are used in a sentence will be influenced by the

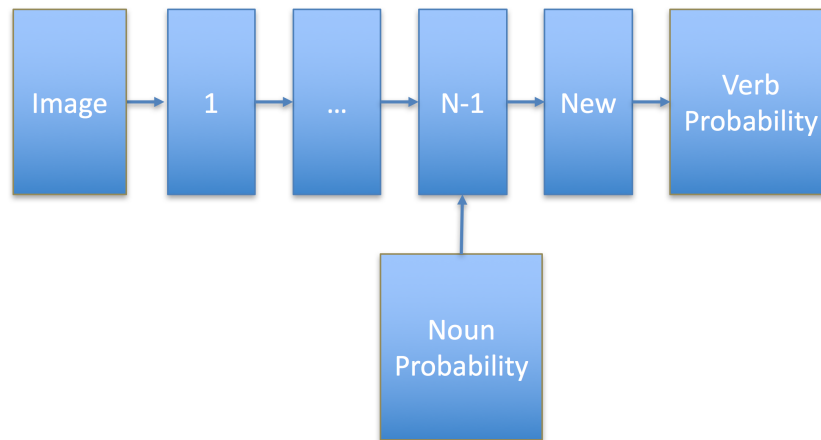


Figure 4.6: Arrangement of the layers in the verb CNN

nouns that are used. To give the verb CNN the context of the nouns in the image, the noun probabilities are concatenated with the penultimate layer of the CNN. This allows the final layer to use both the image and noun information to predict the verbs that are present in the image. The layout of the verb CNN is shown in figure 4.6. At training time, the noun probabilities are taken from the training set, but at test time, the probabilities are taken from the output of the noun CNN.

4.2 Language Model

With the methods for predicting the probabilities of verbs and nouns, there needs to be a way to form these predictions into sentences. Given the large body of text present in the training set, a language model can be trained. The probability of every other word appearing after the current word is calculated, giving the bigram probability. The idea can be extended to the probability of the second word after the current word to get the trigram probability. These probabilities of words appearing after other words are combined with probabilities of the predicted words from the images to create sentences. Each sentence ends with a full stop so the probability of a full stop after each word is also calculated. This allows the probability of a sentence ending to be calculated.

4.3 Sentence Prediction

To generate a caption for a new image that is not in the training set, the image is first put into the noun prediction CNN to get the probabilities of the nouns present in the image. The image is then put into the verb prediction CNN, along with the probabilities of the noun predictions. A diagram of how the sentences are generated is shown in figure 4.7. Given that the probabilities for part of speech (l) other than nouns and verbs are not predicted, their prior probability in the data is given as an input to the language model. The sentence generation is initialised with the most likely noun in the image. The probabilities of the predicted words are multiplied with the probability of the word appearing after the current word, and then the probability of all other words are multiplied by a regularisation coefficient. The maximum probability is used to choose the next word, which then becomes the current word. This continues until the next predicted word is a full stop or the maximum length is reached. This is a greedy approach, as the most likely next word is taken in each situation rather than optimising for the most likely overall sentence. Other methods exist for finding the path through the different possible words at each stage but at this point in the research, the use of Recurrent Neural Networks (RNNs) was emerging and these were explored instead.

4.3.1 Evaluation Metrics

Once descriptions have been generated for the unseen images, a methodology is required to evaluate how the captions compare to the reference captions. Doing this manually for 10,000s of images is implausible and would not necessarily be consistent. To circumvent this, there are several metrics used for automatic scoring of sentences. Many are also used for the scoring of automatic language translation methods. Bilingual Evaluation Understudy (BLEU) [127] is a simple score based on the precision of n-grams in the generated sentence and the reference sentences. The BLEU-n value relates to the length of the n-gram used. This measures precision but not the recall of n-grams in the reference sentences.

To measure the recall there is another metric, Recall-Oriented Understudy for Gisting Evaluation (ROUGE)-L [98]. This uses the longest common subsequence of n-grams

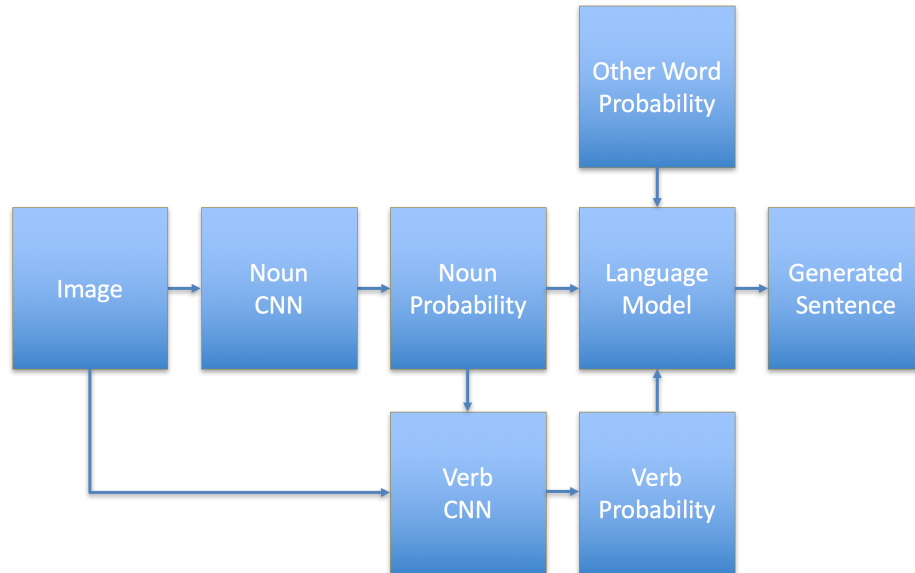


Figure 4.7: Diagram of the flow of data when generating captions for a new image

between the generated sentence and the reference sentences. The Metric for Evaluation of Translation with Explicit ORdering (METEOR) [7] method tries to balance both precision and recall to get a better agreement with humans. It uses the harmonic mean of the unigram precision and recall. It also calculates a penalty based on the fragmentation of the sentence in chunks of unigrams that are aligned. The greater the number of chunks, the higher the penalty. In addition, METEOR takes into account the root form of the words by using a technique called stemming. This is where words are reduced to their root form e.g. “researching” becomes “research”.

Another method, Consensus-based Image Description Evaluation (CIDEr) [165], tries to capture the consensus. Like METEOR the words are mapped to their root form. To measure consensus, the metric tries to measure how often the n-grams in predicted sentences appear in the reference sentences. However, n-grams are weighted by how often they appear in the dataset in general to limit the affect of common n-grams that do not provide useful information. The score is based on the cosine similarity between the generated and the reference sentences. This is designed to account for both precision and recall. N-grams of up to length 4 are used and the score for each

n-gram length is weighted equally.

4.3.2 Probabilistic Model Results

The CNNs used are based on the AlexNet [85] architecture. The network offers good performance and can be trained using relatively low computing power. AlexNet has 8 layers in total, 5 convolutional layers, followed by 3 fully connected layers. The noun and verb networks were fine-tuned for 100,000 iterations each. They were trained on the training set of 80,000 images from the MSCOCO dataset. For evaluation, the validation set of 40,000 images was used. Although there is a test set of 40,000 images, the captions are not publicly available. This means the only way to evaluate is by uploading results to Microsoft's server. There are limitations on the number of times new results can be uploaded to avoid attempts to optimise for the test set.

The results of the prediction system with different language models and different vocabulary sizes are shown in table 4.1. There is an increase in performance for most metrics when moving from a bigram language model to the trigram language model, where the metrics that are more related to the sentence structure improve. Using a trigram model but only taking a single clause gives a very low performance and shows that a great deal of performance comes from the additional clause.

Using the trigram model with a reduced number of nouns from 1,000 to 305 causes a drop in performance, although relatively small compared to the number of words that are dropped. This shows that the majority of the performance comes from the most commonly used words. Two different CNNs were trained and evaluated, using 1,000 nouns, and 305 nouns. Two verb CNNs were trained to output 100 verbs using different number of noun inputs. The vocabulary size of nouns and verbs was chosen based on the drop off in frequency of use.

Another experiment was to use the ceiling of the CNN output values so that the prediction was binary, and the language model only used the presence of the word rather than its probability. This caused a drop in performance, so it shows that being able to weight a word by its predicted probability is a useful function of the model. Less likely

Word Source	Language Model	Nouns	Verbs	BLEU_1	BLEU_2	BLEU_3	BLEU_4	METEOR	ROUGE_L	CIDEr
CNN	Trigram, 1 clause	1000	100	0.384	0.259	0.169	0.102	0.154	0.342	0.423
CNN	Bigram	305	100	0.564	0.371	0.209	0.109	0.189	0.402	0.458
CNN	Trigram	1000	100	0.584	0.352	0.207	0.112	0.200	0.370	0.515
CNN	Trigram	305	100	0.578	0.348	0.203	0.111	0.198	0.368	0.496
Ceil of CNN	Trigram	1000	100	0.571	0.342	0.197	0.107	0.201	0.365	0.486
Ground Truth	Trigram	305	100	0.777	0.518	0.332	0.202	0.265	0.439	0.784
Ground Truth	Trigram	1000	100	0.784	0.523	0.338	0.208	0.268	0.441	0.802
Closest Word	Ground Truth	305	Original	0.904	0.849	0.799	0.751	0.544	0.884	1.847
Closest Word	Ground Truth	1000	Original	0.941	0.904	0.869	0.836	0.640	0.926	2.114

Table 4.1: Results of the prediction using different word vocabularies and different language models

words predicted for the image would be treated the same as words that have a higher output probability, which could lead to incorrect words being chosen.

To measure the performance of the language model separately from the CNN, the language model was tested using the ground truth word labels with different vocabulary sizes. The performance here is surprising as despite the simplicity of the language model, it is capable of achieving a very high performance when given ground truth input. This shows that the language model in isolation is effective and that the performance could be improved if the noun and verb prediction was improved. The change in performance between 305 and 1,000 nouns is still notably small. That is a large drop in vocabulary for a relatively small drop in performance. The MSCOCO dataset has a focus on a narrow domain of 80 object categories, so the vocabulary required is quite small compared with standard English.

The effect of reducing the vocabulary size in the ground truth sentences was tested by replacing words outside of the vocabulary with their closest word inside the vocabulary. The closest word was found by using the Word2vec word vectors [113]. Word vectors map words into a semantic space so that similar words are close together. Sentences containing words outside of the noun vocabulary of 305 or 1,000 had these words replaced with the word that was closest in the Word2vec space. A pre-trained model from Google's word2vec was used and was trained on a very large corpus of internet news data. Reducing the vocabulary size from 1,000 to 305 when using the ground truth sentence, has a much larger effect on performance as seen in table 4.1. This is likely to be because it is the only factor, whereas before the quality of the language model and the predictions was also affecting the performance. Alternatively, the performance of the word vector's closest match may not be good enough.

4.4 Language Analysis

Kučera and Francis [87] presented the word frequency counts on the Brown corpus, and this was long used as the standard. More recent word frequency counts are SUBTLEX-US by Brysbaert and New [18] and Corpus of Contemporary American English (COCA) by Davies [32]. SUBTLEX-US uses a corpus of American subtitles, and COCA uses a

POS	Percentage of Words	BLEU_1	BLEU_2	BLEU_3	BLEU_4	METEOR	ROUGE_L	CIDEr
Prepositions	14.95	0.950	0.840	0.714	0.604	0.481	0.894	1.745
Determiners	18.66	0.914	0.825	0.732	0.648	0.488	0.863	1.821
Adjectives	6.20	0.982	0.952	0.921	0.891	0.553	0.954	2.330
Nouns	32.7	0.745	0.592	0.452	0.344	0.260	0.745	0.623
Verbs	10.65	0.962	0.906	0.843	0.783	0.497	0.924	2.004
Coordinating Conjunction	2.10	0.996	0.984	0.972	0.961	0.677	0.987	2.576

Table 4.2: Parts of speech and what percentage of the words in the dataset belongs to each one along with the scores that the ground truth receives when that part of speech is removed

	Train			Validation		
Position	Word	Count	SUBTLEX-US	Word	Count	SUBTLEX-US
1	a	684,534	6	a	335,181	6
2	.	310,734	N/A	.	152,254	N/A
3	on	150,606	19	on	74,034	19
4	of	142,756	11	of	69,917	11
5	the	137,946	3	the	68,185	3
6	in	128,895	13	in	62,747	13
7	with	107,690	36	with	53,506	36
8	and	98,721	10	and	47,848	10
9	is	68,663	15	is	33,721	15
10	man	51,200	87	man	24,571	87
11	to	47,728	4	to	23,428	4
12	sitting	36,796	797	sitting	18,359	797
13	an	34,980	86	an	17,003	86
14	two	34,083	134	two	16,309	134
15	standing	30,015	931	at	14,659	56
16	at	29,842	56	standing	14,219	931
17	people	29,494	127	people	13,999	127
18	are	28,809	34	are	13,960	34
19	next	25,912	254	next	12,951	254
20	white	25,172	537	white	12,246	537

Table 4.3: Word frequency for the training and testing data in MSCOCO along with the SUBTLEX-US positions of each word

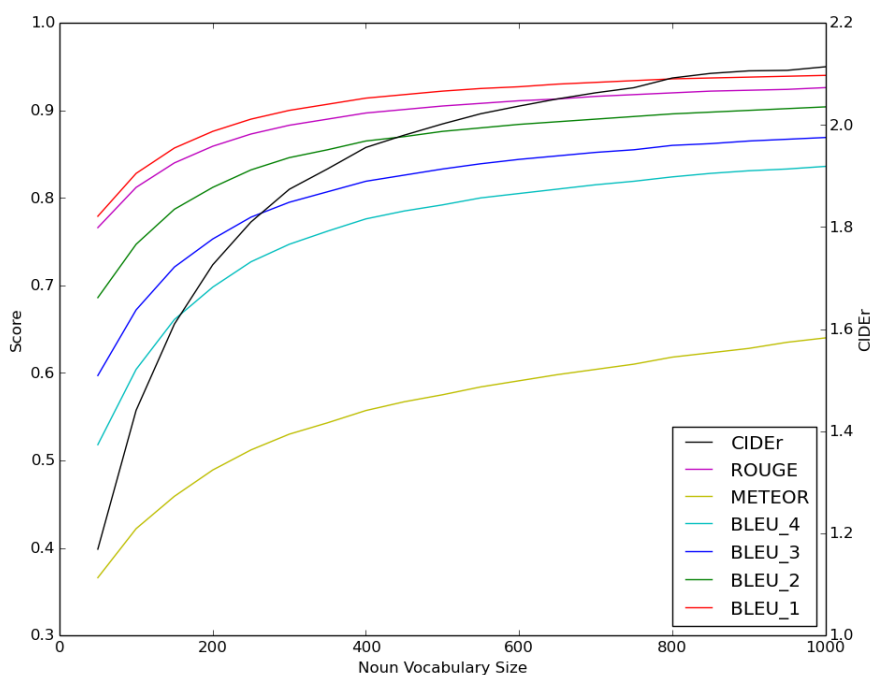


Figure 4.8: Graph of the different performance metrics as the number of nouns available to use changes

corpus of 560 million words and was updated in December 2017. The most frequent words in the MSCOCO training and validation set are shown in table 4.3 next to the position of each word in the SUBTLEX-US data. Most of the words are highly used in standard English, as expected. It is a specialised dataset of descriptions of images, so its word frequencies will not directly match standard English. The training and validation set match each other almost exactly with only the order of “at” and “standing” changing. A model created from the training set should have a suitable vocabulary for the validation set.

The word frequencies in the captions generated from the probabilistic model are shown in table 4.4. The most common words vary from the original dataset. There are more nouns and verbs, this may be a bias from the nouns and verbs predicted by the CNNs. The frequency is lower as there is only one sentence per image, rather than five like the original dataset. “Two” is not present in the high frequency words, this is likely due

	Probabilistic Model		
Position	Word	Count	SUBTLEX-US
1	a	74,045	6
2	and	44,415	10
3	on	30,204	19
4	is	18,309	15
5	in	16,979	13
6	man	16,698	87
7	sitting	16,075	797
8	of	10,365	11
9	are	9,967	34
10	standing	9,658	931
11	people	8,482	127
12	holding	8,159	1,039
13	table	7,779	721
14	field	7,771	1,015
15	street	6,292	586
16	riding	5,924	1,855
17	with	5,796	36
18	plate	3,659	2,199
19	group	3,507	973
20	room	3,098	261

Table 4.4: Word frequency for the output from the language model when using a trigram-based model with a vocabulary of 1,000 nouns

to the fact that the model has no way to perform any counting.

The average sentence length in the training and validation set are both 10.61 words compared with an average sentence length of 14.22 words in the output of the probabilistic model. The average sentence length could be used to help choose when to terminate the sentence but could also lead to sentences being cut short.

To show the effect of the noun vocabulary size on sentence quality, the number of nouns available to use in a sentence was varied between 50 and 1,000. The Performance increases quickly at first as the dictionary size increases but then saturates and only increases slowly with diminishing gains. The graph of the different performance metrics against dictionary size is shown in figure 4.8.

So far, only the influence of nouns and verbs has been analysed. To look at the influence of the other parts of speech on the performance metrics, each part of speech was removed from the sentences. This allows for the individual influence of each part of speech to be analysed. Table 4.2 shows the results of removing the different parts of speech from the ground truth sentences. Unsurprisingly, nouns are very important as they will often be the subject of the sentence, and the dataset is built around descriptions of the objects in images. More surprisingly, the effect of the verbs is lower than expected. Prepositions, which are used to express spatial position are of a quite significant importance. Determiners are important but are more part of the language model than something predicted from the image directly. Adjectives that provide descriptions, only make up a small percentage of the words so have a small effect on the metrics. Coordinating conjunctions e.g. “and” make up only 2% of the words in the dataset and will come from the language model rather than image features.

To investigate the importance of the language model compared with the generated words, the word order in a ground truth sentence was randomised to remove the grammar. The results of this are shown in table 4.5. The BLEU1 score is unchanged as it is the unigram precision. The other metrics are greatly reduced, showing how important the word ordering is for the evaluation metrics that measure more complex rules of language than just single words being correct. The BLEU4 score is very low as it relies on there being 4 words in the correct order after being randomised.

BLEU_1	BLEU_2	BLEU_3	BLEU_4	METEOR	ROUGE_L	CIDEr
1.000	0.402	0.148	0.054	0.415	0.467	1.058

Table 4.5: Evaluation metrics on ground truth sentences where the word order is randomised

4.5 Preposition Prediction

Prepositions are another l that describe the relative positions of objects e.g. the cat is **on** the table. The most frequent prepositions featured in the captions in the training set are shown in table 4.6. “While” can be considered as a preposition but is an outdated usage. Word frequency counts in SUBTLEX-US combine all senses, whereas COCA separates each word based on l . This can account for the large disagreement between the sources for some words. The MSCOCO data is a much smaller body of text and is heavily biased due to the dataset describing the content of images. For example, “near” might be commonly used to describe the relative position of something in an image but used less frequently in general text or spoken language.

Given that prepositions are used to describe the relative position of items, location information is relevant when predicting them. The MSCOCO dataset contains location annotation for the 80 object classes in the dataset. An example image with the segmentation overlaid is shown in figure 4.9.

To test the feasibility of predicting prepositions based on object positions, the ground truth bounding boxes were used to create features. The prepositions were extracted from the sentences, based on the part of speech tagging described previously. The prepositions were used as the labels to train a random forest for classification. A random forest was used as they are able to perform multi-class classification, require a relatively low amount of training data, and the decision process is more easily understood by a human. The dataset was searched to locate sentences containing text in the form of “object 1 preposition object 2”. Additional words were allowed between the objects and the preposition to account for determiners. This limited the amount of data that could be used for training. The features created from the MSCOCO segmentation use

Preposition	Frequency Count	Position in SUBTLEX-US	Position in COCA
on	150,665	19	17
of	142,759	11	4
in	128,884	13	6
with	107,703	36	16
at	29,845	61	22
down	15,098	104	1,027
next	14,535	254	1,518
near	13,259	889	707
by	11,949	112	30
while	10,088	304	N/A
for	7,646	20	13
over	6,633	113	124
from	6,533	76	26
through	6,069	209	112
around	6,050	168	265
as	5,109	72	49
outside	4,609	540	812
that	4,568	9	N/A
under	4,466	383	226
behind	4,423	504	422

Table 4.6: Frequency count of the top 20 Prepositions in the MSCOCO captions

a girl with a bike is sitting on a park bench.
 a woman sitting on a bench next to a her blue bike.
 a woman sits with a bicycle on a park bench.
 a woman sitting on a bench with her bike
 a woman is holding a bike sitting on a bench.



Figure 4.9: Example image from MSCOCO with the captions and segmentation overlay

the shape of the objects bounding boxes, their relative sizes, positions and overlap. The features described in the table are concatenated into a vector.

The “importance” of a feature is based on the percentage of splits in the random forest using that feature, and is shown in table 4.7. Each of the features are used roughly equally showing that no feature dominates and there are no redundant features. Given the 14 features, the average importance would be 7.1%.

The results of the random forest trained to predict prepositions from the features described earlier are shown in table 4.8. The training score is very high, but the score on the test set is much lower. There are 49 prepositions used in the training, and the scores for individual prepositions are shown in the table. The other prepositions not featured in the table all have a score of zero. The test score is better than random chance between the classes but not good enough to use in actual sentence prediction. The best performing preposition is “on”, as it is easily predicted based on the relative positions of two objects. “With” and “in” should also be predictable based on the

Feature Type	Importance
Object 1 X	7.7
Object 1 Y	7.7
Object 1 Width	8.3
Object 1 Height	8.0
Object 2 X	7.5
Object 2 Y	8.1
Object 2 Width	8.1
Object 2 Height	8.3
X difference	5.6
Y difference	6.7
Overlap	4.8
Euclidean Distance	6.2
Area 1 / Area 2	7.0
Difference in Euclidean Position	5.9

Table 4.7: Importance of different features for preposition prediction using a random forest

Features	Percentage
Train Score	82.97
Test Score	35.14
On	75.40
From	0.00
Of	4.26
Under	4.82
Next	3.43
Down	3.20
Behind	0.00
At	4.69
In	14.24
With	32.21
Through	2.38
Into	0.89
Like	0.00
By	0.00

Table 4.8: Results from the preposition prediction random forest

position features but only show a relatively poor performance. “With” is more general and may show more variation in the feature space. Other prepositions have very low or zero scores. For some prepositions such as “like,” it seems that they would be difficult to predict based on the positional features that are used. Some prepositions such as “from” may require additional knowledge that would be difficult to encode in the positional data used for this random forest. Temporal or external knowledge may be required to say that “something is coming from somewhere”, for example.

