Affective computing and deep learning to perform sentiment analysis

NJ Maree

orcid.org 0000-0003-0031-9188

Dissertation accepted in partial fulfilment of the requirements for the degree *Master of Science in Computer Science* at the North-West University

Supervisor:	Prof L Drevin
Co-supervisor:	Prof JV du Toit
Co-supervisor:	Prof HA Kruger

Graduation October 2020

24991759

Preface

But as for you, be strong and do not give up, for your work will be rewarded.

~ 2 Chronicles 15:7

All the honour and glory to our Heavenly Father without whom none of this would be possible. Thank you for giving me the strength and courage to complete this study.

Secondly, thank you to my supervisor, Prof Lynette Drevin, for all the support to ensure that I remain on track with my studies. Also, thank you for always trying to help me find the golden thread to tie my work together.

To my co-supervisors, Prof Tiny du Toit and Prof Hennie Kruger, I appreciate your pushing me to not only focus on completing my dissertation, but also to participate in local conferences. Furthermore, thank you for giving me critical and constructive feedback throughout this study.

A special thanks to my family who supported and prayed for me throughout my *dissertation blues*. Thank you for motivating me to pursue this path.

Lastly, I would like to extend my appreciation to Dr Isabel Swart for the language editing of the dissertation, and the Telkom Centre of Excellence for providing me with the funds to attend the conferences.

Abstract

Companies often rely on feedback from consumers to make strategic decisions. However, respondents often neglect to provide their honest answers due to issues, such as response and social desirability bias. This may be caused by several external factors, such as having difficulty in accurately expressing their feelings about a subject or having an opinion that is not aligned with the norm of society. Nevertheless, the accuracy of the data from such studies is negatively affected, leading to invalid results. Sentiment analysis has provided a means of delving into the true opinions of customers and consumers based on text documents, such as tweets and Facebook posts. However, these texts can often be ambiguous and without emotion. It may, therefore, be beneficial to incorporate affective computing into this process to gain information from facial expressions relating to the customer's opinion. Another useful tool that may ease this process is deep neural networks. In this study, a method for performing sentiment analysis based on a subject's facial expressions is proposed.

Affective computing is employed to extract meaningful metrics or features from the faces, which is then given as input to a deep multilayer perceptron neural network to classify the corresponding sentiment. Five models were trained, using different data sets to test the validity of this approach. For the first two models, which served as a pilot study, a data set consisting of videos taken of nine participants' faces were used for training and testing purposes. The videos were processed to extract 42 affective metrics which served as input for the first model and six emotions as input for the second models. The results obtained from these two models proved that it was better to make use of the 42 metrics instead of merely the six emotions to train a model to perform sentiment analysis. However, the models may have overfitted due to creating the training, validation and test data sets at frame level. A third