4.5.1 Results

Given the importance of the prepositions in the overall score of a sentence, it would be hoped that being able to predict prepositions from the image data would improve the results of the sentence generation. Using the prepositions predicted with the random forest with the trigram-based probability language model gives the results shown in table 4.9. The output of the random forest is a probability of the preposition based on the positions of objects in the image. These probabilities can be used as an additional input to the caption generation pipeline that was shown in figure 4.7. The results show that the preposition prediction lowers the performance of the pipeline. This is likely to be because the prediction quality of the prepositions is lower than the quality of prepositions that are found using the language model and the context from other words. The score on the test data is only 35.14%, which translates to only predicting the correct preposition roughly one third of the time. Given the incorrect preposition predictions, the overall performance is reduced. The biggest impact is on the CIDEr score, which gives a good indication of the drop in agreement with the reference sentences. The drop in BLEU scores is lower as they are more related to n-gram precision. Overall, these results show that predicting prepositions based on this positional features does not appear to be a viable way to increase performance of the language model.

4.6 Alternative approaches

A simple probabilistic model can be used to generate captions using CNNs to generate probabilities of words and then using the probabilities of n-grams to form sentences.

Method	BLEU_1	BLEU_2	BLEU_3	BLEU_4	METEOR	ROUGE_L	CIDEr
Before	0.584	0.352	0.207	0.112	0.200	0.370	0.515
Preps	0.519	0.315	0.190	0.107	0.209	0.364	0.399

Table 4.9: Table of old results and new results with prepositions



Figure 4.10: Basic high level example of an RNN

While meaningful captions are generated through this approach, the complexity of natural language means that performance is limited. The language model itself is very important, while the vocabulary only needs to be small. In the rest of chapter, RNNs are described along with how they are used to learn more complex language models for automatic caption generation. RNNs are like other neural networks, but they contain loops in the structure which allows them to “remember” the flow of information through the network. RNNs are the current state-of-the-art in the areas of machine translation and caption generation. They are capable of learning the rules of language from training examples without the need for any extra annotation.

Figure 4.10 shows a basic RNN design with a loop. This would be the same as having multiple copies of the same layer duplicated multiple times and then connected in a chain, as in figure 4.11.

The loops in the structure make the RNNs suitable for outputting sequences, as the next item is based on the previous state of the RNN. The RNN has an internal hidden state, which is updated at each time step of the network. As with other neural networks,

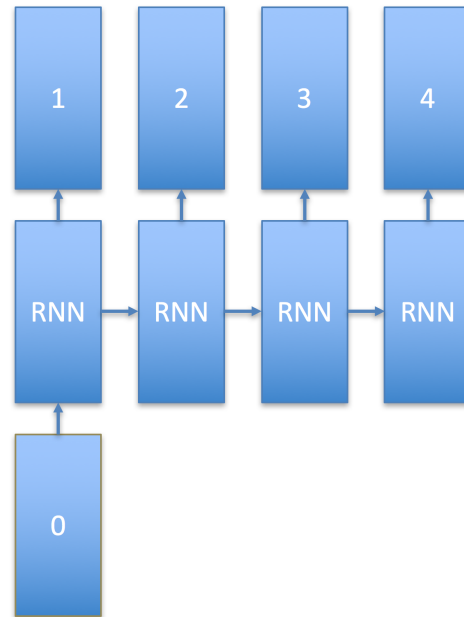


Figure 4.11: Diagram of an RNN with the layers unrolled showing inputs and outputs

non-linearities can be applied to create deep networks where the output of the RNN layer is fed into another RNN layer. This basic principle of RNNs is improved by the Long Short-Term Memory (LSTM) architecture, which is described in section 4.7.1. The following sections describe how RNNs are applied to automatic image captioning.

4.7 RNN

Bengio et al. [11] used an RNN design to learn language models that perform better than n-gram-based models. Socher et al. [151] used a deep RNN design for scene and sentence parsing. Karpathy et al. [81][80] use a bidirectional RNN design for generating captions based on input from the final layer of AlexNet [85]. Rather than using the final hidden layer of a CNN, Xu et al. [174] used the convolutional layers from VGGNet [146]. These represent different parts of images, to allow an RNN to learn visual attention and caption generation. Vinyals et al. [167] also used an RNN design with CNN input but with input from GoogleNet [158]. Given the effectiveness of GoogleNet, VGG, RNNs, and visual attention, creating a network that can take input from multiple CNNs and

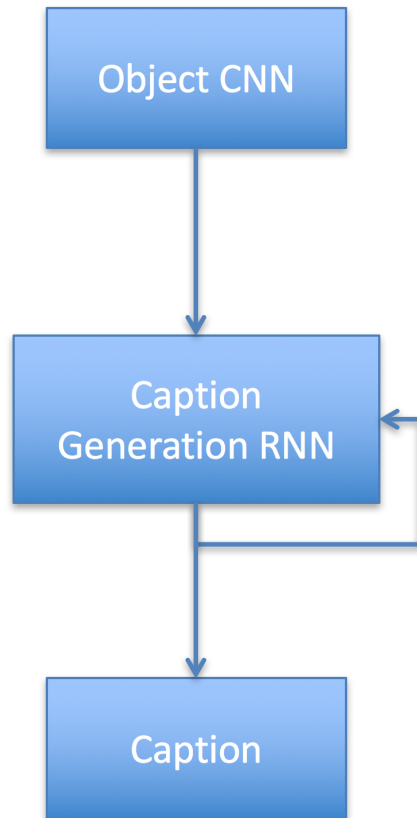


Figure 4.12: Overview of the pipeline

use both visual attention from convolutional layers and a high level representation of the whole image could be an effective method for generating captions.

Predicting single or multiple classes from an image using a neural network is described in section 4.1. To extend this idea to work for sequences of words, such as natural language, an RNN can be used. CNNs are very good at extracting features from images, which are used to identify their contents. The CNN architecture is used to provide inputs to RNNs for the purpose of caption generation. This is useful as the CNN has already accomplished the task of processing the image, and allows the RNN to focus on the language generation part of the problem. It is common to use the final hidden layer of a CNN as the input to an RNN as discussed in section 2.3.1. This high level overview of the architecture is shown in figure 4.12.

At each step of the RNN, the probability of the output words is predicted. Each word

is like a class in a classifier's output and is assigned an index. The training sentence is used to construct a sequence of indices and the output sentence is reconstructed from this. The RNN is trained to minimise the Negative Log Likelihood (NLL) of ground truth sentences.

The RNN contains an encoder step where the input words are encoded using an embedding matrix, to create an internal word vector representation. This represents the caption in a high-dimensional space that is useful to the RNN and has semantic meaning. The caption, y , is represented using a sequence of 1-of-K encoded words,

$$y = \{\mathbf{y}_1, \dots, \mathbf{y}_C\}, \mathbf{y}_i \in \mathbb{N}^k. \quad (4.5)$$

The captions are of length C with a vocabulary size of k .

The word vectors are created using an embedding matrix, $\mathbf{E} \in \mathbb{R}^{m \times k}$, where m is the dimensionality of the embedding.

A CNN is used to extract features from an image to get the initial input x_{-1} ,

$$x_{-1} = \text{CNN}(i). \quad (4.6)$$

The input at each time step of the RNN is given by the word from the previous time step and the embedding matrix,

$$x_t = \mathbf{E}\mathbf{y}_{t-1} \quad (4.7)$$

The word vectors and CNN features are used as the inputs to the RNN, and multiplied with its hidden state. Then the output is sent to output layers, which can also take other inputs before going through a softmax layer.

4.7.1 Long Short Term Memory

The RNN design works well with sequential signals but the relevance of previous states to the current output can vary in the number of time steps. In natural language, for example, the gender of the subject may only be relevant until a new subject is introduced. The LSTM architecture facilitates this by allowing the cell state c to be

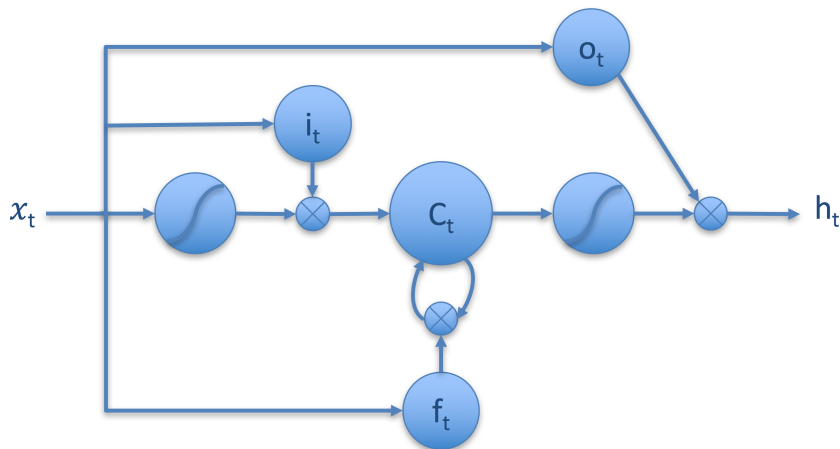


Figure 4.13: Diagram of an LSTM cell

controlled by gates. There are three gates: input (i), output (o), and forget (f). The gates consist of a sigmoid function σ to force the values to between 0 and 1 and then a multiplication to perform the gating. High activation of the sigmoid will allow data through and low activation will stop it. Figure 4.13 shows an overview of the LSTM layout.

The input gate is used to filter potential new input to the cell state,

$$i_t = \sigma(W_{ix}x_t + W_{ih}h_{t-1} + b_i). \quad (4.8)$$

The forget gate is used to purge the cell state of previous information that is no longer required,

$$f_t = \sigma(W_{fx}x_t + W_{fh}h_{t-1} + b_f). \quad (4.9)$$

The output gate is used to decide what from the cell state should be output to the new hidden state h_t ,

$$o_t = \sigma(W_{ox}x_t + W_{oh}h_{t-1} + b_o). \quad (4.10)$$

The cell state is set depending on the state of the forget and input gates,

$$c_t = f_t \odot c_{t-1} + i_t \odot \tanh(W_{cx}x_t + W_{ch}h_{t-1} + b_c). \quad (4.11)$$

Then the output gate is used to update the new hidden state,

$$h_t = o_t \odot \tanh(c_t). \quad (4.12)$$

The LSTM cell can be replicated many times, to create an LSTM layer of an arbitrary size.

4.7.2 Attention

Rather than using the final hidden layer of a CNN, it is also possible to use the convolutional layers as an input the RNN. The convolutional layers are 3-dimensional, with height, width, and a number of channels rather than just 1-dimensional. They represent activations corresponding to localised sections of the input image. This spatial information can be useful to locate parts of the image corresponding to specific words. The network needs to be trained to select the parts of the image where it should focus. It can do this by either using a weighted combination of the input or by selecting a single location. This is soft attention versus hard attention. Another weighting term (W_z) is added for each gate along with the attention vector (\hat{z}). This gives us updated equations for the LSTM,

$$i_t = \sigma(W_{ix}x_t + W_{ih}h_{t-1} + W_{iz}\hat{z}_t + b_i), \quad (4.13)$$

$$f_t = \sigma(W_{fx}x_t + W_{fh}h_{t-1} + W_{fz}\hat{z}_t + b_f), \quad (4.14)$$

$$o_t = \sigma(W_{ox}x_t + W_{oh}h_{t-1} + W_{oz}\hat{z}_t + b_o), \quad (4.15)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tanh(W_{cx}x_t + W_{ch}h_{t-1} + W_{cz}\hat{z}_t + b_c). \quad (4.16)$$

The attention vector (\hat{z}) is calculated using a neural network layer, which uses the previous hidden state to calculate weightings for the L different dimensions (i) of the convolutional layer (a) input,

$$e_{ti} = \text{att}(a_i, h_{t-1}). \quad (4.17)$$

The output of the network is used to create the attention weightings (α),

$$\alpha_{ti} = \frac{\exp(e_{ti})}{\sum_{k=1}^L \exp(e_{tk})}. \quad (4.18)$$

This allows the LSTM to focus on a different section of the input image. The weightings (α) are combined with the convolutional input (a). Using it as a probability for that dimension to give hard attention or by using it as a blending weight for soft attention. The function *phi* gives a single vector determined by the weights,

$$\hat{z}_t = \phi(a_i, \alpha_i). \quad (4.19)$$

The cell state and hidden state are initialised using the average of the convolutional input (a), each processed by a neural network.

4.7.3 General Context

While the attention provides the ability for the RNN to focus on different parts of the convolutional features, it can also be useful to still have more general features as input to the LSTM layers. This can be achieved by creating an LSTM that is capable of taking input from the word embedding matrix, the convolutional features from a CNN, and the final hidden layer of a CNN. This gives the LSTM the ability to select its output based on the previous words, where in the image the attention is required, and the contextual features extracted from the image as a whole. This LSTM design combines the designs of section 4.7.1 and section 4.7.2. The general context CNN features (v) can also be used as an additional input to the output layers of the RNN. A new weighting is required for each of the gates as the context (v) is input,

$$i_t = \sigma(W_{ix}x_t + W_{ih}h_{t-1} + W_{iz}\hat{z}_t + W_{iv}v + b_i), \quad (4.20)$$

$$f_t = \sigma(W_{fx}x_t + W_{fh}h_{t-1} + W_{fz}\hat{z}_t + W_{fv}v + b_f), \quad (4.21)$$

$$o_t = \sigma(W_{ox}x_t + W_{oh}h_{t-1} + W_{oz}\hat{z}_t + W_{ov}v + b_o), \quad (4.22)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tanh(W_{cx}x_t + W_{ch}h_{t-1} + W_{cz}\hat{z}_t + W_{cv}v + b_c). \quad (4.23)$$

The context is also used as an additional input to the output layer of the whole RNN design.

4.7.4 Beam Search

At each step in the RNN prediction process, there is a predicted probability for each of the words in the vocabulary. To generate an output sentence, a greedy approach can be used where the most likely word is taken and used. While the greedy approach can produce a good sentence, it is possible that the most likely sentence does not start with the first most likely word. With beam search, there is a preset beam size (n) that determines the number of potential sentences to consider. At each step the candidate sentences are scored and only n sentences are allowed to continue to the next step. The length of time to generate sentences scales linearly with the beam size.

4.8 Experiments

In these experiments, the RNNs were trained using only the training set from the MSCOCO dataset. The dictionary used for training was also generated from the captions in the training set. The standard performance metrics are reported for each of the models.

4.8.1 CNN Features

There are many different CNN designs and models available that can be used as input to the RNN. The CNNs that are tested here are the Oxford VGG 19 layer network and GoogleNet. VGGNet has a final convolutional layer of shape $14 \times 14 \times 512$, and GoogleNet has $7 \times 7 \times 1024$. GoogleNet is a state-of-the-art CNN with the VGG network being not far behind.

The results in table 4.10 show how the RNN performance on the MSCOCO validation set changes with the training epoch. An epoch is the number of times it has been trained on each example in the training data. The beam size was fixed at 1 and the RNN was trained using a convolutional layer from the final inception module in GoogleNet. Even after just 2 epochs, the performance is better than the best performance from the CNN and greedy probabilistic model in most metrics. The performance in all but BLEU

Epoch	BLEU_1	BLEU_2	BLEU_3	BLEU_4	METEOR	ROUGE_L	CIDEr
2	0.625	0.429	0.277	0.172	0.183	0.452	0.503
4	0.676	0.490	0.341	0.234	0.210	0.486	0.699
6	0.686	0.503	0.355	0.248	0.215	0.493	0.744
8	0.690	0.508	0.362	0.257	0.221	0.497	0.777
10	0.693	0.511	0.363	0.258	0.223	0.499	0.784
12	0.696	0.516	0.370	0.264	0.225	0.502	0.803
14	0.694	0.511	0.364	0.258	0.224	0.499	0.791
15	0.692	0.510	0.365	0.261	0.226	0.500	0.798
16	0.698	0.515	0.368	0.262	0.226	0.502	0.809
17	0.694	0.510	0.363	0.258	0.226	0.500	0.797
18	0.696	0.512	0.365	0.259	0.227	0.501	0.807
19	0.696	0.513	0.366	0.260	0.227	0.500	0.803
20	0.694	0.511	0.366	0.261	0.227	0.501	0.809
21	0.690	0.507	0.362	0.258	0.226	0.497	0.801

Table 4.10: Table of epoch scores for GoogleNet trained RNN with beam size of 1

Epoch	BLEU_1	BLEU_2	BLEU_3	BLEU_4	METEOR	ROUGE_L	CIDEr
2	0.625	0.429	0.277	0.172	0.183	0.452	0.503
4	0.666	0.478	0.325	0.213	0.201	0.479	0.616
8	0.682	0.500	0.349	0.238	0.214	0.492	0.718
10	0.689	0.506	0.355	0.245	0.216	0.495	0.739
12	0.691	0.511	0.361	0.251	0.219	0.498	0.758
14	0.698	0.514	0.362	0.251	0.221	0.500	0.774
16	0.697	0.512	0.360	0.249	0.220	0.498	0.768
20	0.694	0.513	0.364	0.256	0.221	0.499	0.777
22	0.693	0.512	0.362	0.255	0.222	0.499	0.776
24	0.701	0.516	0.365	0.254	0.224	0.501	0.789
26	0.696	0.514	0.364	0.255	0.223	0.500	0.783
28	0.696	0.513	0.364	0.256	0.223	0.500	0.784
–	0.668	0.493	0.349	0.246	0.225	0.498	0.773

Table 4.11: Table of results for the RNN with VGG context

1 is better than using the probabilistic model with ground truth noun input. As the training time increases, the performance increases quickly, before slowing down and then decreasing. The decrease is likely due to the model starting to overfit to the training data. The CIDEr score benefits the most from increased training time and is the metric that represents the best comparison for real world performance. The performance peaks at epoch 16.

4.8.2 Context Performance

The results showing the effects of the addition of the final hidden layer from the VGG CNN, to provide extra context to the RNN, are shown in table 4.11. The performance of the RNN is similar. Although it reaches peak performance at epoch 24, rather than epoch 16, more training time is required due to increased complexity. The top BLEU 1 score that is reached is higher although all other metrics seem to be lower. The beam

Beamsize	BLEU_1	BLEU_2	BLEU_3	BLEU_4	METEOR	ROUGE_L	CIDEr
1	0.701	0.516	0.365	0.254	0.224	0.501	0.789
2	0.706	0.528	0.381	0.272	0.228	0.508	0.821
3	0.706	0.530	0.386	0.279	0.229	0.510	0.831
4	0.705	0.530	0.387	0.281	0.230	0.511	0.833
5	0.705	0.530	0.388	0.282	0.230	0.512	0.836
6	0.704	0.529	0.388	0.283	0.230	0.511	0.836
7	0.704	0.529	0.388	0.282	0.229	0.511	0.835
GNET 5	0.688	0.510	0.375	0.277	0.232	0.504	0.833

Table 4.12: Results showing how the beam size affects the scores

search is disabled and the convolutional layer is also taken from GoogleNet.

4.8.3 Beam Size

The use of beam search allows for more possible paths through the output probabilities to be considered. Table 4.12 shows how the performance of the model from epoch 24 (in table 4.11) of the RNN with GoogleNet convolutional features, and the final layer from the VGG network as context, as the beam size changes. With the increase in beam size, the scores in all metrics increase. The BLEU scores increase quickly and then saturate, but the CIDEr score continues to increase up to a beam size of 5. Using the beam search allows the RNN with context to outperform the RNN without context, and get reach levels of performance. The CIDEr performance is now significantly better than the output of the probabilistic model, even when it is using ground truth input. For comparison, the result of changing the beam size to 5 for the GoogleNet RNN at epoch 16 from table 4.10 is shown in the bottom row. The performance also benefits from the beam search, but the BLEU 1 score decreases, and the CIDEr score does not reach the same level as with the context.

4.9 Performance Comparison

Caption generation has been a very active area of research in recent years. Many of the state-of-the-art models are based on RNN designs with RNN inputs. The MSCOCO dataset maintains a leaderboard of scores using the previously referenced metrics. Figure 4.14 shows the scores achieved by different methods against the date they were submitted to the leaderboard. The x-axis shows the number of days since the first submission to the leaderboard with day 0 being in March 2015, and the y-axis shows the scores for each of the metrics. The flat lines represent the best scores achieved by the RNN with context input. For most metrics, there is a small upward trend, but the standard of performance is still in a similar range to that achieved using the RNN design in this chapter. The biggest gains and the biggest upward trend are with the CIDEr metric. This metric tries to represent a consensus between the generated caption and the reference captions. It may represent an increase in the caption performance that is not well represented by the alternative metrics. The work of Mun et al. [116] represents the current state-of-the-art on the MSCOCO dataset. This is still using an RNN with visual attention. It features a process where a nearest neighbour caption is found and used to guide the visual attention for generating another, more detailed, caption.

4.10 Vocabulary Analysis

Figure 4.13 shows the top 20 words used in the RNN with, and without context. The position of the words in the SUBTLEX-US [18] data is also shown for reference. The words are roughly the same although the ordering of the words is slightly different. Compared with the probabilistic model in table 4.4, the top vocabulary more closely matches the original dataset shown in table 4.3. Like the probabilistic model, the number “two” does not appear in the top words. The RNN does not explicitly have any mechanism for counting objects, so might struggle to learn when to use numbers properly.

The RNN with context produces sentences with an average length of 9.84 words, while

	Without Context			Context		
Position	Word	Count	SUBTLEX-US	Word	Count	SUBTLEX-US
1	a	87,699	6	a	87,051	6
2	.	40,497	N/A	.	40,504	N/A
3	on	16,336	19	of	20,249	11
4	of	14,208	11	on	18,825	19
5	in	12,184	13	in	12,692	13
6	with	11,369	36	with	8,217	36
7	is	9,102	15	sitting	7,567	797
8	and	9,081	10	the	6,721	3
9	man	8,076	87	standing	6,576	931
10	the	7,161	3	man	6,452	87
11	sitting	5,813	797	is	6,243	15
12	standing	5,275	931	and	5,228	10
13	group	4,300	973	table	5,128	721
14	down	3,971	104	group	5,112	973
15	street	3,693	586	people	4,850	127
16	people	3,521	127	street	3,992	586
17	woman	3,474	267	top	3,123	624
18	table	3,297	721	riding	2,858	1,854
19	riding	3,190	1,854	down	2,802	104
20	are	3,067	34	field	2,699	1,015

Table 4.13: The top 20 words from the RNN with and without context

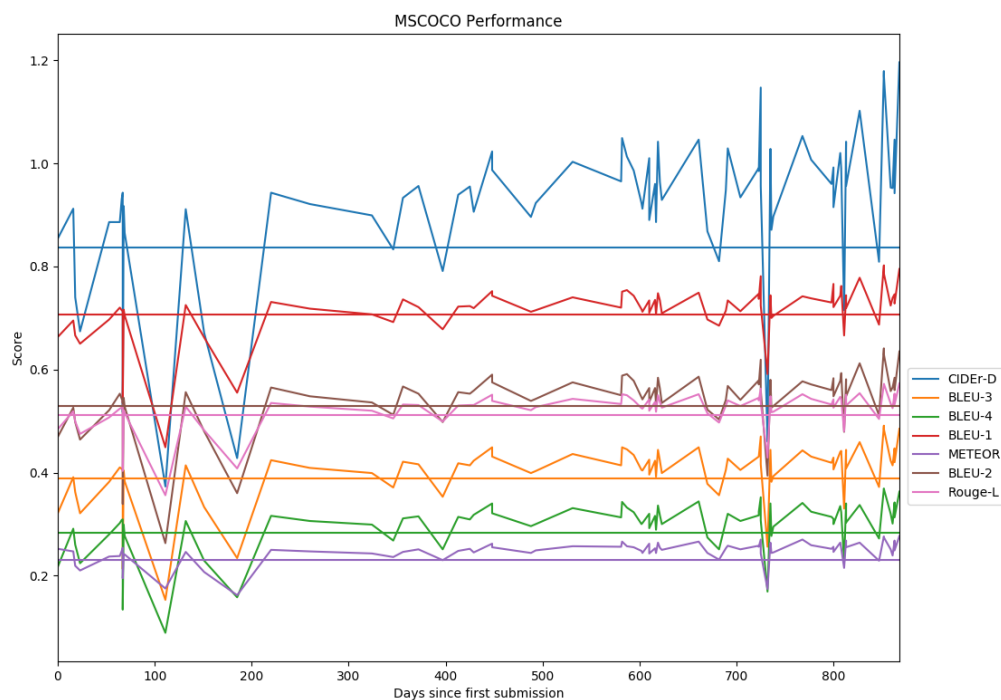


Figure 4.14: MSCOCO leaderboard results for the different metrics over time

the RNN without context produces an average sentence length of 10.04 words. These are closer to the average sentence length of the MSCOCO data of 10.61 words than the probabilistic model's output. Both RNNs are trained with a dictionary size of 10,000 words. Brysbaert et al. [19] estimated that a 20 year old American knows around 42,000 words on average, increasing to 48,200 for a 60 year old. This shows there is still a long way to go to match the number of words known by an adult. Hazenberg and Hulstun [62] claimed that a vocabulary size of around 11,000 is required to gain a coverage of 95% of texts, compared with a value of 3,000 to 5,000 that is often stated. This is close to achievable but is relying on the ability of readers to make guesses of the meaning of words based on existing knowledge [118]. The RNN without context uses 1,015 unique words whereas the RNN with context uses only 832 unique words but gains a better performance. This suggests that there may be very little benefit to using a dictionary size of 10,000 words or that it may reduce performance. The probabilistic language model of section 4.2 used a vocabulary of 860 unique words, but this had a

lower performance in general. Compared to the size of the training set (30,543 words), the vocabulary is quite small, but many of the words used are infrequent or have spelling mistakes. Using automatic spelling correction could be a way to improve performance by increasing the number of examples and reducing the number of words the system has to try to learn.

It is possible to compare the overlap of the vocabularies of the different methodologies. The RNN with context and without context have an overlap of 730 words. It is not surprising that the overlap is high, but it means that there are words that are unique to each RNN rather than the one being a subset of the other. The unique words are mostly very infrequent (under 100 occurrences) nouns, and some verbs. Compared with the probabilistic model's output, the vocabularies with and without context have an overlap of 602 words and 647 words respectively. The underlying CNN features in the convolutional and final layers used as input may be influencing which words are learnt and used by the RNN model.

4.11 Chapter Conclusions

It is possible to generate captions by using CNNs to generate individual words relating to objects and actions, and then combining them into full sentences with a greedy probabilistic language model. The size of the dictionary of nouns does not need to be that large to create a functioning system. This is likely to be a product of the biases in the dataset, though every day language will only use a small subset of the entire language. Nouns are very important as they often form the basis of the description of an image along with the verb. Prepositions are also important but are more difficult to simply predict. Even a simple language model with perfect input can score well but cannot compete with a more complex model such as an RNN.

An RNN can learn a complex language model using features extracted from an image, using a CNN as input. Using attention it can learn to look at different parts of an image to show how they relate to the different parts of a sentence. More general features from a CNN can also be used in conjunction, to change how the RNN learns. The different CNN features used have a big influence on the performance of the RNN.

The RNN design that uses input from GoogleNet and VGGNet by combining convolutional input and the final hidden layer representation can achieve performance at a similar level to the state-of-the-art by some metrics. The vocabulary used by the RNN output differs based on the CNN features used as inputs. Given that the context input to the RNN can influence the language used by the RNN, other sources of data and annotation could be used to make an RNN perform different types of language generation tasks.

When training an RNN with a vocabulary of 10,000 words only a much smaller portion is used in practice. Unsurprisingly, the frequency of words in the output closely matches that of the training set. In the case of the MSCOCO data, this varies a little from the word frequencies in English given the more narrow focus of the dataset. Despite training on the same data, the language used by different models trained on the same data still varies.

Chapter 5

Caption Generation for Broadcast Television

The caption that is created for an image can be influenced by contextual information for the image. For example, knowing the identities of the people in the image allows the caption to use their names. Building on the RNN design from chapter 4, additional information can be used to alter the way the RNN generates captions for images. Using an RNN in this way allows more specialised captions to be produced. This can be applied to generating captions for automatically labelling television shows with the help of additional context, provided by character and location identifiers. This combines work in the previous chapters of this thesis.

5.1 Broadcast Television Captioning

One application for captioning is automatically labelling footage from broadcast television. This is useful for both image or video retrieval, and for audio descriptions for the blind. It is a difficult task as the description needs to capture the specifics of the scene, such as the character names, and the location. Standard image captioning datasets such as MSCOCO focus on more generic scenes, with people usually labelled as “a man” or “a woman”. Using a network pre-trained on such datasets only gives results with surface level descriptions. Scripts tend to not contain enough detail in the

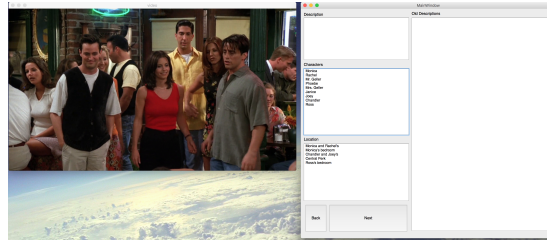


Figure 5.1: Screenshot of the ground truth labelling GUI tool created for labelling “Friends” with a description, the characters and the location

descriptions, with only a sentence or two to set the scene and sporadic stage descriptions. This means that a fully labelled dataset from broadcast television is required for training.

5.2 Dataset

The data used is taken from the 3rd season of “Friends”, which contains 25 episodes of between 20 and 25 minutes. Frames were labelled at a 1 second interval, with the title sequence, empty frames, cross-fading frames, and credit sequence frames being skipped. Each frame was labelled with a single sentence description using the names of the characters, and the location. In addition to this, they were labelled with the names of the characters, and the name of the location. The annotation was performed using a GUI tool created for this purpose. A screenshot of this tool is shown in figure 5.1. The data was split into 24 episodes of training data, and 1 episode left out for testing. Splitting the data in this manner assures that no frames from the same scenes are in both the training, and test set. It also creates a more realistic example of the RNN “watching” an unseen episode.

5.3 RNN training

To generate captions for the test episode, a pipeline based on the RNN with context input (section 4.7.3) was chosen based on its performance on the MSCOCO data. The

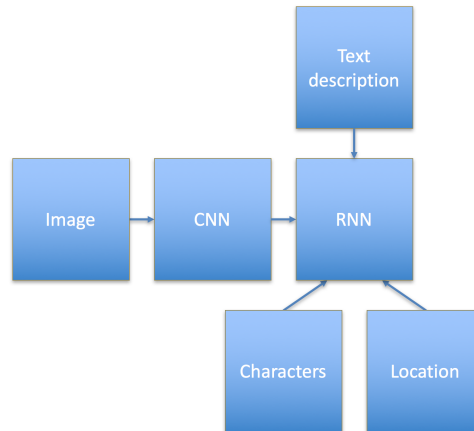


Figure 5.2: Pipeline for training the RNN for generating descriptions with character and location input

RNN design can accept the character and location information as additional inputs, as seen in figure 5.2. This allows the RNN to utilise external information about the locations, and characters when creating the description of a frame. The character and location information can come from character and location prediction methods such as those featured in this thesis. A set of all the characters and locations from the training set was created and then a binary vector was created for each frame, indicating if a particular character was present and the location. These binary vectors were used as the context input to the RNN for their respective frames. During the training stage, the ground truth characters, and locations were used. The training set available is smaller than MSCOCO with only a single caption available for each image.

5.4 Friends Location Recognition

In appendix A, the combination of scripts and television shows was used to perform location recognition. Some of these techniques can be applied to perform location recognition on the data taken from “Friends” to help the description use the correct location. To predict the location in frames from the test episode, Scale-Invariant Feature Transform (SIFT) descriptors were extracted from each frame in the training set, and

the test set. The SIFT descriptors are used to calculate a homography from each test frame to each frame in the training set, using Random Sample Consensus (RANSAC) to find the inliers. The test frame is assigned the location with the highest number of inliers. If multiple locations have an equal number of inliers, the prior probability is used to decide. Using this method to classify the locations in the unseen episode from season 3, an accuracy of 48.83% was achieved. Season 3 contains 68 different locations in total, with many episodes repeating locations such as the accommodation, and work places of the main characters.

For comparison, CNNs were also trained to classify the locations from the data in the training set. This was further split into a set of 75% for training and 25% for validation. The GoogleNet [158] design was used with multiple soft max loss layers. The GoogleNet design was chosen due to its very high performance on the ImageNet challenge. Its design allows for a very deep network without requiring multiple training stages where the depth is increased over time. It was trained to directly classify each of the frames from the unseen episode with a location. The output from the final layer of the network is used for the classification. The AlexNet [85] design was also used to show the difference in performance between these two models. Both models were trained with, and without random cropping.

Using this approach, an accuracy of 75.02% was achieved in with GoogleNet and no cropping. The results for the tests are shown in table 5.1. The CNNs outperform the homography-based matching method. This is because the method is capable of learning the features that identify the locations during its training process, and does not need to rely on generic features extracted by SIFT and the geometric constraints. A CNN can recognise the same location from more variable viewpoints. The homography-based method may only be able to cope with variations of previously seen viewpoints with a planar transformation. The performance margin between the lowest performing CNN and the homography-based matching is not as high as expected. When restricting the homography-based matching to only the locations featured in the episode, the performance increases to 59.42%. This is an easier task as it only needs to find the best match in a much smaller set of locations. On the training sets, the CNNs achieve close to 100% performance but on the test set, the best performance is only around

Location	Homography	AlexNet	AlexNet (crop)	GoogleNet	GoogleNet (crop)
Ross’s Bedroom	0.00	0.00	0.00	0.00	0.00
Monica & Rachel’s	70.58	97.29	97.29	96.93	96.93
Healing Hands	0.00	0.00	0.00	12.5	15
Monica’s bedroom	38.49	33.88	31.91	86.86	68.75
Ross’s	0.00	27.84	12.37	17.53	15.46
The Xerox place	6.52	0.00	1.09	3.26	0.00
Chandler & Joey’s	77.57	84.11	64.49	82.24	80.37
Central Perk	33.33	80.95	76.19	80.95	100.00
Total	48.83	62.73	59.34	75.02	70.65

Table 5.1: Results of the location classification on the “Friends” data from the SIFT inlier method and the different CNN varieties

75%. Unsurprisingly, the most commonly occurring locations achieve the highest performance using each of the methods. Interestingly, on the infrequent location of the “Xerox Place”, the homography-based approach gets the highest score, albeit a very low one. This is likely because the CNN approaches do not have enough training data for the location, whereas the homography-based method will work well for near duplicate images. In all cases, Ross’s bedroom fails to be correctly recognised, as this a very infrequent location in the series. It is seen from very different angles in the test episode compared with the training data. The cropping seems to decrease overall performance for the CNNs, although gives an increase to “Central Perk” and “Healing Hands” with GoogleNet. Only GoogleNet is capable of recognising images of “Healing Hands”, but AlexNet manages to get the best performance on Ross’s apartment. This drops drastically when using cropping. This is possibly because useful features for identifying the location are lost during the cropping process.

5.5 Character Identification

The character recognition from chapter 3 can be used to label the characters in the unseen episode, so that the RNN is able to correctly name the characters in the frame. Features were extracted from the frames in the training set using the face regression technique previously described 3.1. The character prediction pipeline is then used to predict the characters that are present in each frame. The best results for automatic character labelling were from the use of automatically harvested web data combined with training data from the test episode. Positive data is taken from the harvested web data and scenes containing only that character. Negative data is taken from other character's web data, and scenes from the videos without the character. This data is used to train a random forest, per character, to get a score for each character. As this method had the best performance in chapter 3, it was chosen for predicting the character labels as input into the RNN as context.

Both the SIFT-based features and OpenFace features were used, to show how they both perform on the data. When the random forests were trained, there was an imbalance of positive and negative training data for each character. More negative data was available, as there are more examples of the other characters. For each descriptor type, a random forest was trained with all the available negative training data. To prevent a bias in the system from unbalanced data, a random forest was also trained with a subset of the negative training data to balance the training set. The random forest classifiers were applied to each face detected in the test set. The results for the test episode are shown in table 5.2.

The results show a poor performance for labelling the characters in each frame of the test set. However, the percentage of correctly labelled detected faces is much higher. This is because the face detector has failed to detect the face in the frame. In some scenes, the character's face may not be present. There is very little difference between the unbalanced and balanced random forests. However, the less frequently appearing characters have a lower performance without balancing. The OpenFace features perform better than the SIFT-based features by a small margin on the subset of detected faces. The percentage of correctly labelled characters is higher with the

Character	Open Face	Open Face (unbalanced)	SIFT	SIFT (unbalanced)
Monica	39.61	35.71	19.81	19.48
Chloe	3.33	3.33	0.00	0.00
Gunther	54.54	54.54	36.36	0.00
Rachel	53.27	54.00	43.83	45.76
Jasmine	34.29	0.00	2.86	0.0
Phoebe	61.39	65.68	29.37	32.67
Isaac	16.67	2.78	2.78	0.0
Joey	48.82	49.83	27.61	28.28
Chandler	59.85	62.41	33.58	36.86
Ross	53.36	54.89	36.39	38.23
Characters	50.19	50.27	30.90	32.26
Faces	87.31	87.46	82.51	86.14

Table 5.2: Results of the character classification on the “Friends” data using random forests trained with SIFT and Open Face descriptors

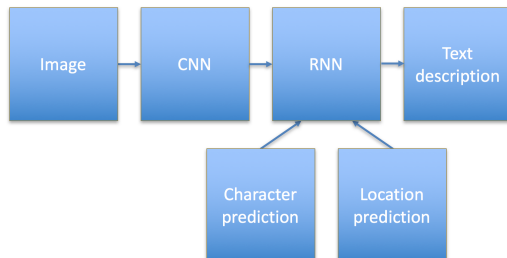


Figure 5.3: Diagram of the RNN pipeline used for generating captions based on character and location predictions

OpenFace descriptor as a greater proportion of the faces are detected. Compared with chapter 3, there are more training examples where a scene contains a single character. N.B. The tracking has not been applied to improve non-frontal faces as the frames in this dataset are taken at one second intervals.

5.6 Description Generation

To generate descriptions for frames from the unseen episode, the characters and location need to be predicted and inserted into the trained RNN. Figure 5.3 shows the pipeline used for description generation with the character and location input. As with the training step, the labels need to be turned into a binary vector that the RNN can accept. The character labels from the highest performing random forest for the SIFT and Open Face features were used to generate labels. Location predictions were taken from GoogleNet and the homography-based methods.

5.7 Results

To test the description generation on the unseen episode, the frames from the episode were passed through the GoogleNet CNN and the final convolutional layer from the final inception module was used as input to the RNN. To test the effectiveness of the location and character input to the RNN, different combinations of labels from ground

Location	Character	BLEU1	BLEU2	BLEU3	BLEU4	METEOR	ROUGE	CIDEr
GT	GT	0.462	0.369	0.305	0.254	0.223	0.473	1.585
SIFT	SIFT	0.438	0.342	0.275	0.225	0.209	0.443	1.343
GoogleNet	OpenFace	0.438	0.342	0.275	0.225	0.209	0.443	1.343

Table 5.3: Results of the caption generation pipeline with different character and location label inputs

truth and methods of character and location prediction were used. In contrast to the training set, the test set was annotated with three captions per image. This allows for more variance in the captions.

Table 5.3 shows the scores from the standard automatic captioning metrics for the captions generated by the RNN with the different sources of character and location labels used as context. These results were generated using convolutional input from GoogleNet, and a beam size of 5, that was found to be the best in chapter 4. The model was trained for 30 epochs as this gave the best results when using ground truth context input. As expected, using the ground truth location and character labels produced the best performance. The performance difference between ground truth and predicted context is not as large as expected. The ground truth input represents the best possible output for the method and the predicted input manages to approach this. Compared with the MSCOCO results, the scores are lower. Particularly for the BLEU1 and BLEU2 scores. Surprisingly, however, the CIDEr scores are much higher. This metric is based on consensus with reference sentences and with fewer reference sentences it is easier to achieve a consensus. The method should perform best on the most frequent characters and locations.

The training data is small compared to MSCOCO. Using more training data could be another way to improve the performance of the caption generation. With only 1 caption per image the training could be overfitting. The data itself is also more specialised so has less variation compared to MSCOCO.



Figure 5.4: Example frame from “Friends” with the automatically generated description “Phoebe is standing in Monica and Rachel’s apartment while Monica is standing behind her”

5.8 Friends Examples

Examples of frames from the test set with automatically generated captions are shown in figures 5.4, 5.5, 5.6 and 5.7. These captions were generated with the ground truth character and location input to the RNN. The examples show the captioning is working well across a number of different characters and locations. The correct character names and locations are being used in these examples.

When using predicted input, some mistakes can occur with the captioning. An example where the captioning has failed is shown in figure 5.8. The predicted location and characters are both incorrect, causing the caption to fail. The generated caption features the predicted location and character but does not match the image. The image does however, feature a car shaped object. The caption is one that is included in the training set, rather than a novel generated one.

Figure 5.9 shows another example where the incorrect identification of characters has caused an error in the caption. Joey has been correctly identified, whereas Chandler has been classified instead of Ross. This causes the generated description to use “Chandler”



Figure 5.5: Example frame from “Friends” with the automatically generated description “Ross is lying in his bed”

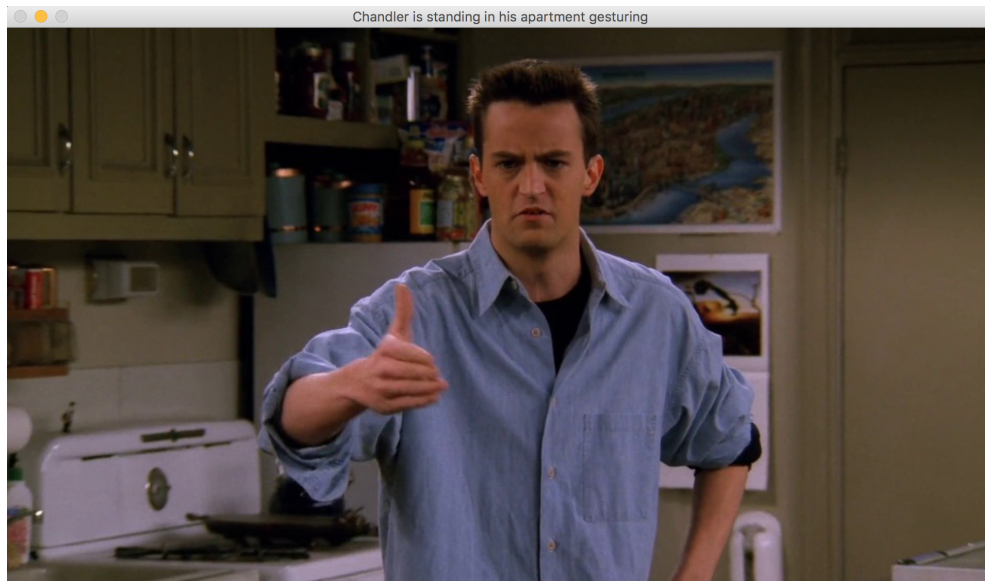


Figure 5.6: Example frame from “Friends” with the automatically generated description “Chandler is standing in his apartment gesturing”



Figure 5.7: Example frame from “Friends” with the automatically generated description “Rachel is sitting on the sofa in her apartment”

instead of “Ross”.

The example in figure 5.10 shows a situation where the caption fails to capture the detail of the scene. The position of the characters varies from the description and does not include their actions. Monica is using a blender, but this is not a situation that is in the training set. The captioning also lacks detail in the description, such as Rachel holding some objects.

5.9 Chapter Conclusions

In this chapter, the insertion of location and character data allowed an RNN to generate descriptions of content from broadcast television. Taking this contextual information from character and location prediction methods creates a complete captioning pipeline for unseen television episodes. The use of automatically labelled characters and locations was compared with ground truth input to the RNN. While using ground truth input gives the best results, it is still possible to generate captions based on automatically predicted input with only a relatively small drop in performance.

The location and character prediction both performed well on the test episode. The



Figure 5.8: Example frame from “Friends” with the automatically generated description “Chandler is climbing on a car shaped bed and pretending to drive in Monica’s bedroom”



Figure 5.9: Example frame from “Friends” with the automatically generated description “Chandler is standing next to Joey in their apartment”



Figure 5.10: Example frame from “Friends” with the automatically generated description “Rachel is standing opposite Monica in their kitchen”

CNN-based approaches scored better than the SIFT-based homography method, but the difference is not as large as expected. The SIFT-based method performs marginally better for one of the locations that has a very small training set. The homography-based matching works better for near duplicate detection, whereas a CNN is capable of learning features that can distinguish the different locations. However, this requires an appropriate amount of training data to perform well. All of the methods perform well on the most frequent locations.

The character identification using random forests trained on a combination of data from the web, and the training set performs well, achieving up to 87.46% accuracy. Failure to detect non-frontal faces and reliance on visible faces causes the performance to drop to around 50% for character labelling. Using non-facial recognition techniques and non-frontal face methods could enhance the performance of the character identification.

The use of context as an additional input to an RNN allows for flexible usage to apply the RNN to different situations. Using character and location predictions as additional inputs allows for the generation of natural language descriptions of data from unseen television footage. Contextual input to an RNN could also be used in a variety of other situations to allow the generation of sequences with specialised applications.

Chapter 6

Caption Generation with Emotion

Emotion is a powerful force and constitutes an important part of language. An image can be described not only by what it contains, but also how it makes the viewer feel. This represents a different level of understanding of an image. Language and emotion are heavily intertwined. One way that emotion can be represented in language is through adjectives. Adjectives are descriptive words that provide a greater depth of meaning to sentences. Combining computers and emotions is known as “affective computing” [131]. This field covers a variety of topics including automatically recognising emotional expressions in faces, and using computers to convey emotion. Conveying emotion can be through tone of voice or through the use of language. Automatically generating language that expresses emotion is a challenging task. It requires not only learning a link between the image and the language, but with emotion as well. A realistic emotional AI could help bridge the gap for human and AI interaction by allowing for empathy.

6.1 Representation of Emotion

Before the emotions can be learnt, a representation is required. Ortony and Turner [124] presented a comparison of work that aimed to define the set of basic human emo-

tions. Basic emotions are the primitive building blocks from which other emotions are comprised. Many of the proposed sets of basic emotions overlap heavily. The basis for deciding what constitutes a set of basic emotions varies between publications. Tomkins [164] used the density of neural firing to define a set of basic emotions. Plutchik [132] presented a psychoevolutionary theory of emotions, using Darwinian evolution as a justification. Plutchik’s wheel of emotions, shown in figure 6.1 described eight basic emotions, how they vary in intensity and their opposites. Each emotion has 3 variants, this gives 24 emotions that can be used to describe the contents of an image. Tomkin’s model included the same emotions included in Plutchik’s wheel but also included contempt and shame. The wheel was chosen to represent the emotions for caption generation as it provides a method to represent them as a vector. It is also used in the Visual Sentiment Ontology (VSO) dataset. The VSO is a dataset containing images taken from Flickr that contain content that is associated with emotion. The MSCOCO dataset is not labelled with emotions, but the language used in the captions contains sentiment and emotional content.

6.2 Adjective Noun Pairs

Adjectives and nouns are combined to create Adjective Noun Pairs (ANPs) that describe an entity and its properties. Humans will have a different emotional response to different ANPs depending on how it affects them. The captions in MSCOCO contain adjective noun pairs. The pairs are found by utilising the parts of speech tags extracted previously, and analysing them for any nouns that follow an adjective. The VSO dataset contains a mapping between ANPs and emotions. The images associated with an ANP can then be described using the emotions from Plutchik’s wheel of emotions. The distribution of emotions for each ANP were used to create training labels associated with the images in the dataset.

For each image in the MSCOCO dataset that contains an ANP in its caption, a 24-dimensional vector is created showing the intensity of each emotion from the wheel. The vector can also be used as part of the input to an RNN so that the emotional context is used to influence the predicted sentence.

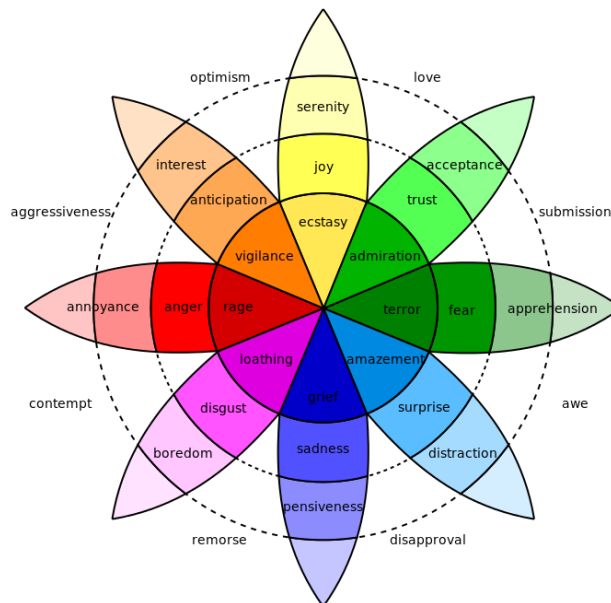


Figure 6.1: Plutchik’s wheel of emotions, which shows the main human emotions and how they are related

6.3 Emotion RNN

The 24-dimensional vector that is generated for each ANP is associated with the images used in training the RNN. This is done by matching ANPs in the MSCOCO captions and then labelling the image with the corresponding vector. The RNN (Section 4.7.3) is trained with GoogleNet input as the convolutional input. In contrast, the emotional vector is used as the context input rather than the final hidden layer of a CNN. At test time, to alter the emotional context of the RNN, vectors with different values for emotions are used as the context input. This allows the captions to be influenced by the emotion that is encoded in the context vector.

6.4 Results

The results of the RNN trained with emotional input on the whole MSCOCO validation set are shown in table 6.1. No beam search was used. For this test, the ground truth emotion data was used to test that the caption generation quality is still comparable

Dataset	BLEU_1	BLEU_2	BLEU_3	BLEU_4	METEOR	ROUGE_L	CIDEr
Validation	0.665	0.488	0.347	0.246	0.233	0.499	0.781

Table 6.1: Results with using the emotion RNN and ground truth emotional context

with the previous results. For images with no ANP, a vector with an equal distribution across all emotions was used. The performance is similar to the results of previous experiments. This shows that the natural language generation performance of the RNN has not been compromised by the addition of the emotional context.

The result of using the emotion RNN where the emotional context vector is artificially set to each of the 24 different emotions is shown in table 6.2. This is to force the RNN to generate captions that express a particular emotion. These results are on the subset of the validation set with sentences that contain ANPs. The performance is lowered for every emotion, although by differing amounts. This is not unexpected, as the forced emotion will not necessarily agree with the reference captions for a particular image. For example, a caption generated with disgust would differ from the reference captions for an image that people would generally consider pleasing. Different emotions and ANPs will be reflected in different proportions in the dataset so emotions that are less frequent in the dataset may give a lower performance than more frequent emotions.

Table 6.3 shows the number of different ANPs present in the captions generated by the RNN. Some emotions such as ecstasy, generate almost no ANPs. It appears that the more ANPs that are present in the output, the lower the CIDEr score for that emotion. This is likely because inputting that emotion has caused it to use more ANPs related to that emotion and is therefore less related to the reference captions for those images.

The most frequent emotion (calculated from the ANPs present in the output captions) is shown in table 6.4. For the emotions joy, sadness, disgust, amazement, and serenity, the most frequent emotion is the same as the input. For other emotions, the prevailing emotional output is sadness. Some of the most common examples of ANPs in the MSCOCO dataset are “little girl” and “little boy”. These are associated with sadness in the VSO emotion data and therefore, when using the ground truth emotional input,

Emotion	BLEU_1	BLEU_2	BLEU_3	BLEU_4	METEOR	ROUGE_L	CIDEr
Rage	0.565	0.381	0.243	0.154	0.183	0.433	0.496
Acceptance	0.618	0.433	0.297	0.204	0.209	0.466	0.678
Admiration	0.609	0.425	0.287	0.194	0.202	0.461	0.638
Amazement	0.621	0.432	0.291	0.197	0.204	0.461	0.648
Anger	0.561	0.378	0.239	0.150	0.183	0.430	0.490
Annoyance	0.540	0.354	0.224	0.143	0.174	0.410	0.430
Anticipation	0.596	0.414	0.281	0.191	0.201	0.455	0.618
Apprehension	0.593	0.405	0.269	0.178	0.196	0.445	0.593
Boredom	0.580	0.398	0.263	0.175	0.193	0.442	0.561
Disgust	0.554	0.374	0.248	0.165	0.185	0.428	0.488
Distraction	0.627	0.443	0.300	0.203	0.211	0.468	0.670
Ecstasy	0.589	0.412	0.278	0.186	0.198	0.456	0.605
Fear	0.593	0.407	0.267	0.177	0.195	0.451	0.587
Grief	0.611	0.429	0.295	0.202	0.212	0.462	0.658
Interest	0.449	0.286	0.164	0.093	0.146	0.379	0.260
Joy	0.517	0.346	0.220	0.139	0.180	0.411	0.414
Loathing	0.614	0.429	0.290	0.195	0.206	0.459	0.651
Pensiveness	0.471	0.309	0.185	0.110	0.172	0.401	0.349
Sadness	0.557	0.363	0.230	0.142	0.177	0.409	0.451
Serenity	0.579	0.393	0.258	0.171	0.186	0.438	0.530
Surprise	0.384	0.230	0.124	0.067	0.139	0.344	0.184
Terror	0.587	0.404	0.266	0.176	0.195	0.449	0.571
Trust	0.627	0.434	0.294	0.200	0.207	0.466	0.681
Vigilance	0.602	0.413	0.277	0.186	0.197	0.452	0.617
Ground Truth	0.681	0.508	0.366	0.258	0.242	0.507	0.797

Table 6.2: Results of the emotion RNN with synthetic input on the subset of MSCOCO that contains ANPs

emotion	count
ecstasy	2
anger	1,483
fear	984
trust	40
loathing	379
annoyance	2,028
anticipation	211
acceptance	44
pensiveness	117
interest	4,970
terror	829
vigilance	3
disgust	484
joy	556
grief	245
boredom	984
surprise	913
amazement	226
distraction	941
admiration	4
rage	158
apprehension	557
sadness	4,255
serenity	246
ground truth	1,093

Table 6.3: Count of ANPs in the output for different emotions

the most common emotion is sadness. “Person” is a category in the MSCOCO data and there are many images of children in the data. This leads to sadness being a common output from the system.

While the most frequent emotion in each output is important, the correlation of the output caption with the input emotion allows the influence of that emotion to be evaluated. Each ANP expresses a range of emotions, shown by a distribution over the 24-dimensional vector. The Pearson correlation coefficients are shown in table 6.5. The highest correlation values are where the input and output emotions match (shown in table 6.4). This is not surprising, as the most common emotion will have a greater influence and correlate with itself. However, other emotions that have a different predominant emotion in the output also show a correlation with the input emotion. This shows that there is still an effect on the emotion of the output text, even if it is not the most common output emotion. Some emotions, and the ANPs associated with them, will be more related to the visual content of images. Emotions such as disgust, serenity and joy can more easily be depicted in visual form than loathing and acceptance. This matches with their correlation coefficients.

Given that Plutchik’s wheel of emotions is configured in a way that gives each emotion an opposite, the correlation between each emotion and its opposite on the wheel could also be relevant. These results are shown in table 6.6. Unfortunately, in most cases, there does not seem to be a negative correlation between the emotion and its opposite. This is not something that is explicitly trained for.

Another feature of Plutchik’s wheel of emotions is the grouping of emotions, allowing each emotion to be represented a magnitude of its parent emotion. This allows the 24-dimensional vector to be recalculated as an 8-dimensional vector. Treating the different levels of emotions as different weightings the 24-dimensional vector to be collapsed to an 8-dimensional vector. Performing this allows a new correlation value to be calculated. This is shown in table 6.7. This leads to an increase in the correlation for the 24 emotions.

input	top output
ecstasy	anticipation
anger	sadness
fear	sadness
trust	sadness
loathing	sadness
annoyance	sadness
anticipation	sadness
acceptance	joy
pensiveness	sadness
interest	sadness
terror	sadness
vigilance	anticipation
disgust	disgust
joy	joy
grief	sadness
boredom	sadness
surprise	sadness
amazement	amazement
distraction	sadness
admiration	anticipation
rage	sadness
apprehension	sadness
sadness	sadness
serenity	serenity
ground truth	sadness

Table 6.4: The strongest emotion calculated from the ANPs for each input emotion.

emotion	correlation coefficient
ecstasy	-0.117
anger	0.339
fear	0.518
trust	-0.138
loathing	-0.205
annoyance	0.216
anticipation	-0.062
acceptance	-0.128
pensiveness	0.150
interest	0.426
terror	0.322
vigilance	-0.147
disgust	0.927
joy	0.656
grief	-0.035
boredom	0.350
surprise	0.286
amazement	0.490
distraction	-0.088
admiration	-0.158
rage	0.246
apprehension	-0.149
sadness	0.709
serenity	0.864

Table 6.5: Correlation coefficient between the output ANPs and the reference emotion

Emotion	Opposite	Correlation
ecstasy	grief	-0.108
anger	fear	-0.020
fear	anger	-0.015
trust	disgust	0.048
loathing	admiration	-0.165
annoyance	apprehension	-0.187
anticipation	surprise	0.193
acceptance	boredom	0.085
pensiveness	serenity	-0.056
interest	distraction	-0.071
terror	rage	-0.086
vigilance	amazement	0.187
disgust	trust	-0.092
joy	sadness	0.368
grief	ecstasy	-0.174
boredom	acceptance	-0.201
surprise	anticipation	-0.094
amazement	vigilance	-0.265
distraction	interest	0.225
admiration	loathing	-0.214
rage	terror	-0.042
apprehension	annoyance	0.148
sadness	joy	0.235
serenity	pensiveness	-0.027

Table 6.6: Emotion input to RNN and correlation of output with the opposite emotion

Emotion	Correlation
ecstasy	0.116
anger	0.532
fear	0.793
trust	-0.432
loathing	-0.130
annoyance	0.059
anticipation	-0.312
acceptance	-0.285
pensiveness	0.396
interest	-0.362
terror	0.795
vigilance	0.847
disgust	0.939
joy	0.361
grief	0.469
boredom	-0.145
surprise	0.740
amazement	0.848
distraction	0.699
admiration	-0.604
rage	0.708
apprehension	-0.220
sadness	0.693
serenity	0.652

Table 6.7: Correlation between input emotion and reference emotion when the emotions are condensed to different intensities of the 8 main emotions

	Generated Emotion	Generated	Human
Most	38.39	9.52	52.08
Second	41.96	27.01	31.03
Least	19.64	63.47	16.88

Table 6.8: Average distribution of responses for the subjective emotional content survey

6.5 Subjective Evaluation

Although the generated sentences can be evaluated quantitatively, subjective qualitative assessment is also important. Humans are better at judging the emotional content of a sentence, due to the subtleties and complexities of language and emotion. The emotions represented in Plutchik’s wheel of emotion are designed to reflect the emotions that can be felt by people. To get human feedback on the emotional content in the generated sentences, a survey was designed to compare these sentences with the human written ground truth, and sentences generated without emotion. For each question, the user had to rank the three sentences from most emotive, to least emotive. The human captions were picked at random from the five ground truth captions available for each image in the MSCOCO dataset. The best performing model from chapter 4 was used to generate the “standard generated captions”. There were four questions for each emotion, meaning a total of 96 questions in the survey. The order of the questions and answers were randomised to try and make the survey as fair as possible.

The survey was taken by 14 people to capture a variety of different opinions. All of the participants were fluent English speakers with an age range from 26 to 82, and originating from 3 different countries. There were 10 male and 4 females in the test group. Given the number of questions, the survey took participants between 30 and 60 minutes to complete.

The results for each question and each individual were combined to show the distribution of responses for each of the sentence sources. This is shown in table 6.8. The results show that the generated emotional caption contained a lot more emotional information compared to the default generated caption, which was very rarely chosen as

the most emotive. The human caption is unsurprisingly the most frequently chosen as the most emotive caption. The generated emotion caption was the most frequently chosen in second place, this is because it was often chosen as the second choice behind the human caption. Both the human caption and generated caption with emotion were rarely chosen as the least emotional option, whereas the default caption was the most frequently chosen as the least emotive. The emotional caption was chosen over the standard caption 74.85% of the time. This shows that the emotional caption was considered to have more emotion by a considerable margin. However, when compared with the human caption it was chosen 43.75% of the time. Although the human captions were not necessarily written with emotional descriptions in mind, the size of the gap is still relatively small.

Given the number of participants and questions, it is possible to find a confidence interval for the results. The responses are a binomial distribution of success and failure of the generated caption to beat the default generated caption or beat the human caption. If the captions were considered equally emotive, the probability of either being picked should be 50%. The p-value is calculated to be $p < .000001$. This shows a very high probability that the emotional captions were not selected by chance.

Another factor that can be considered is the consistency of the responses between different participants. This gives a way to demonstrate the reliability of the responses and agreement between them. Entropy can be used as a way to measure how ordered and disordered the results are. If the results are more evenly distributed due to people giving different answers, then there is a high entropy. If people generally answer the same way then there will be a lower entropy. Calculating the entropy of the distribution of answers for each question gives 301.13 bits of entropy. Given an equal distribution of answers there would be 456.47 bits of entropy. The empirical entropy is significantly less than the maximum, showing that the answers given are much more ordered than randomly distributed.

It is also possible to split the results into the 24 different emotions, to test the effectiveness of the emotional caption generation at each of the emotions. There were only four examples for each emotion so results from the survey will be less consistent than

Emotion	Better than standard caption (%)	Better than human (%)
ecstasy	19.64	3.57
anger	60.71	55.35
fear	87.50	23.21
trust	87.50	37.50
loathing	53.57	21.34
annoyance	82.14	48.21
anticipation	80.36	57.14
acceptance	96.43	41.07
pensiveness	82.14	53.57
interest	89.29	30.36
terror	71.43	44.64
vigilance	57.14	25.0
disgust	76.79	57.14
joy	94.64	75.00
grief	44.64	23.21
boredom	71.43	19.64
surprise	96.43	64.29
amazement	80.36	44.64
distracted	85.71	30.35
admiration	62.50	51.79
rage	60.71	32.14
apprehension	85.71	71.43
sadness	85.71	62.50
serenity	83.93	55.36

Table 6.9: Per emotion results showing percentage of responses that rate it better than standard or better than human

for the overall result. The per emotion results are in table 6.9. The table shows the percentage of captions that were rated better than the standard generated caption and the human written caption. The results show that there is a great deal of variation between the different emotions, reaching 96.43% for surprise and acceptance and falling to 19.64% for ecstasy. The performance drops when comparing with the human captions, however.

The per emotion results can be compared with the results from the analysis of the output ANPs. The correlation coefficient in table 6.5 does not appear to reflect the level of performance shown in table 6.9. The captions may still be more emotive than the standard captions, even if the correlation with the input emotion is not strong. Looking at table 6.3, the emotion ecstasy showed very few ANPs in the output captions and shows a very low performance in the subjective results. However, other emotions that showed very few ANPs in the output captions such as admiration and vigilance, show a lower than average performance, but not as low as the number of ANPs suggest. Visual inspection of the captions shows that they vary in more than just the ANPs used. When compared with human captions, joy gave the best results. This could be because the emotional captions tend to use phrases talking about colourful flowers, young children and similar concepts, that could cause a strong response in participants.

6.6 Qualitative Examples

An interesting example of the performance of the emotional caption generation is shown in figure 6.2. The caption generated with emotion was overwhelmingly chosen as the most emotive caption. The emotional caption was “A view of a beautiful night time in a city” whereas the generated caption was “A large clock tower towering over a city street”, and the example human caption was “A bridge over water in front of a castle with a clock”. The emotional caption is very different and describes the beauty of the scene, rather than providing a plain description of the scene. The caption was generated to portray pensiveness. Most other captions generated with emotion are similar to the generated caption and vary in the use of adjectives and other descriptive language.

Figure 6.3 shows another image which was used to generate captions. With the emotion

anger, the generated caption was “A red truck is driving down a busy street”, whereas the default generated caption is “A double decker bus driving down a street”, and the human written ground truth was “A large truck on the street in front of a music store”. The caption generated with emotion is more similar to the ground truth, and uses more descriptive words to describe the colour of the truck. It also uses the ANP, “busy street”, which is associated with anger in the VSO data.

The image in figure 6.4 was used to generate a caption to show disgust. The caption generated with disgust is “A cow standing in a dirty water next to a dirty water”, the standard generated caption is “A dog standing in the grass near a river”, and the ground truth is “Two cows standing in a lake with several ducks around”. While the generated caption contains repetition, it uses an ANP to describe the river as “dirty” rather to show disgust. Interestingly, the standard generated caption is accurate in this case also.

The captioning system can describe an image accurately but in a different way when given an emotion. An example of this is in figure 6.5. When generating a caption with fear the output is “A young girl is sitting in a dark room.”, without emotion “A young boy is cutting a cake on a table”, and the human caption was “A little girl uses scissors to cut up bits of yellow paper”. Both the ground truth and fear captions correctly describe the image but they both use different language and focus on different elements. The fear caption focuses on the dark room, which is an ANP associated with fear.

It is interesting to see how the same image is described differently when different emotions are used. Figure 6.6 is given the caption “A group of people standing on a beach next to a large building” when using amazement as the emotion, and “A group of people standing in front of a calm ocean” when using serenity. The amazement caption chooses to focus on the large building in the background, whereas the serenity caption focuses on the calm ocean.



Figure 6.2: Image of the houses of parliament with a very different caption generated when using emotional context



Figure 6.3: Image of a truck on a street used to generate a caption showing anger

6.7 Failure Cases

There are also cases where the emotional influence causes the captioning to fail. This may be due to the emotion taking priority over the image input. This causes the caption to contain ANPs that do not occur in the input images. An example of this is in figure 6.7, where the caption showing sadness is “a little girl is eating a zebra on the ground”. As previously mentioned, one of the most common ANPs for sadness is “little girl”. In this case the caption uses the ANP even though it is not relevant to the image of the zebra. The caption correctly mention the zebras, but caption fails to describe the image accurately.

Another example is in figure 6.8. The caption generated for this image using the



Figure 6.4: Image of cows in a river used to generate a caption showing disgust

“interest” emotion is “a busy street with a busy street with a busy street”. Although the caption has correctly identified the busy street in the image. The interest emotion has caused the caption to focus solely on the busy street and ignore the man performing a skateboard trick in the foreground.

6.8 Emotional Captioning for Broadcast Television

Although the emotional captioning system was trained on the MSCOCO dataset, it is also possible to evaluate its output on other datasets. The “Friends” dataset from chapter 5 was used as an input to the emotional RNN, to generate captions that express different emotions. Captions were created for each of the 24 emotions. Figure 6.9 shows a frame which was described as “A man and a woman standing in a dark room” using the emotion fear. This focuses on the dark room but is not capable of using character names, as it has no knowledge of them. Using the disgust emotion, a caption was created for Figure 6.10. The generated caption was “A man is sitting on a bed in front of a dirty bathroom”, which focuses on the bathroom in the image and uses the word “dirty”. The frame in figure 6.11 was used to create a pensive caption, “A woman and a woman standing in front of a woman smiling.” This caption focuses on the expression of the character in the foreground, rather than the actions and objects.



Figure 6.5: Image of girl which was used to generate a caption showing fear

6.9 Chapter Conclusions

Another use for contextual input to an RNN was investigated by using emotional data as an input. This was input in the form of a 24-dimensional vector, based on the emotions defined by Plutchik's wheel of emotions. This provides a numerical value to the complex emotions that can be experienced by a person. Adjective Noun Pairs (ANPs) are parts of language that are more emotional and descriptive. Using this as an additional RNN input allows the language generated by the RNN to use more emotional words. While it this did not increase the captioning performance on the MSCOCO dataset, the baseline captioning performance was not greatly reduced.

Analysis of the output ANPs showed a correlation between the emotion of the output ANPs and the input emotion for several of the emotions described by Plutchik. The correlation was higher for emotions that are more strongly linked to visual content such



Figure 6.6: Image of a beach which is described differently when different emotions are used



Figure 6.7: Image of a zebra in the MSCOCO dataset where the captioning fails when the emotion is set to sadness

as disgust and joy. Emotions that are harder to display in an image such as acceptance and admiration show a lower correlation coefficient.

The human testing shows that the emotionally generated captions are considered more emotive than the standard generated captions, and even compare well with the original human captions in many cases. Even for cases where the correlation with the input emotion was not high, the emotionally generated captions still perform well against the captions generated without emotion. The output captions can show more variation than just using relevant ANPs.



Figure 6.8: Image of a man skateboarding on a street where the generated caption fails using the interest emotion

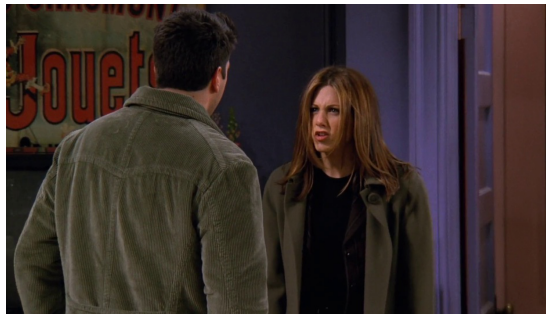


Figure 6.9: A frame from “Friends”, which was used to generate a caption showing fear



Figure 6.10: A frame from “Friends”, which was used to create a caption expressing disgust



Figure 6.11: A frame from Friends, which was used to generate a pensive caption

Chapter 7

Discussion

7.1 Discussion

This thesis has explored the combination of images and video with textual annotation, using machine learning. This has been applied to learning character identities in broadcast video, as well as learning to automatically describe images. This work was combined to create a method for generating descriptions for broadcast media that takes contextual input from character, and location predictions. In addition, it includes foundational work on generating natural language that can express emotions.

Chapter 3 focused on the automatic naming of characters. “Friends” was chosen as it contains a large cast of recurring characters in a real world setting. Many locations are common to all episodes, such as the character’s apartments, work places, and their local coffee shop. These properties also make the series useful for location recognition, and description generation, as the repeated locations can be learnt. This means description generation techniques that are trained on a number of episodes will still be applicable to unseen episodes.

A method for regressing faces and creating face descriptors was described. These face descriptors are created from the concatenation of Scale-Invariant Feature Transform (SIFT) features from the facial landmarks. The features are very high-dimensional. Faces were detected in episodes of “Friends”, and then facial landmarks were regressed.

A state-of-the-art level feature extractor, OpenFace [3], was used to provide a comparison for the performance of the techniques. This descriptor was chosen for its performance and its ability to represent faces in a way that is suitable for clustering and classification.

To attempt to separate characters in an unsupervised way, facial descriptors were clustered. Character identities were assigned based on the size of the different clusters, and the amount of screen time. However, this process not only clusters based on character but also on expression and lighting. A performance of 29% of characters identified based on the exemplars was achieved using the SIFT-based descriptor, and 31% using the OpenFace representation. The OpenFace descriptor created clusters with a greater character purity. This is likely to be because the OpenFace feature space is better at separating between people and less susceptible to pose or lighting. Although the characters were clustered, the method of applying labels to the clusters using the script was not accurate and a better labelling system was required.

To try and achieve this, a Gaussian Mixture Model (GMM) was employed but given the high dimensionality of the SIFT face features, work was needed to reduce the dimensionality. A random forest was trained to classify different people from an unrelated dataset. The output of the forest was used to create a new feature that represented the faces as a probability distribution over the classes in the training set. This aimed to transform the unseen SIFT-based features into this new, lower-dimensional feature space. The lower-dimensional features were then used to model a distribution of the faces for the scenes containing a known group of characters. Rules were automatically created to subtract sets of characters, so that individual characters could be isolated using the probability distributions.

With this method, an average performance of 63% was achieved with the random forest output descriptor, and 85% using the OpenFace descriptor. This was a significant improvement over the previous clustering method and is a more direct way to identify the characters. However, confusion between characters of opposite genders was higher than might be expected. Using the OpenFace descriptor led to a much higher performance and greatly reduced the confusion between the characters. Confusion between genders

was also lower. The representation from the OpenFace embedding allowed for a better separation of characters than the random forest representation. Performance was still not at a level that would be suitable for generating captions.

Given the amount of external data that is available on the web, the approach was adapted to exploit this data. Character names were automatically matched with their actors and the web data was harvested. In many cases, the data was sourced from photos of their personal life, promotional photo shoots or other acting roles. Random forests were trained on this actor data, with negative training data obtained from other scenes within the episode. This provided increased performance at the cost of requiring additional data. Using the SIFT-based descriptor with this method gave a character average performance of 83%. This was better than the GMM approach with the SIFT-based descriptor, but lower than using the GMM approach with the OpenFace descriptor. As with the GMM approach, there was still confusion between the characters with similar hairstyles. Confusion between characters of opposite genders still remained, with one character being classified as another male character more frequently than as herself. This shows that the features are not inherently good at encoding information that can discriminate between genders. Using this same method with the OpenFace descriptor gave an overall performance increase of 96.82%. This is a very high level of performance, with only a small room for improvement, and little confusion between characters and shows the effectiveness of the OpenFace representation at discriminating between people.

Chapter 4 examined the problem of finding a method for automatically generating captions. The Microsoft Common Objects in Context (MSCOCO) dataset was chosen for testing due to its extensive size, number of classes and its evaluation framework. The first technique in this chapter aimed to identify the nouns and verbs present in an image, and then create a sentence using them. Convolutional Neural Networks (CNNs) were trained to classify a single class from a subset of classes in ImageNet. To create a method for identifying multiple nouns and verbs in a single image, nouns and verbs were extracted from the captions and used to create sparse training labels. A CNN was trained to regress the noun training labels so that multiple nouns could be predicted for an input image. A CNN was then trained to generate verbs that takes the noun

probabilities as input to provide additional context for the verbs.

These CNNs were able to predict relevant words for a given input image, but a language model was needed to create a complete caption. To generate complete sentences, a probabilistic model of bigrams and trigrams was created from the training captions. Combined with the predicted probability of words from the CNNs, a method for generating captions was created. This method could create complete sentences but struggled with more complicated sentences and grammar. The performance of this captioning was below the level of the state-of-the-art techniques, which were using Recurrent Neural Networks (RNNs).

The use of RNNs allows for the generation of sentences directly from the output of a CNN. The CNN performs the task of image recognition and then the RNN can perform the task of natural language generation. The RNN design took intermediate layers from a CNN and gave attention to different parts of the image for different words. The RNN design was modified to allow it to also take in the final hidden layers from a CNN to provide context for the whole image. This altered the rate of training a but provided a slight improvement to the performance of the caption generation, compared to just using convolutional features from the same network. Performance achieved through this was at a similar level to that of the state-of-the-art. The design also allowed for more flexibility by allowing contextual input to the RNN, and demonstrated how the context input can be used to influence the output language. This allowed for the creation of a caption generation system that used additional inputs from character identification and location recognition techniques to improve the captioning performance.

RNNs are very effective at capturing the rules of natural language, due to their ability to change the data flow based on a memory of the previous states of the system. Combined with the high level of image recognition performance that can be achieved with a CNN, excellent captioning performance can be achieved. The complexities of natural language are too difficult to capture with a much more simple n-gram-based probability model. They cannot take into account language rules that occur over longer distances between words. Combining this with visual attention created a system that functioned in a way that is similar to how language is formed in the human brain.

Chapter 5 explored how the contextual input to an RNN could be used to adapt the RNN for use in different captioning scenarios. The first of these was the use of character and location input to use the RNN for generating captions to describe content from broadcast television. The television show “Friends” was used, as in chapter 3. The third season of the show, consisting of 25 episodes was annotated with the characters, location and a text description. This data was divided into a training set of 24 episodes and a test set of one unseen episode. Dividing the data on an episode level prevents data from the test scenes appearing in the data used for training the system.

To create a complete method for captioning broadcast media, the character identification and location recognition also needed to be performed. Multiple location recognition systems were created, utilising different CNN designs and a location matching technique from appendix A. The best performing technique used the GoogleNet [158] CNN, with a 75% accuracy. The AlexNet [85]-based design achieved 63%, and the lowest performance was with the SIFT-based homography matching technique at 49%, although the difference between the performance of AlexNet and the homography-based matching was only around 10%. The training data was relatively small compared to the size of data that is often used for training CNNs, and the homography-based system should work well for near duplicate match detection.

To generate character labels for use with the RNN, the best performing character identification technique from chapter 3 was applied. This was the random forest-based classification using automatically harvested web data. This was tested with both the SIFT-based facial regressor features and the OpenFace representation. Both descriptor types had a high performance on the faces that were detected (around 85%). However, the percentage of characters correctly labelled was quite different. Using OpenFace descriptors, the technique had a character-based performance of around 50%, and the SIFT-based regressor had a performance of around 30%. This difference is due to the number faces correctly detected. The OpenFace detector is much more effective, but many of the frames in the data do not feature frontal faces, causing a low overall performance. This performance could be increased through the use of techniques that work from a greater number of angles and by using techniques to identify individuals without visible faces.

The RNN was trained on 24 episodes of “Friends” data in the training set, with the location and character labels supplied as binary vectors indicating the location and characters present for each frame. At test time, the performance was tested using both ground truth and predicted labels. With the ground truth input, a Consensus-based Image Description Evaluation (CIDEr) score of 1.585 was attained. Using the predicted labels caused only a relatively small drop in the performance of the captioning system with a CIDEr score of 1.343. The performance metrics were lower than that achieved on the MSCOCO data, but the dataset is much smaller in size, with only a single text caption per frame in the training set. The CIDEr metric was much higher on this data, exceeding that of state-of-the-art techniques on the MSCOCO data. This demonstrated a complete broadcast television captioning system that combines character identification and location recognition techniques with an RNN.

In chapter 6, an additional use of contextual input to the RNN was examined. By training an RNN to understand emotion, and manipulating the emotion used to generate new captions. The emotional content in images and captions was encoded through the use of the Visual Sentiment Ontology (VSO) and Plutchik’s wheel of emotion. The VSO was used to create a mapping from Adjective Noun Pair (ANP) to Plutchik’s wheel of emotion, so that emotional features could be created. These emotional features were used as an input into the RNN to attempt to change the emotional content of generated captions. A correlation was found between the selected emotion and the output vocabulary.

The emotionally generated captions were compared with captions generated without emotion, and ground truth human captions using a survey. The survey results showed the emotionally generated captions were considered more emotive 75% of the time when compared with the captions generated without emotion, and about equal with human written captions. This is a very good level of performance and shows a great improvement over captions generated without any emotional influence.

Qualitative assessment of the emotional captions showed that the language used by the RNN reflects the input emotion. This can lead to a sentence that varies greatly from the captions generated without emotion, and the ground truth. The captions can focus

on different things in the image, which are more closely associated with the chosen emotion. Understanding emotion in images is something very new for a captioning system, and something that could lead to interesting breakthroughs in the Turing test.

To summarise, this thesis has presented work relating to the combination of computer vision and machine learning from text annotation. Its contributions include the following: a method for automatic character naming in broadcast video when varying levels of annotation are available; an RNN design that takes into account contextual information when generating captions; a complete caption generation method for broadcast television content; exploration of the emotional content of images and their captions along with the generation of captions that express emotion.

7.2 Future Work

This thesis has covered a broad range of topics, which have significant scope for extension and further work.

Character identification was performed using only facial features. The use of methods involving other features to uniquely identify characters should be explored. This would allow for better character identification performance in situations where faces are not fully visible. Full body person identifiers, and other biometric features could be applied. These features could also make use of the motion in the video rather than just utilising still image techniques on individual frames.

The RNN used for generating captions is capable of learning weightings for its attention on the images. The full segmentation of objects in MSCOCO could be used to aid the training of the visual attention. It may also be possible to pre-train RNNs on a larger corpus of language data, to improve the grammatical rules that are learnt. This would mean that the word embedding space is not being learnt from scratch on just the captioning data. The use of word vectors from other sources may allow for the expansion of vocabulary and the use of zero shot learning techniques, such as attributes, and relative attributes could be used to create captions with words outside of the training set. The ability to guess words in context is an important skill that

would represent a big leap forward for natural language generation.

The work in this thesis has focused more on still imagery or single frames captured from video. Description generation for video that takes into account multiple frames of input would allow better description of actions that involve movement. Humans are capable of using the context of a still image to assume the motion that is taking place. This is something that would be more difficult for a system to learn. For example, whether a car is being driven or is parked.

Further examination of the generated sentences could be performed. Comparison of the sentence quality compared with human and other caption generation techniques should give a better indication of the performance of the system than the automated metrics. This is more time consuming and is not fully automatic. Analysis of the unique sentences generated by the method compared with sentences that exist in the training set would show how well the technique copes with new situations.

Although a few types of additional context input were tested with the RNN, there is room for other types of data to be utilised. Time information could be used to decide between ambiguous scenes. Knowing the time of day would allow the image of a meal to be classified as breakfast or dinner. It could know whether someone was going to, or leaving work.

With the addition of location and character labels, the RNN was used for description generation for broadcast media. Forms of data augmentation could be used to expand the training set from its current size. This could be performed on both the image data and on the text data. The text description could potentially be automatically rewritten in different forms and using synonyms to create more variety in the descriptions. This could be extended to utilise training data from audio descriptions to expand the training data that is available. Newer RNN designs could be used to try and improve the performance from its current level.

The broadcast video description was trained on data from the series itself. While this allows it to be tailored to the content in the series, it does not allow for flexibility when new concepts are encountered. New episodes will often feature new objects, actions and locations, that have not been previously encountered. This is in order to keep the

content interesting for the viewer. Utilising more external training data from other datasets could allow for a more diverse vocabulary, and avoid any potential overfitting to the series.

The location and character labels could also be integrated into a broadcast media caption generation system in alternative ways. The generated captions could use generic place holder tokens, which are replaced by the character and location predictions after the caption is generated. This would strictly enforce the use of correct character and locations in the description created by the system.

In addition to creating automatic audio descriptions for the blind or enabling users to better search for content within a series, the caption generation could be used for other tasks. With a model that has been trained on all the existing content in a series, writers could generate new ideas for scripts that could help inspire them when creating new episodes. This could be extended to training RNN on the dialogue to generate new dialogue.

The emotional content of images and captions is an area that is ripe for further exploration. Replacing the emotional vectors based on Plutchik's wheel of emotions with features learnt by a CNN could be powerful. A CNN is capable of learning more nuanced and visually distinct emotions instead of a discrete set of categories. They may provide a much higher dimensional, and therefore complex representation of the emotional information contained in images.

The emotion for a caption was selected by setting the context emotion to a single value. By triggering multiple emotions simultaneously or by using continuous values, more complex combinations of emotions could be expressed. Emotional context could be coupled with the broadcast video description generation to allow for more emotionally rich descriptions that are potentially more atmospheric. This could create more appropriate and engaging captions for horror films, for example.

The emotional content of the captions was assessed through a survey of comparisons with generated captions and human generated captions. This could be expanded to ask the user to select which emotion they feel a caption is attempting to convey. This would be a more challenging task for the user to undertake, but also a more rigorous test of

the emotional content of the captions. The captions could also be judged in a Turing test, where the user is asked to judge whether they think a caption is automatically generated or written by a human. This would show if the emotional captions are considered more human.

Research into psychological disorders may also benefit from the modelling of emotional expression and its relation with images and language. Furthermore, combining this method for expressing emotion in natural language with a method that can recognise emotion could lead to a form of artificial empathy for the next generation of AI.

Appendix A

Location Recognition

Given a broadcast video and its associated script, there is a lot of extra data that can be leveraged from the script and associated with the video. One example is a label for the location of each scene. Using this information it is possible to learn the location names and recognise them in unseen footage. The locations may be seen from many different possible camera angles under many possible lighting conditions. In addition to this, the script may contain differences from the final production video. This creates extra difficulty in the script and video alignment as well as the learning of locations.

To learn the locations, the scenes in the script need to be aligned with the scenes in the video. The subtitles can provide coarse alignment with the script by matching the text for dialogue. Finer alignment can be achieved using shot boundary detection. To provide a way to summarise the location contained in a shot, a mosaic is created from multiple images of the same location. When assembling the mosaic, foreground objects are removed to make sure only the background location is visible. Features from the mosaics can then be matched to compare locations. An example mosaic created from *The Prisoner* is shown in figure A.1. A diagram of the location recognition pipeline is shown in figure A.2. The script alignment process is described in section A.2. Section A.2.2 details the shot boundary detection. Sections A.3 and A.4 discuss the mosaic creation process and matching process respectively.

This chapter uses many sets and symbols to formalise the processes it contains. Firstly, the script Γ is defined as set of text elements, $\Gamma = \{t_{0...n}\}$. The set of scenes in the text



Figure A.1: Example mosaic created from a shot from The Prisoner

are U^t and the scene boundaries are A . The subtitles are called sub with superscripts for the text, start and end times. A video is a set of images referred to as I . A video also contains a set of shots S and a set of shot boundaries SB . A set of shots that make up a scene is represented by U^s and the images in a scene use U^i . The mosaics that are created from shots use the symbol M . The functions used to composite mosaics are called ϕ .

The Prisoner (1967) was chosen as a large number of the external locations were filmed in Portmeirion in North Wales. This is a small village meaning that many of the locations will be repeated. In addition to this, the original production scripts are available, providing the required annotation. The series is available on Blu-Ray disc format with 1080p video at 24 Frames Per Second (FPS). The dataset consists of 17 episodes of 50 minutes each with a script for each episode.

A.1 Background

A shot in a video sequence is a group of continuous frames between cuts. The cuts can take various forms such as fades or abrupt transitions. A scene within a video sequence is a group of shots that are spatially and temporally contiguous. Dividing a video into shots is useful and can help with the scene boundaries. Hanjalic [61] defines the problem of Shot Boundary Detection (SBD) as finding discontinuities in

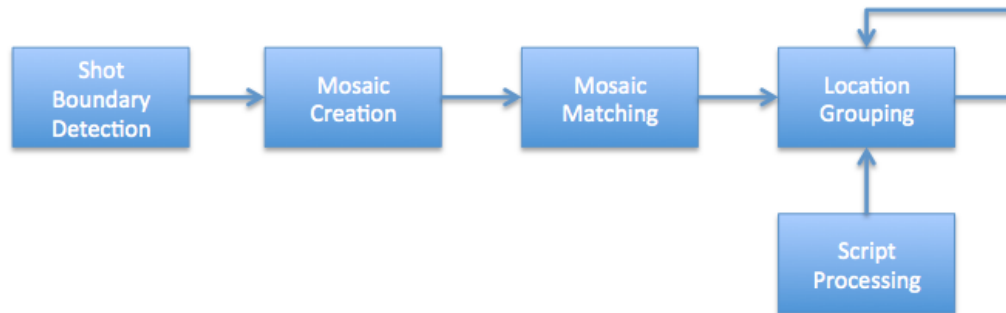


Figure A.2: Layout of the location recognition pipeline

the feature distance between frames. Many different feature types have been used for SBD including: Pixel differences [188], Colour histograms [109], tracking of features [51] and mutual information [20]. Yuan et al.[187] try to provide a formal framework for SBD and review many existing techniques while suggesting optimal criteria for the different sections of the framework. To take advantage of existing information, Patel et al. [130] use the motion vectors from the video compression to detect cuts. Due to the large number of approaches available, there have been many survey papers such as Boreczky and Rowe [13] and Smeaton et al. [149]. Boeczky and Rowe find that in general the approaches based on region-based comparisons and motion vectors worked better and simpler algorithms out performed more complex ones. Smeaton et al. found that there was usually very little difference in performance between the top approaches and abrupt shot cut performance is still a lot higher than gradual transitions.

Once the video has been divided into shots, information is still spread across all the frames within the shot and at the same time there is a great deal of redundancy. These frames can be combined to create a mosaic, which summarises the whole shot. The first step of mosaic creation is to register the frames. Image registration is the process of transforming 2 or more frames into the same co-ordinate space. Current approaches can be divided into those which operate in the spatial domain and those operating in the frequency domain. They can also be further sub-divided into approaches based on dense image intensity comparisons and approaches based on sparse feature comparisons. The general pipeline for a feature based system involves detecting the features, match-

ing the features, estimating the transformation and then transforming the image with resampling [194]. Brown and Lowe [15] use Scale-Invariant Feature Transform (SIFT) features and then a Random Sample Consensus (RANSAC) based homography estimator to perform the registration. They also allow other constraints to be applied such as using the horizon to straighten an image. Other approaches have used simpler features such as line segments [67] and image regions [50]. Zitova and Flusser note that feature based approaches work very well in cases where input images contain lots of detail but in cases where there is low detail image patch based systems can work better. Reddy and Chatterji [135] use the phase difference in frequency domain to perform the image registration. The Fast Fourier Transform (FFT) allows very fast conversion into the frequency domain and as correlation and convolution in the time domain are equivalent to multiplication in the frequency domain, this can provide speed benefits. In image intensity based systems, the feature detection and comparison is replaced by the matching of image patches. Intensity based matches are more common in medical imaging such as Kim and Fessler [83] and Johnson and Christensen [76]. Steedly et al. [153] propose a method for speeding up registration on a large number of frames in video sequences by identifying key frames based on overlap. The key frames are all matched to each other but the intermediate frames are matched linearly. This reduces the complexity of calculations.

Optical flow is a special case of non-rigid registration and is used to estimate the motion vectors of pixels between frames. This allows the use of optical flow to estimate objects moving in the shot. Optical flow can be based around densely tracking the entire image or using sparse interest points. Some of the earliest work in this field was by Horn and Schunck [66] who suggested an approach based on brightness and smoothness constraints. Lucas and Kanade [105] use an affine model for registering image patches and the flow field between images. Bruhn et al. [17] combine local approaches such as Lucas and Kanade with global approaches such as Horn and Schunck. They propose 2 combined methods, one based on spatial smoothing and the other based on spatiotemporal smoothing. Farnebäck [44] extends previous work on using orientation tensors and parametric motion to segment the motion field simultaneously to estimating the motion. This allows for more accurate estimation of motion as there are different re-

gions, which have different motion models applied to them. Roth and Black [139] use a statistical approach to learn a model of flow fields from example natural scenes. Sun et al. [154] take the learning of optical flow further by learning both the data and spatial terms. Liu et al. [102] propose using SIFT to densely create descriptors for an image and track them into the next image. Brox et al. [16] propose a method based on optimising the brightness, gradient and a discontinuity preserving smoothness constraint. The variational approach allows for large discontinuities in the motion and implements a coarse-to-fine warping. Xu et al. [175] is a multi-scale variational approach that builds upon Brox et al. and extends the coarse-to-fine warping to improve fine motion estimation. Deep learning has also been used to improve optical flow by Weinzaepfel et al. [169].

Once the frames within a shot have been registered, the frames may be combined into a mosaic, which summarises the shot information improving robustness and removing redundancy. This is difficult because in a shot there will often be moving objects, people or the effects of parallax. There are many different approaches to filtering and blending the images to create the final mosaic. Brown and Lowe [15] used techniques based on multi-band blending, exposure compensation and seam detection. This technique allows for a very high quality final mosaic, but is not suitable for situations with moving objects. Irani et al. [73], who first proposed the idea of using mosaics to summarise shot information, suggest several techniques for dealing with motion within a scene. These techniques include temporal average, temporal median and the weighted median. Aner and Kender [4] use a similar median filter to remove moving objects in the mosaic. Davis [33] uses the distance between pixels and Dijkstras algorithm to find the best path to decide which pixels to use. The advantage of using the median or Davis' approach is that a single pixel from the input is used in the output so the mosaic is sharp compared to averaging pixel values. Hsu and Tsan [68] use a block based method that compares the motion of blocks to a global motion model and remove blocks which disagree.

A.1.1 Location Matching

To allow them to be grouped into scenes so that a higher level understanding of the video content can be achieved, the background mosaics that represent the shots need to be compared. Schaffalitzky and Zisserman [142] proposed a method for automatic location matching based on comparing shots in videos. The process involved calculating invariant features and removing those that have many intra-frame matches to eliminate non-discriminative features. These features are then matched to those of other images and filtering ensures the robustness of the computed matches, including neighbourhood consensus and fitting epipolar geometry. The number of successful feature matches is used to score the match between images. Chum et al. [24] used a similar method but with a much larger dataset. They first used a SIFT bag of visual words system to find the best candidate matches in the dataset before using a homography to perform spatial verification and ranking the matches by number of inliers. These previous methods are designed for matching single frames or images. Schaffalitzky and Zisserman extended this by tracking their features within shots to find the average value for their features. Aner and Kender [4] instead divided the mosaics into vertical strips and compare them. This was applied to shot mosaics rather than keyframes to allow the whole shot to be utilised.

A.2 Script Alignment

Television and film scripts contain the details for the scenes within a video sequence. There is a standardised format for scripts and their contents. This format specifies the location in which the scene occurs, editorial information for the shots, a description of the activities and actions occurring within the scene and the dialogue spoken by the characters. The location information is contained within the “slug lines”, which state if the scene is exterior or interior, where it is and the time of day it occurs. To formalise, the script Γ contains text elements t , $\Gamma = \{t_{0...n}\}$. The scene boundaries (A) from the script are defined as a subset of the script, $A \subset \Gamma$. The text in the scene U^t is the text

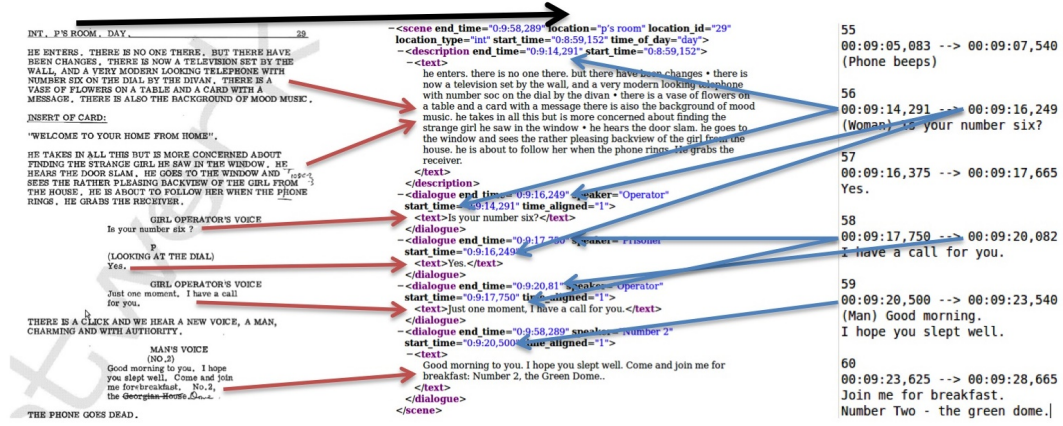


Figure A.3: Matching between the script and the subtitles

between the scene boundaries in the script,

$$U^t_j = \{j | A_{k-1} < j < A_k\}. \quad (\text{A.1})$$

The script does not, however, contain any information about the timing of the scenes within the video. To use the scripts as annotation, the contents of the video and the script need to be aligned.

A.2.1 Subtitles

Initial alignment can be performed by matching the video subtitles with the dialogue in the scripts. The subtitles contain text and timing information. Figure A.3 shows the alignment between the script and subtitles and an example of how the script and subtitles are matched. In figure A.4 the black elements are well aligned, blue elements are interpolated and orange elements are when multiple script elements occur simultaneously. The time axis is showing 5 second intervals. The subtitles (sub) have text, $\text{sub}^t = t$, as well as start frames, sub^s , and end frames, sub^e . Fuzzy matching between subtitle text and script text is used to assign start and end frames to script text. This identifies the images that belong to a scene U^i ,

$$U^i_j = \{j | \text{argmin}(\text{sub}^s) < j < \text{argmax}(\text{sub}^e) \text{ and } \text{sub}^t \in U^t_j\}. \quad (\text{A.2})$$

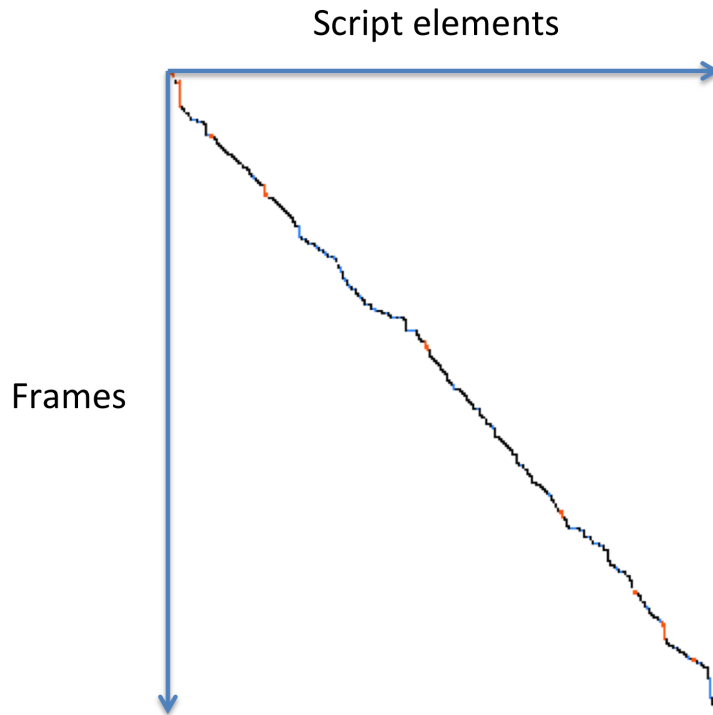


Figure A.4: Interpolated timing for the script and subtitles

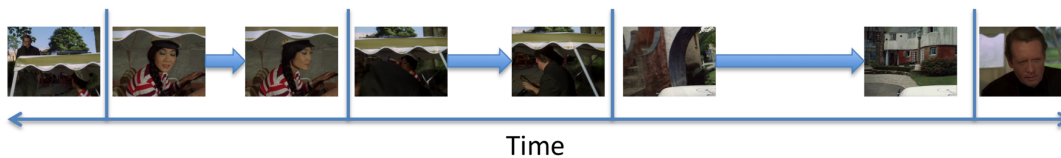


Figure A.5: Diagram showing a timeline of frames with lines representing shot boundaries

A.2.2 Shot Cuts

The frames that occur between subtitles are ill-defined. A scene can contain many shots and a scene cannot change during a shot. A shot represents part of the video sequence that contains sequential frames from a camera over time. Shots are typically short but can last much longer in some instances. Figure A.5 shows a timeline of frames with lines representing shot cuts. Using location annotation from the script requires that the video is divided into shots so that they can be matched to scenes

that have location labels. A shot boundary detection method based on calculating the homography from one frame to the next is used. This was chosen to make the shot boundary detection more robust to changes in illumination between frames within a scene. Methods that use average frame colour or a histogram of colour within a frame can trigger false positives when an illumination change occurs. Very gradual changes between shots might be missed by these methods also. A shot boundary is detected when the homography between shots is invalid. The use of homographies accounts for moving cameras (pan, tilt, zoom and translation).

A video consists of a set of images, $I = \{i_{1..n}\}$. A shot, s is a set of images between shot boundaries, $S = \{i_0 \dots i_T\}$. The Shot Boundary (SB) is where the homography ($h_{j,j+1}$) between consecutive images (i_j and i_{j+1}) leads to a transformed image with an area smaller than a threshold (t),

$$SB = \left\{ j \mid \frac{|h_{j,j+1}(i_j)|}{|i_j|} < t \right\}. \quad (\text{A.3})$$

A particular shot (S_k) is defined as the set of frames between 2 consecutive shot boundaries (SB_k and SB_{k+1}),

$$S_k = \{i_j \mid SB_k < j < SB_{k+1}\}. \quad (\text{A.4})$$

This means that the shot contains the frames that occur between one shot boundary and the next shot boundary. More gradual scene transitions may have partially transitioned frames at the end of one shot and at the start of the next. Alternatively there may be a false positive shot found between the two shots. The scenes U^s are defined by using the shots that belong to them,

$$U^s_k = \left\{ \bigcup_j S_j \mid S_j \cap U^s_k \neq \emptyset \right\}. \quad (\text{A.5})$$

This provides all the shots belonging to a scene so that frames that occur after or before a subtitle are also included. This allows location information after the end of dialogue to be used. For example a camera may pan away from the speakers across the location after the dialogue ends. There may also moments of silence or further shots with no dialogue.

A.3 Mosaic Creation

To discover repeated locations in the video sequence, matching every frame to every other frame is too computationally expensive. A video at 24 FPS will contain 86,400 frames per hour of footage, relating to over 7 billion comparisons. In contrast, the number of shots per hour of footage is typically less than 1,000 and frames within a shot are by definition highly redundant. To be able to perform a single comparison for each shot, it is necessary to summarise the information contained within the constituent frames of that shot.

A simple approach would be to select a single frame to represent the shot. This could be the first frame, the middle frame or a frame chosen by a technique that tries to find a representative frame. This leads to lost information from the rest of the frames within the shot. Another approach is to create a histogram representation for each frame in a shot and then average the histograms over the length of the shot. Histogram intersection could also be used to combine the histogram representation for multiple frames into a single representation. This does not create a representation that is as useful to humans and geometric information is lost. To create a representation that tries to retain as much information as possible, multiple frames can be combined into a single compound image.

A set of mosaics (M) are created with a mosaic (m_j) for each shot (S_j), $M = \{m_{0...|S}\}$. An implementation of the method described by Brown and Lowe [15] is used to match each frame with every other frame in the shot and bundle adjustment is used to find a transformation (h_{j0}) for each frame into the mosaic co-ordinate space. Note that the bundle adjustment is computationally expensive so an upper bound was set on the number of frames per shot with some frames being skipped in longer shots.

A.3.1 Compositing

Many shots will contain objects that move and obscure parts of the background location. To improve matching between locations, it is desirable to remove these objects. This ensures the mosaic encodes as much of the location as possible when matching with

other mosaics. With all the frames transformed into the same co-ordinate space there is a great deal of redundancy between the frames. For each pixel co-ordinate in the mosaic space, there are many possible pixel values from the different input frames.

The mosaics are defined as the result of applying an accumulator function (ϕ), to all images (i_j) belonging to the relevant shot, after a transformation (h_{j0}),

$$m_k = \{\phi(h_{j0}(i_j)) \text{ for all } i_j \in S_k\}. \quad (\text{A.6})$$

The function ϕ outputs a single pixel for each location in the mosaic (m). Allowing a single value to be chosen resulting in a sharper mosaic image. A very simple option is to take the final pixel at each location, this aims to remove most instances of a moving object by assuming that an object will have moved from that location by the time a pixel is last updated. To achieve this, the accumulator function ϕ_m , which for each mosaic pixel x , takes the most recent (i.e. highest index) images pixel y , that corresponds to pixel x after transformation by h_{j0} ,

$$\phi_m(x) = \{i_{\max(j)}(y) \mid i_j \in S_k \text{ and } h_{j0}(y) = x\}. \quad (\text{A.7})$$

This is not effective and results in the objects still being present at their final position within the shot. An example of this is shown in figure A.7a. This is because at the end of the shot, the last updated pixel values at the object locations belong to the object.

A.3.2 Median filter

By assuming the majority of the pixels falling at each point on the mosaic are of the background, the accumulator function ϕ_n takes the median pixel at each location to try and choose a pixel that belongs to the background. The RGB pixel values are converted to luminance values and then sorted so that the median value can be found.

$$\phi_n(x) = \text{median}(\{i(y) \mid i_j \in S_k \text{ and } h_{j0}(y) = x\}). \quad (\text{A.8})$$

A.3.3 Optical Flow

Another way to increase the chance of picking a pixel that belongs to the background is to detect and filter foreground pixels by their motion. This can be achieved by using optical flow techniques. Optical flow is a technique for estimating the motion vector of each pixel from one image to another. The output is a vector, which describes how each pixel in one image moves to the second image. Since the frames from the shot have all been registered by a homography, the movement of pixels between the warped frames should relate to residual non-rigid movement within the shot. Optical flow is used to estimate this motion between consecutive registered frames. Pixels that have a motion vector with a magnitude greater than a threshold are not used in the mosaic compositing stage. Figure A.6 shows an example of a mask created from the magnitude of the motion vectors estimated by optical flow. $O(x)$ is defined as an optical flow thresholding function, which returns 0 if the pixel experiences negligible motion, and 1 if the pixel represents a moving object. This optical flow function is combined with the median filter of equation A.8 to create mosaics,

$$\phi_o(x) = \text{median}(\{i_j(y) \mid i_j \in S_k \text{ and } h_{j0}(y) = x \text{ and } OF_j(y) = 0\}). \quad (\text{A.9})$$



Figure A.6: Example mask created from thresholding the magnitude of the motion vectors in a flow field estimated using optical flow

With fewer pixels of foreground objects being used in the mosaic creation, it is more likely that a background pixel will be chosen.

A.3.4 Mosaic Results

Example mosaics from the 3 different composition techniques are shown in figure A.7. Figure A.7a shows where the most recent pixel has been used, this leaves the vehicle still visible in one location highlighted in the red box. In figure A.7b, the median pixel has been used and this leads the vehicle being removed from most parts of the image. There are still some artefacts visible that are highlighted in the red boxes. The last mosaic, in figure A.7c, shows where the optical flow mask has been used to remove foreground pixels before using the median filter. In this image, the vehicle has been removed with only small artefacts remaining in the red boxes. Using optical flow allows for the creation of the mosaics that contain the least foreground pixels out of the 3 techniques demonstrated.

Examples of the frames used to create the mosaics shown in figure A.7 are given in figure A.8. The frames show that the vehicle is present in different locations during the shot. More mosaics, which have been created using the optical flow based method are shown in figures A.9, A.10, A.11, A.12 and A.13. In figure A.9 moving objects such as the front door have been removed from the image as well as “The Prisoner” character, who walks through the room. The removal of the door is an unintended consequence of removing objects based on their motion but is not inherently good or bad. In figure A.10 only the static characters are visible and in figure A.11 the actor is not visible despite walking through the scene. Figure A.12 shows a mosaic created from a shot dominated by two moving characters and a third background character. This is the same shot from which the optical flow mask in figure A.6 originates. Despite the moving characters the mosaic creation process still works. There is a gap in the image behind the character on the left where the background is never visible. In figure A.13 the actor is again not visible despite being obviously visible in all frames in the shot. An example where the process fails is shown in figure A.14. There is a lack of distinctive features on the flat sandy beach and this has led to failures with the optical flow and



(a) Top pixel mosaic



(b) Median mosaic



(c) Optical flow median mosaic

Figure A.7: Mosaic created from a shot in *The Prisoner*. A.7a shows the most recent pixel at each point. The A.7b shows the results of the median filter and A.7c shows the results of using optical flow to remove the moving object.



Figure A.8: 5 examples of the frames used to create the mosaics in figure A.7. The car moves from left to right and obscures parts of the scene



Figure A.9: Example mosaic of the interior of The Prisoner's flat with the example frames above

registration. The lack of features will make it more difficult to match with other images of the same location.

The mosaic creation process will be most effective on wide panoramic shots. Close ups of faces do not work very well but by definition do not contain substantial location information. Shots where there is little background detail could cause the object to be registered to itself so the background pixels will be removed by the optical flow mask.

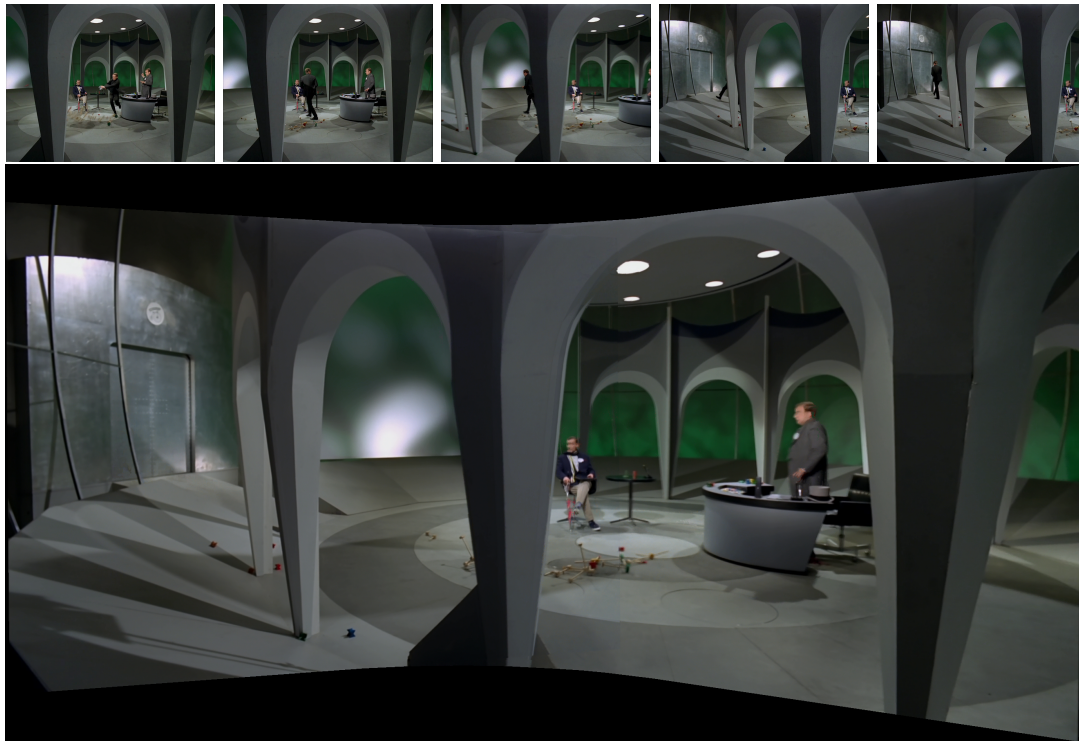


Figure A.10: Example mosaic showing the interior of the labour exchange with example frames above

A.4 Location Matching

To recognise which shots occur in the same location, it is necessary to match mosaics together. This matching system must be robust to mosaics with only a partial overlap while rejecting mosaics of different locations. Similar to the approach to SBD in section A.2.2, the matching techniques are based on estimating homographies from one mosaic to the other. As such, the matching relies on the spatial properties of the locations. The function (SiftP) uses SIFT to create a set of points (P_i) for a mosaic (m_i),

$$P_i = \text{SiftP}(m_i). \quad (\text{A.10})$$

As in equation A.11, the key points (P_i) are given to a function (SiftD) that generates a set of feature vectors (F_i) for each mosaic (m_i),

$$F_i = \text{SiftD}(m_i(P_i)). \quad (\text{A.11})$$



Figure A.11: Example mosaic showing the street just outside the cafe with example frames above

These feature vectors are used to match points between images using their distance in the feature space to find corresponding points between images. The ratio between the first and second best match for a feature is used to reject poor quality matches as suggested by Lowe [104]. The inliers ($\psi_{i,j}$) of the inter-frame homography ($h_{i,j}$) are defined as the points (from the set of 2D key points (P)), where the distance of the transformed points to their correspondences is less than the threshold (t),

$$\psi_{ij} = \left\{ k \mid \mathbf{p}_k \in P_i \text{ and } |h_{ij}(P_k) - P'| < t \text{ and } P' = \min_P(P_j, P_k) \right\}. \quad (\text{A.12})$$

A value for the threshold can be found by using an Receiver Operating Characteristic (ROC) curve to find an appropriate balance between the true positives and false positives. Given these definitions, the quality of the matches can be measured by a number of distance functions. A simple distance function based on the total number of inliers similar to the work of Schaffalitzky and Zisserman [142] and Chum et al. [24] is defined as,

$$d_i(m_i, m_j) = \frac{1}{|\psi_{ij}|}. \quad (\text{A.13})$$



Figure A.12: Example mosaic showing a close view of the cafe with example frames above

An alternative approach, similar to the SBD method in section A.2.2 is to examine the area of the mosaic (m_i) transformed by the homography (h_{ij}) divided by the area of (m_i),

$$d_a(m_i, m_j) = \left| 1 - \frac{|h_{ij}(m_i)|}{|m_i|} \right|. \quad (\text{A.14})$$

Another method using the point correspondences between mosaics is to calculate the distance between the feature vectors. This tries to find a similarity between images as if the images are similar the distance between feature vectors should be low. Equation A.15 uses the mean distance between the (\mathbf{f}) and their nearest neighbours (\mathbf{f}') from the other mosaic. This gives a comparison based on how similar the features are in each mosaic.

$$d_F(m_i, m_j) = \frac{1}{|F_i|} \sum_{\mathbf{f} \in F_i} |\mathbf{f} - \mathbf{f}'| \text{ where } \min_{\mathbf{f}' \in F_j} |\mathbf{f} - \mathbf{f}'| \quad (\text{A.15})$$

This idea can also be restricted to use just the inliers (ψ_{ij}) as in equation A.16.

$$d_{F\psi}(m_i, m_j) = \frac{1}{|F_i|} \sum_{\mathbf{f} \in F_i} |\mathbf{f}_k - \mathbf{f}'_k| \text{ where } k \in \psi_{ij} \text{ and } \min_{\mathbf{f}'_k \in F_j} |\mathbf{f}_k - \mathbf{f}'_k| \quad (\text{A.16})$$

Inliers are found using RANSAC [49], where random samples are used to try to fit a



Figure A.13: Example mosaic of the street outside the cafe including the information point with example frames above

model to a subset of the data to find the best model.

A.5 Results

From the first episode, a subset of 30 mosaics were annotated to create a test set. Mosaics were chosen based on their quality (higher quality mosaics were preferred) and at least 2 mosaics were included for each location. A few examples of mosaics without matches were also chosen to test for false positives. Ground truth was created for this subset and is shown in figure A.15. It shows each mosaic matched against every other mosaic and is white when they match and black when they do not match. The diagonal shows where each mosaic is matched with itself. The blocks on the diagonal are where multiple mosaics from the same location are consecutive.

The ROC curves in figure A.16 show the effects of changing the thresholds for the different matching functions (equations A.13, A.14, A.15 and A.16). The inlier count based method that is shown in figure A.16a gives a 65% True Positive Rate (TPR) with a very low False Positive Rate (FPR). It has an Area Under Curve (AUC) of 0.85,



Figure A.14: Example case where the mosaic creation process has resulted in a low quality mosaic

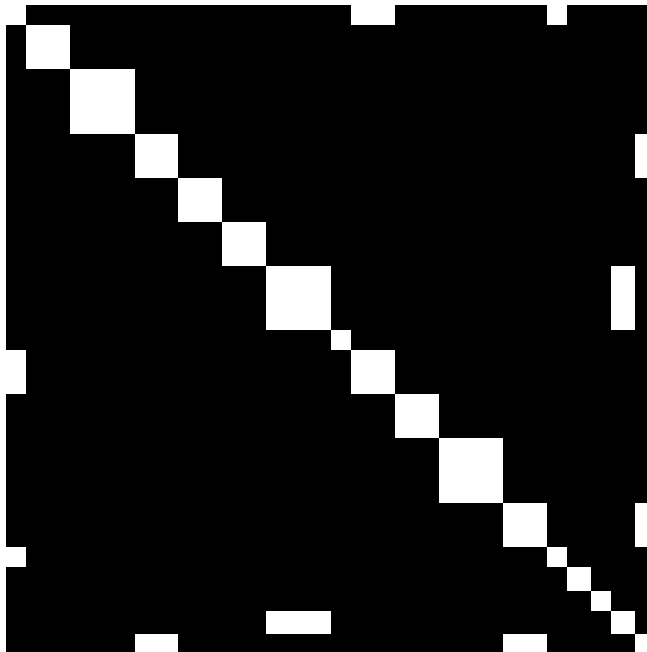


Figure A.15: The ground truth for the 30 mosaic dataset

which is the best out of the four methods. The second best AUC is 0.84 from the mean SIFT inliers method, which is shown in figure A.16d. This gives its best performance with a TPR of 80% where the FPR is around 20%. This means that the inlier count method gives a good TPR when a very low FPR is required but the mean SIFT inlier method gives a better TPR when a slightly higher FPR is acceptable. The methods in figures A.16a and A.16b work best at a low FPR while the methods in figures A.16c

and A.16d work best at a high FPR. This may be due to the homography matching being more likely to either match or fail, whereas the distance in SIFT space can allow for a more gradual failure.

The best results are gained from the inlier count based matching despite its simplicity. The results of the inlier based matching function on the 30 mosaic dataset are shown in figure A.17. As with figure A.15 it shows the comparison of each mosaic against every other mosaic. The white blocks in the black space away from the diagonal show false positives. There is a lack of symmetry in the matching matrix and it may be possible to use this to remove some false positives where the match is only found in one direction. If one image can be transformed onto the other but not in the other direction this may mean that there is not a good match between locations. The location matching matrix was also computed over the entire set of mosaics using the inlier threshold selected from the ROC curve and is shown in figure A.18. The mosaics are in the order that they appear in the episode so the axes represent mosaics and time. The blocks on the diagonal represent scenes where the same repeated location has been detected as mosaics have been matched to other matches in the same temporal section. A zoomed in section of the full results matrix is shown in figure A.19, in this it is possible to see a large block representing part of a scene in the video. The shots of the store inside are matched together and the close ups of the actor's face are matched together correctly. This shows the pattern of moving shots back and forth between viewpoints that is common in broadcast video.

A.6 Chapter Conclusions

The script and subtitles can be used to gain aligned annotation for the video. The use of shot cuts allows for further refinement of this alignment to match the boundaries of scenes within the script. This allows for the location from each scene in the script to be associated with the shots belonging to the scene.

To summarise the background visual content of a shot, it is possible to create a mosaic of the location. Multiple frames within the shot are combined into a single image and using median filtering and optical flow it is possible to further remove foreground

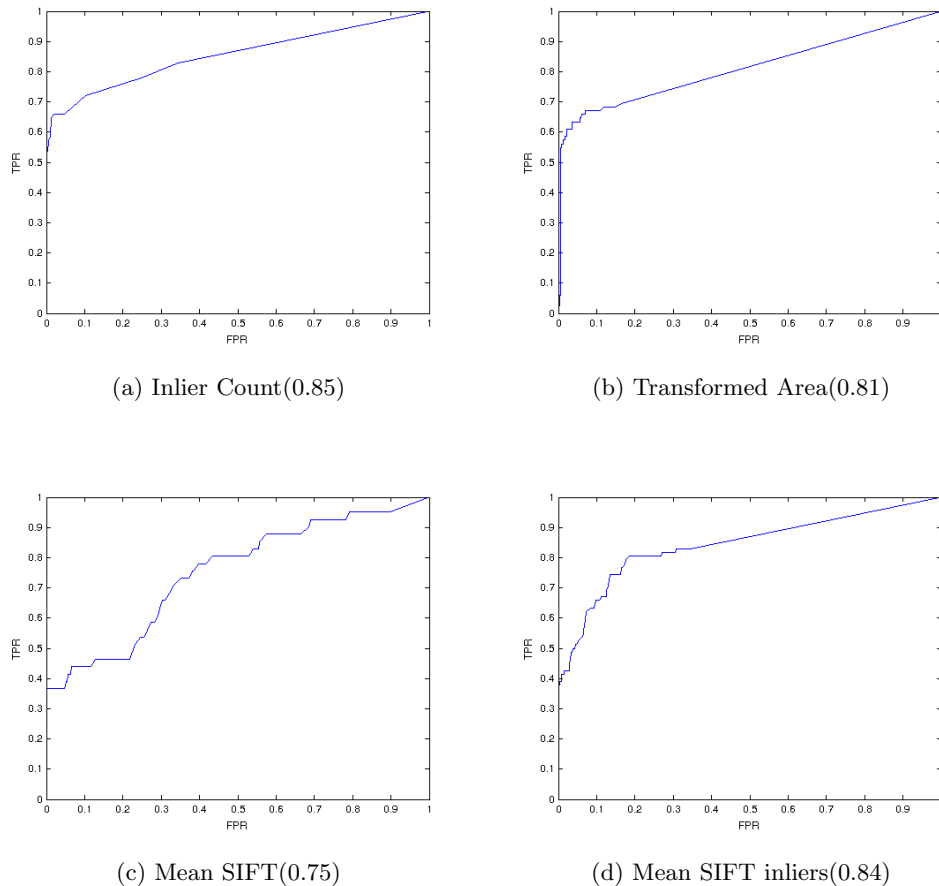


Figure A.16: ROC curves for the different mosaic comparison functions. The AUC for each is shown in the brackets.

objects that obstruct the view of the location. However, mosaics are not possible for each and every shot such as when camera motion occurs or when a shot contains frames that are difficult to register due to lack of good features. Parallax is another problem that can effect the mosaic creation process and create false positives for the motion detection from optical flow. When mosaics are well created the output is potentially more useful to a human for creating thumbnails or other manual work on videos.

Given a set of automatically extracted mosaics, common locations for shots can be identified by extracting features and matching them between mosaics. Filtering based on the number of inliers provides the best matching quality. Other techniques for extracting features could be used with the mosaics to provide potentially better matching.

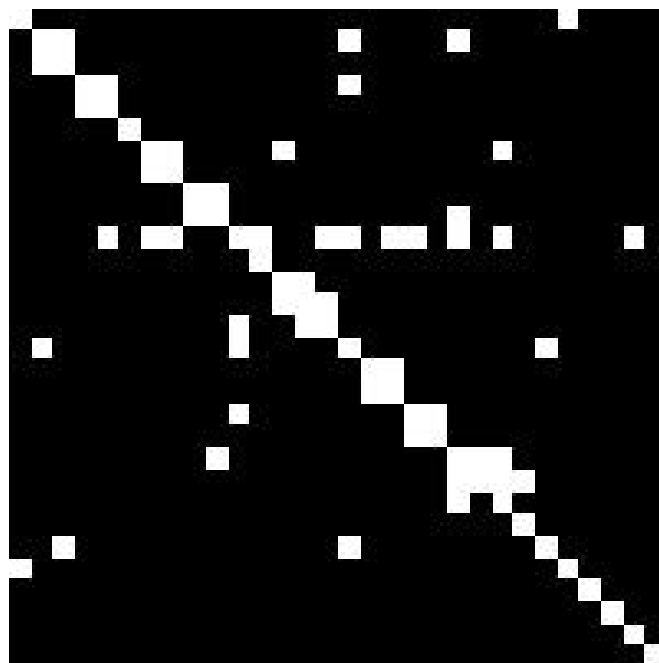


Figure A.17: The results using the matching defined in equation A.13

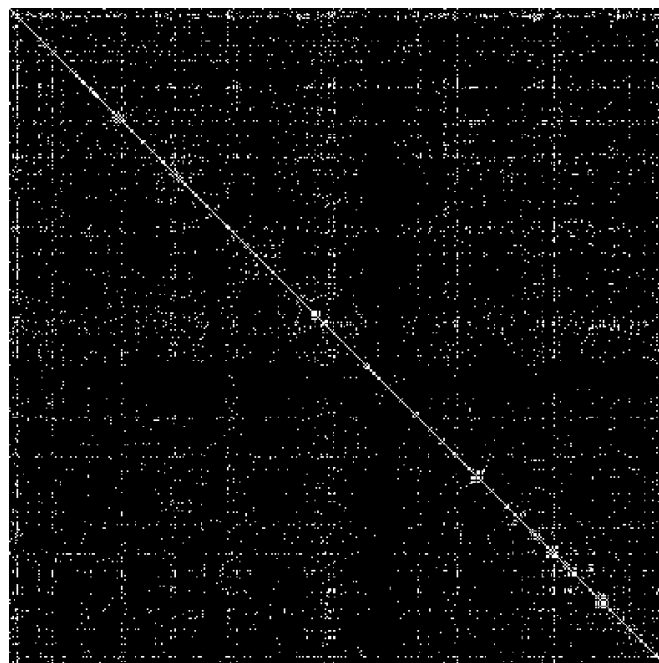


Figure A.18: Results of the inlier based mosaic matching on the full mosaic dataset.

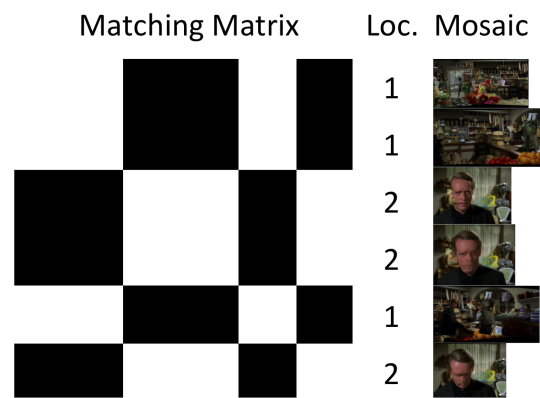


Figure A.19: A zoomed section with example mosaics.

References

- [1] *IMDb Statistics*. <https://www.imdb.com/pressroom/stats/>.
- [2] Alexe, B., Deselaers, T., and Ferrari, V. Measuring the objectness of image windows. *IEEE transactions on pattern analysis and machine intelligence*, 34(11):2189–2202, 2012.
- [3] Amos, B., Ludwiczuk, B., and Satyanarayanan, M. Openface: A general-purpose face recognition library with mobile applications. Technical report, CMU-CS-16-118, CMU School of Computer Science, 2016.
- [4] Aner, A. and Kender, J. Video summaries through mosaic-based shot and scene clustering. In *European Conference on Computer Vision*, pages 388–402. 2002.
- [5] Asghar, N., Poupart, P., Hoey, J., Jiang, X., and Mou, L. Affective neural response generation. In *European Conference on Information Retrieval*, pages 154–166. Springer, 2018.
- [6] Bandhakavi, A., Wiratunga, N., Massie, S., and Padmanabhan, D. Lexicon generation for emotion detection from text. *IEEE Intelligent Systems*, 32(1):102–108, Jan 2017.
- [7] Banerjee, S. and Lavie, A. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, volume 29, pages 65–72, 2005.
- [8] Barr, J. R., Bowyer, K. W., Flynn, P. J., and Biswas, S. Face recognition from

-
- video: A review. *International Journal of Pattern Recognition and Artificial Intelligence*, 26(05), 2012.
- [9] Bauml, M., Tapaswi, M., and Stiefelhagen, R. Semi-supervised Learning with Constraints for Person Identification in Multimedia Data. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3602–3609, 2013.
- [10] Belhumeur, P., Jacobs, D., Kriegman, D., and Kumar, N. Localizing parts of faces using a consensus of exemplars. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 545–552, 2011.
- [11] Bengio, Y., Ducharme, R., Vincent, P., and Jauvin, C. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155, 2003.
- [12] Bojanowski, P., Bach, F., Laptev, I., Ponce, J., Schmid, C., and Sivic, J. Finding actors and actions in movies. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2280–2287, 2013.
- [13] Boreczky, J. S. and Rowe, L. A. Comparison of video shot boundary detection techniques. *Journal of Electronic Imaging*, 5(2):122–129, 1996.
- [14] Borth, D., Ji, R., Chen, T., Breuel, T., and Chang, S.-F. Large-scale visual sentiment ontology and detectors using adjective noun pairs. In *Proceedings of the 21st ACM international conference on Multimedia*, pages 223–232. ACM, 2013.
- [15] Brown, M. and Lowe, D. G. Automatic panoramic image stitching using invariant features. *International journal of computer vision*, 74(1):59–73, 2007.
- [16] Brox, T., Bruhn, A., Papenber, N., and Weickert, J. High accuracy optical flow estimation based on a theory for warping. In *European conference on computer vision*, pages 25–36. Springer, 2004.
- [17] Bruhn, A., Weickert, J., and Schunck, C. Lucas/kanade meets horn/schunck: Combining local and global optic flow methods. *International journal of computer vision*, 61(3):211–231, 2005.

-
- [18] Brysbaert, M. and New, B. Moving beyond kučera and francis: A critical evaluation of current word frequency norms and the introduction of a new and improved word frequency measure for american english. *Behavior research methods*, 41(4):977–990, 2009.
- [19] Brysbaert, M., Stevens, M., Mandera, P., and Keuleers, E. How many words do we know? practical estimates of vocabulary size dependent on word definition, the degree of language input and the participant’s age. *Frontiers in psychology*, 7:1116, 2016.
- [20] Cernekova, Z., Nikou, C., and Pitas, I. Shot detection in video sequences using entropy based metrics. In *Proc. ICIP*, volume 1, pages 156–165, 2002.
- [21] Chen, T., Borth, D., Darrell, T., and Chang, S. Deepsentibank: Visual sentiment concept classification with deep convolutional neural networks. abs/1410.8586, 2014.
- [22] Chen, X. and Zitnick, C. L. Learning a recurrent visual representation for image caption generation. 2014.
- [23] Chopra, S., Hadsell, R., and LeCun, Y. Learning a similarity metric discriminatively, with application to face verification. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 539–546. IEEE, 2005.
- [24] Chum, O., Philbin, J., Sivic, J., Isard, M., and Zisserman, A. Total recall: Automatic query expansion with a generative feature model for object retrieval. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8, 2007.
- [25] Cinbis, R. G., Verbeek, J., and Schmid, C. Unsupervised metric learning for face identification in tv video. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1559–1566. IEEE, 2011.
- [26] Cootes, T. F., Ionita, M. C., Lindner, C., and Sauer, P. Robust and accurate

-
- shape model fitting using random forest regression voting. In *European Conference on Computer Vision*, pages 278–291. Springer, 2012.
- [27] Cootes, T. F., Edwards, G. J., and Taylor, C. J. Active appearance models. In *Proc. ECCV*, 1998.
- [28] Cour, T., Sapp, B., Jordan, C., and Taskar, B. Learning from ambiguously labeled images. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 919–926, 2009.
- [29] Cui, Z., Li, W., Xu, D., Shan, S., and Chen, X. Fusing robust face region descriptors via multiple metric learning for face recognition in the wild. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 3554–3561. IEEE, 2013.
- [30] Cunado, D., Nixon, M. S., and Carter, J. N. Automatic extraction and description of human gait models for recognition purposes. *Computer Vision and Image Understanding*, 90(1):1–41, 2003.
- [31] Dantone, M., Gall, J., Fanelli, G., and Van Gool, L. Real-time facial feature detection using conditional regression forests. In *Computer vision and pattern recognition (CVPR), 2012 IEEE conference on*, pages 2578–2585. IEEE, 2012.
- [32] Davies, M. The corpus of contemporary american english as the first reliable monitor corpus of english. *Literary and linguistic computing*, 25(4):447–464, 2010.
- [33] Davis, J. Mosaics of scenes with moving objects. In *Computer Vision and Pattern Recognition, 1998. Proceedings. 1998 IEEE Computer Society Conference on*, pages 354–360, 1998.
- [34] Deng, J., Dong, W., Socher, R., Li, L. J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.
- [35] Devlin, J., Cheng, H., Fang, H., Gupta, S., Deng, L., He, X., Zweig, G., and Mitchell, M. Language models for image captioning: The quirks and what works. 2015.

-
- [36] Devlin, J., Gupta, S., Girshick, R., Mitchell, M., and Zitnick, C. L. Exploring nearest neighbor approaches for image captioning. 2015.
- [37] Donahue, J., Anne Hendricks, L., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., and Darrell, T. Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2625–2634, 2015.
- [38] Eickeler, S., Wallhoff, F., Lurgel, U., and Rigoll, G. Content based indexing of images and video using face detection and recognition methods. In *Acoustics, Speech, and Signal Processing, 2001. Proceedings.(ICASSP'01). 2001 IEEE International Conference on*, volume 3, pages 1505–1508. IEEE, 2001.
- [39] Elliott, D. and Keller, F. Image description using visual dependency representations. In *EMNLP*, volume 13, pages 1292–1302, 2013.
- [40] Everingham, M., Sivic, J., and Zisserman, A. “Hello! My name is... Buffy” – automatic naming of characters in TV video. In *Proc. BMVC*, 2006.
- [41] Everingham, M., Sivic, J., and Zisserman, A. Taking the bite out of automatic naming of characters in TV video. *Image and Vision Computing*, 2009.
- [42] Fang, H., Gupta, S., Iandola, F., Srivastava, R. K., Deng, L., Dollár, P., Gao, J., He, X., Mitchell, M., Platt, J. C., et al. From captions to visual concepts and back. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1473–1482, 2015.
- [43] Farhadi, A., Hejrati, M., Sadeghi, M. A., Young, P., Rashtchian, C., Hockenmaier, J., and Forsyth, D. Every picture tells a story: Generating sentences from images. In *European Conference on Computer Vision*, pages 15–29. Springer, 2010.
- [44] Farneback, G. Very high accuracy velocity estimation using orientation tensors, parametric motion, and simultaneous segmentation of the motion field. In *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, volume 1, pages 171–177. IEEE, 2001.
- [45] Fellbaum, C. *WordNet*. Wiley Online Library, 1998.

-
- [46] Feng, Y. and Lapata, M. How many words is a picture worth? automatic caption generation for news images. In *Proceedings of the 48th annual meeting of the Association for Computational Linguistics*, pages 1239–1249. Association for Computational Linguistics, 2010.
- [47] Feng, Y. and Lapata, M. Automatic caption generation for news images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(4):797–812, 2013.
- [48] Ferraro, F., Mostafazadeh, N., Huang, T.-H. K., Vanderwende, L., Devlin, J., Galley, M., and Mitchell, M. A survey of current datasets for vision and language research. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 207–213, 2015.
- [49] Fischler, M. A. and Bolles, R. C. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.
- [50] Flusser, J. and Suk, T. A moment-based approach to registration of images with affine geometric distortion. *IEEE transactions on Geoscience and remote sensing*, 32(2):382–387, 1994.
- [51] Gao, X., Li, J., and Shi, Y. A video shot boundary detection algorithm based on feature tracking. In *International Conference on Rough Sets and Knowledge Technology*, pages 651–658, 2006.
- [52] Ghosh, S., Chollet, M., Laksana, E., Morency, L., and Scherer, S. Affect-lm: A neural language model for customizable affective text generation. *CoRR*, abs/1704.06851, 2017.
- [53] Girshick, R. Fast R-CNN. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 1440–1448, 2015.
- [54] Girshick, R., Donahue, J., Darrell, T., and Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

-
- [55] Goldberger, J., Hinton, G. E., Roweis, S. T., and Salakhutdinov, R. R. Neighbourhood components analysis. In *Advances in neural information processing systems*, pages 513–520, 2005.
- [56] Gross, R., Matthews, I., Cohn, J., Kanade, T., and Baker, S. Multi-pie. *Image and Vision Computing*, 28(5):807–813, 2010.
- [57] Gupta, A. and Mannem, P. From image annotation to image description. In *International Conference on Neural Information Processing*, pages 196–204. Springer, 2012.
- [58] Gupta, S., Kim, J., Grauman, K., and Mooney, R. Watch, listen & learn: Co-training on captioned images and videos. In Daelemans, W., Goethals, B., and Morik, K., editors, *Machine Learning and Knowledge Discovery in Databases*, pages 457–472. Springer Berlin Heidelberg, 2008.
- [59] Hadfield, S. and Bowden, R. Hollywood 3D: Recognizing actions in 3D natural scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3398–3405, 2013.
- [60] Hadid, A. and Pietikäinen, M. Combining appearance and motion for face and gender recognition from videos. *Pattern Recognition*, 42(11):2818–2827, 2009.
- [61] Hanjalic, A. Shot-boundary detection: unraveled and resolved? *IEEE transactions on circuits and systems for video technology*, 12(2):90–105, 2002.
- [62] Hazenberg, S. and Hulstun, J. H. Defining a minimal receptive second-language vocabulary for non-native university students: An empirical investigation. *Applied linguistics*, 17(2):145–163, 1996.
- [63] He, K., Zhang, X., Ren, S., and Sun, J. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *European Conference on Computer Vision*, pages 346–361. Springer, 2014.
- [64] Hochreiter, S. and Schmidhuber, J. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

-
- [65] Hodosh, M., Young, P., and Hockenmaier, J. Framing image description as a ranking task: Data, models and evaluation metrics. *Journal of Artificial Intelligence Research*, 47:853–899, 2013.
- [66] Horn, B. and Schunck, B. Determining optical flow. *Artificial Intelligence*, 17(1):185–203, 1981.
- [67] Hsieh, Y., McKeown, D. M., and Perlant, F. Performance evaluation of scene registration and stereo matching for cartographic feature extraction. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 14(2):214–238, 1992.
- [68] Hsu, C.-T. and Tsan, Y.-C. Mosaics of video sequences with moving objects. In *ICIP*, 2001.
- [69] Huang, G. B., Ramesh, M., Berg, T., and Learned-Miller, E. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts, Amherst, 2007.
- [70] Huber, B., McDuff, D., Brockett, C., Galley, M., and Dolan, B. Emotional dialogue generation using image-grounded language models. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, page 277. ACM, 2018.
- [71] Hurley, D. J., Nixon, M. S., and Carter, J. N. Force field feature extraction for ear biometrics. *Computer Vision and Image Understanding*, 98(3):491–512, 2005.
- [72] Hutto, C. and Gilbert, E. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Eighth International Conference on Weblogs and Social Media (ICWSM-14)*. Available at (20/04/16) <http://comp.social.gatech.edu/papers/icwsm14.vader.hutto.pdf>, 2014.
- [73] Irani, M., Anandan, P. a., Bergen, J., Kumar, R., and Hsu, S. Efficient representations of video sequences and their applications. *Signal Processing: Image Communication*, 1996.
- [74] Jia, Y., Salzmann, M., and Darrell, T. Learning cross-modality similarity for

-
- multinomial data. In *2011 International Conference on Computer Vision*, pages 2407–2414. IEEE, 2011.
- [75] Jin, S., Su, H., Stauffer, C., and Learned-Miller, E. End-to-end face detection and cast grouping in movies using erdos-rényi clustering. 2017.
- [76] Johnson, H. J. and Christensen, G. Consistent landmark and intensity-based image registration. *IEEE Transactions on Medical Imaging*, 21(5):450–461, 2002.
- [77] Johnson, J., Karpathy, A., and Fei-Fei, L. Densecap: Fully convolutional localization networks for dense captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4565–4574, 2016.
- [78] Jurie, F. and Dhome, M. Real time robust template matching. In *BMVC*, pages 1–10, 2002.
- [79] Karayil, T., Blandfort, P., Borth, D., and Dengel, A. Generating affective captions using concept and syntax transition networks. In *Proceedings of the 2016 ACM on Multimedia Conference*, pages 1111–1115. ACM, 2016.
- [80] Karpathy, A. and Fei-Fei, L. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3128–3137, 2015.
- [81] Karpathy, A., Joulin, A., and Li, F. F. F. Deep fragment embeddings for bidirectional image sentence mapping. In *Advances in neural information processing systems*, pages 1889–1897, 2014.
- [82] Keshtkar, F. and Inkpen, D. A pattern-based model for generating text to express emotion. In D’Mello, S., Graesser, A., Schuller, B., and Martin, J.-C., editors, *Affective Computing and Intelligent Interaction*, pages 11–21, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.
- [83] Kim, J. and Fessler, J. Intensity-based image registration using robust correlation coefficients. *IEEE transactions on medical imaging*, 23(11):1430–1444, 2004.
- [84] Kiros, R., Salakhutdinov, R., and Zemel, R. S. Unifying visual-semantic embeddings with multimodal neural language models. 2014.

-
- [85] Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.
- [86] Krüger, V., Gross, R., and Baker, S. Appearance-based 3-d face recognition from video. In *Joint Pattern Recognition Symposium*, pages 566–574. Springer, 2002.
- [87] Kučera, H. and Francis, W. N. *Computational analysis of present-day American English*. Dartmouth Publishing Group, 1967.
- [88] Kulkarni, G., Premraj, V., Dhar, S., Li, S., Choi, Y., Berg, A. C., and Berg, T. L. Baby talk: Understanding and generating image descriptions. In *Proceedings of the 24th CVPR*. Citeseer, 2011.
- [89] Kuznetsova, P., Ordonez, V., Berg, A. C., Berg, T. L., and Choi, Y. Collective generation of natural image descriptions. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 359–368. Association for Computational Linguistics, 2012.
- [90] Kuznetsova, P., Ordonez, V., Berg, T. L., and Choi, Y. Treetalk: Composition and compression of trees for image descriptions. *TACL*, 2(10):351–362, 2014.
- [91] Laptev, I., Marszalek, M., Schmid, C., and Rozenfeld, B. Learning realistic human actions from movies. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8, 2008.
- [92] Le, Q. V. and Mikolov, T. Distributed representations of sentences and documents. In *ICML*, volume 14, pages 1188–1196, 2014.
- [93] Le, V., Brandt, J., Lin, Z., Bourdev, L., and Huang, T. Interactive facial feature localization. In Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., and Schmid, C., editors, *Computer Vision – ECCV 2012*, volume 7574 of *Lecture Notes in Computer Science*, pages 679–692. Springer Berlin Heidelberg, 2012.
- [94] LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

-
- [95] Lee, K.-C., Ho, J., Yang, M.-H., and Kriegman, D. Visual tracking and recognition using probabilistic appearance manifolds. *Computer Vision and Image Understanding*, 99(3):303–331, 2005.
- [96] Li, H., Hua, G., Lin, Z., Brandt, J., and Yang, J. Probabilistic elastic matching for pose variant face verification. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 3499–3506. IEEE, 2013.
- [97] Li, S., Kulkarni, G., Berg, T. L., Berg, A. C., and Choi, Y. Composing simple image descriptions using web-scale n-grams. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning, CoNLL ’11*, pages 220–228, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics.
- [98] Lin, C.-Y. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out: Proceedings of the ACL-04 workshop*, volume 8. Barcelona, Spain, 2004.
- [99] Lin, M., Chen, Q., and Yan, S. Network in network. 2013.
- [100] Lin, T., Maire, M., Belongie, S. J., Bourdev, L. D., Girshick, R. B., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. Microsoft COCO: common objects in context. abs/1405.0312, 2014.
- [101] Lindner, C., Bromiley, P. A., Ionita, M. C., and Cootes, T. F. Robust and accurate shape model matching using random forest regression-voting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9):1862–1874, 2015.
- [102] Liu, C., Yuen, J., and Torralba, A. Sift flow: Dense correspondence across scenes and its applications. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33(5):978–994, 2011.
- [103] Liu, X. and Cheng, T. Video-based face recognition using adaptive hidden markov models. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 1, pages I–I. IEEE, 2003.
- [104] Lowe, D. G. Distinctive image features from scale-invariant keypoints. *IJCV*, 2004.

-
- [105] Lucas, B. and Kanade, T. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th international joint conference on Artificial intelligence*, 1981.
- [106] Mao, J., Xu, W., Yang, Y., Wang, J., and Yuille, A. L. Explain images with multimodal recurrent neural networks. 2014.
- [107] Marszalek, M., Laptev, I., and Schmid, C. Actions in context. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 2929–2936, 2009.
- [108] Marter, M., Hadfield, S., and Bowden, R. Friendly faces: Weakly supervised character identification. In *Face and Facial Expression Recognition from Real World Videos workshop at ICPR*, pages 121–132, 2014.
- [109] Mas, J. and Fernandez, G. Video shot boundary detection based on color histogram. *Notebook Papers TRECVID2003*, 15, 2003.
- [110] Matas, J., Zimmermann, K., Svoboda, T., and Hilton, A. Learning efficient linear predictors for motion estimation. In *Computer Vision, Graphics and Image Processing*, pages 445–456, 2006.
- [111] Mathews, A. P., Xie, L., and He, X. Senticap: Generating image descriptions with sentiments. In *AAAI*, pages 3574–3580, 2016.
- [112] Messer, K., Matas, J., Kittler, J., Lüttin, J., and Maitre, G. Xm2vtsdb: The extended m2vts database. In *In Second International Conference on Audio and Video-based Biometric Person Authentication*, pages 72–77, 1999.
- [113] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119, 2013.
- [114] Mitchell, M., Han, X., Dodge, J., Mensch, A., Goyal, A., Berg, A., Yamaguchi, K., Berg, T., Stratos, K., and Daumé III, H. Midge: Generating image descriptions from computer vision detections. In *Proceedings of the 13th Conference of the*

-
- European Chapter of the Association for Computational Linguistics*, pages 747–756. Association for Computational Linguistics, 2012.
- [115] Mitra, S., Savvides, M., and Kumar, B. V. Human face identification from video based on frequency domain asymmetry representation using hidden markov models. In *International Workshop on Multimedia Content Representation, Classification and Security*, pages 26–33. Springer, 2006.
- [116] Mun, J., Cho, M., and Han, B. Text-guided attention model for image captioning. In *AAAI*, pages 4233–4239, 2017.
- [117] Munezero, M. D., Montero, C. S., Sutinen, E., and Pajunen, J. Are they different? affect, feeling, emotion, sentiment, and opinion detection in text. *IEEE Transactions on Affective Computing*, 5(2):101–111, April 2014.
- [118] Na, L. and Nation, I. Factors affecting guessing vocabulary in context. *RELC Journal*, 16(1):33–42, 1985.
- [119] Nguyen, H. V. and Bai, L. Cosine similarity metric learning for face verification. In *Asian conference on computer vision*, pages 709–720. Springer, 2010.
- [120] Ojala, T., Pietikäinen, M., and Harwood, D. A comparative study of texture measures with classification based on featured distributions. *Pattern Recognition*, 29(1):51–59, 1996.
- [121] Ong, E.-J. and Bowden, R. Robust facial feature tracking using shape-constrained multiresolution-selected linear predictors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(9), 2011.
- [122] Ong, E.-J., Lan, Y., Theobald, B., Harvey, R., and Bowden, R. Robust facial feature tracking using selected multi-resolution linear predictors. In *computer vision, 2009 IEEE 12th international conference on*, pages 1483–1490. IEEE, 2009.
- [123] Ordonez, V., Kulkarni, G., and Berg, T. L. Im2text: Describing images using 1 million captioned photographs. In *Neural Information Processing Systems (NIPS)*, pages 1143–1151, 2011.

-
- [124] Ortony, A. and Turner, T. J. What's basic about basic emotions? *Psychological Review*, 97(3):315, 1990.
- [125] Pang, B. and Lee, L. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd annual meeting on association for computational linguistics*, pages 115–124. Association for Computational Linguistics, 2005.
- [126] Panksepp, J. Toward a general psychobiological theory of emotions. *Behavioral and Brain Sciences*, 5(3):407–422, 1982.
- [127] Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics, 2002.
- [128] Parkhi, O. M., Vedaldi, A., and Zisserman, A. Deep face recognition. In *British Machine Vision Conference*, volume 1, page 6, 2015.
- [129] Parkhi, O. M., Simonyan, K., Vedaldi, A., and Zisserman, A. A compact and discriminative face track descriptor. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1693–1700, 2014.
- [130] Patel, N. V. and Sethi, I. K. Compressed video processing for cut detection. *Proc. Vision, Image and Signal Processing*, pages 315–323, 1996.
- [131] Picard, R. W. *Affective computing*. 1995.
- [132] Plutchik, R. Emotions: A general psychoevolutionary theory. *Approaches to emotion*, 1984:197–219, 1984.
- [133] Rahwan, I., Yanardag, P., Cebrian, M., and Obradovich, N. *Nightmare Machine*, 2016. <http://nightmare.mit.edu/>.
- [134] Ramanathan, V., Joulin, A., Liang, P., and Fei-Fei, L. Linking people in videos with “their” names using coreference resolution. In Fleet, D., Pajdla, T., Schiele, B., and Tuytelaars, T., editors, *Computer Vision – ECCV 2014*, volume 8689 of

-
- Lecture Notes in Computer Science*, pages 95–110. Springer International Publishing, 2014.
- [135] Reddy, B. S. and Chatterji, B. N. An fft-based technique for translation, rotation, and scale-invariant image registration. *Image Processing, IEEE Transactions on*, 5(8):1266–1271, 1996.
- [136] Ren, S., He, K., Girshick, R., and Sun, J. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Neural Information Processing Systems (NIPS)*, pages 91–99, 2015.
- [137] Rennie, S. J., Marcheret, E., Mroueh, Y., Ross, J., and Goel, V. Self-critical sequence training for image captioning. In *CVPR*, volume 1, page 3, 2017.
- [138] Roberts, K., Roach, M. A., Johnson, J., Guthrie, J., and Harabagiu, S. M. Em-patweet: Annotating and detecting emotions on twitter. In *LREC*, volume 12, pages 3806–3813. Citeseer, 2012.
- [139] Roth, S. and Black, M. On the spatial statistics of optical flow. *IJCV*, 2007.
- [140] Sagonas, C., Tzimiropoulos, G., Zafeiriou, S., and Pantic, M. 300 faces in-the-wild challenge: The first facial landmark localization challenge. In *Computer Vision Workshops (ICCVW), 2013 IEEE International Conference on*, pages 397–403, 2013.
- [141] Sankar, P., Jawahar, C. V., and Zisserman, A. Subtitle-free movie to script alignment. In *Proc. BMVC*, 2009.
- [142] Schaffalitzky, F. and Zisserman, A. Automated location matching in movies. *Computer Vision and Image Understanding*, 92(2):236–264, 2003. Special Issue on Video Retrieval and Summarization.
- [143] Schroff, F., Kalenichenko, D., and Philbin, J. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015.

-
- [144] Shahraray, B. and Gibbon, D. C. Automatic generation of pictorial transcripts of video programs. In *Multimedia Computing and Networking 1995*, volume 2417, pages 512–519. International Society for Optics and Photonics, 1995.
- [145] Shivhare, S. N. and Khethawat, S. Emotion detection from text. *arXiv preprint arXiv:1205.4944*, 2012.
- [146] Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.
- [147] Sivic, J., Everingham, M., and Zisserman, A. “Who are you?” – learning person specific classifiers from video. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1145–1152, 2009.
- [148] Sivic, J., Everingham, M., and Zisserman, A. Person spotting: video shot retrieval for face sets. In *International Conference on Image and Video Retrieval*, pages 226–236. Springer, 2005.
- [149] Smeaton, A. F., Over, P., and Doherty, A. R. Video shot boundary detection: Seven years of trecvid activity. *Computer Vision and Image Understanding*, 114(4):411–418, 2010.
- [150] Socher, R., Karpathy, A., Le, Q. V., Manning, C. D., and Ng, A. Y. Grounded compositional semantics for finding and describing images with sentences. *Transactions of the Association for Computational Linguistics*, 2:207–218, 2014.
- [151] Socher, R., Lin, C. C., Manning, C., and Ng, A. Y. Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 129–136, 2011.
- [152] Socher, R., Perelygin, A., Wu, J. Y., Chuang, J., Manning, C. D., Ng, A. Y., and Potts, C. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the conference on empirical methods in natural language processing (EMNLP)*, volume 1631, pages 1631–1642. Citeseer, 2013.

-
- [153] Steedly, D., Pal, C., and Szeliski, R. Efficiently registering video into panoramic mosaics. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 2, pages 1300–1307 Vol. 2, 2005.
- [154] Sun, D., Roth, S., Lewis, J., and Black, M. Learning optical flow. In *European Conference on Computer Vision*, pages 83–97. Springer, 2008.
- [155] Sun, Y., Chen, Y., Wang, X., and Tang, X. Deep learning face representation by joint identification-verification. In *Advances in neural information processing systems*, pages 1988–1996, 2014.
- [156] Sun, Y., Wang, X., and Tang, X. Deep learning face representation from predicting 10,000 classes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1891–1898, 2014.
- [157] Sun, Y., Wang, X., and Tang, X. Deeply learned face representations are sparse, selective, and robust. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2892–2900, 2015.
- [158] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.
- [159] Taigman, Y., Yang, M., Ranzato, M., and Wolf, L. Deepface: Closing the gap to human-level performance in face verification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1701–1708, 2014.
- [160] Tao, J. Context based emotion detection from text input. In *Eighth International Conference on Spoken Language Processing*, 2004.
- [161] Tapaswi, M., Bauml, M., and Stiefelwagen, R. “Knock! Knock! Who is it?” Probabilistic Person Identification in TV Series. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2658–2665, 2012.
- [162] Tapaswi, M., Parkhi, O. M., Rahtu, E., Sommerlade, E., Stiefelwagen, R., and Zisserman, A. Total cluster: A person agnostic clustering method for broad-

-
- cast videos. In *Proceedings of the 2014 Indian Conference on Computer Vision Graphics and Image Processing*, page 7. ACM, 2014.
- [163] Tistarelli, M., Bicego, M., and Grosso, E. Dynamic face recognition: From human to machine vision. *Image and Vision Computing*, 27(3):222–232, 2009.
- [164] Tomkins, S. S. Affect theory. *Approaches to emotion*, 163:163–195, 1984.
- [165] Vedantam, R., Lawrence Zitnick, C., and Parikh, D. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4566–4575, 2015.
- [166] Venugopalan, S., Rohrbach, M., Donahue, J., Mooney, R., Darrell, T., and Saenko, K. Sequence to sequence-video to text. In *Proceedings of the IEEE international conference on computer vision*, pages 4534–4542, 2015.
- [167] Vinyals, O., Toshev, A., Bengio, S., and Erhan, D. Show and tell: A neural image caption generator. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3156–3164, 2015.
- [168] Weinberger, K. Q. and Saul, L. K. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10(Feb):207–244, 2009.
- [169] Weinzaepfel, P., Revaud, J., Harchaoui, Z., and Schmid, C. DeepFlow: Large displacement optical flow with deep matching. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1385–1392, 2013.
- [170] Wolf, L., Hassner, T., and Taigman, Y. Similarity scores based on background samples. In *Asian Conference on Computer Vision*, pages 88–97. Springer, 2009.
- [171] Wolf, L. and Levy, N. The svm-minus similarity score for video face recognition. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 3523–3530. IEEE, 2013.
- [172] Wright, J. and Hua, G. Implicit elastic matching with random projections for pose-variant face recognition. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1502–1509. IEEE, 2009.

-
- [173] Xing, E. P., Jordan, M. I., Russell, S. J., and Ng, A. Y. Distance metric learning with application to clustering with side-information. In *Advances in neural information processing systems*, pages 521–528, 2003.
- [174] Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhutdinov, R., Zemel, R., and Bengio, Y. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057, 2015.
- [175] Xu, L., Jia, J., and Matsushita, Y. Motion detail preserving optical flow estimation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(9):1744–1757, 2012.
- [176] Xuehan-Xiong and De la Torre, F. Supervised descent method and its application to face alignment. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 532–539, 2013.
- [177] Yanardag, P., Cebrian, M., and Rahwan, I. *Norman: World’s first psychopath AI*, 2018. <http://norman-ai.mit.edu/>.
- [178] Yanardag, P., Cebrian, M., and Rahwan, I. *Shelley: Human-AI Collaborated horror stories*, 2017. <http://shelley.ai/>.
- [179] Yanardag, P., Rahwan, I., Herranz, M. G., Fabian, C., Rahwan, Z., Obradovich, N., Dubey, A., and Cebrian, M. *Deep Empathy*, 2017. <https://deepempathy.mit.edu/>.
- [180] Yang, Y., Teo, C. L., Daumé III, H., and Aloimonos, Y. Corpus-guided sentence generation of natural images. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 444–454. Association for Computational Linguistics, 2011.
- [181] Yao, B. Z., Yang, X., Lin, L., Lee, M. W., and Zhu, S.-C. I2t: Image parsing to text description. *Proc. IEEE*, 98(8):1485–1508, 2010.
- [182] Yatskar, M., Vanderwende, L., and Zettlemoyer, L. See no evil, say no evil:

-
- Description generation from densely labeled images. *Lexical and Computational Semantics*, page 110, 2014.
- [183] Ye, N. and Sim, T. Towards general motion-based face recognition. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2598–2605. IEEE, 2010.
- [184] You, Q., Jin, H., and Luo, J. Image captioning at will: A versatile scheme for effectively injecting sentiments into image descriptions. *arXiv preprint arXiv:1801.10121*, 2018.
- [185] You, Q., Jin, H., Wang, Z., Fang, C., and Luo, J. Image captioning with semantic attention. 2016.
- [186] Young, P., Lai, A., Hodosh, M., and Hockenmaier, J. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Transactions of the Association for Computational Linguistics*, 2:67–78, 2014.
- [187] Yuan, J., Wang, H., Xiao, L., Zheng, W., Li, J., Lin, F., and Zhang, B. A formal study of shot boundary detection. *IEEE transactions on circuits and systems for video technology*, 17(2):168–186, 2007.
- [188] Zhang, H., Kankanhalli, A., and Smoliar, S. W. Automatic partitioning of full-motion video. *Multimedia systems*, 1993.
- [189] Zhang, N., Paluri, M., Taigman, Y., Fergus, R., and Bourdev, L. Beyond frontal faces: Improving person recognition using multiple cues. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4804–4813, 2015.
- [190] Zhang, Y., Tang, Z., Wu, B., Ji, Q., and Lu, H. A coupled hidden conditional random field model for simultaneous face clustering and naming in videos. *IEEE Transactions on Image Processing*, 25(12):5780–5792, 2016.
- [191] Zhou, H., Huang, M., Zhang, T., Zhu, X., and Liu, B. Emotional chatting

-
- machine: Emotional conversation generation with internal and external memory. *arXiv preprint arXiv:1704.01074*, 2017.
- [192] Zhou, S., Krueger, V., and Chellappa, R. Face recognition from video: A condensation approach. In *Automatic Face and Gesture Recognition, 2002. Proceedings. Fifth IEEE International Conference on*, pages 221–226. IEEE, 2002.
- [193] Zhu, X. and Ramanan, D. Face detection, pose estimation, and landmark localization in the wild. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2879–2886, 2012.
- [194] Zitova, B. and Flusser, J. Image registration methods: a survey. *Image and Vision Computing*, 21(11):977–1000, 2003.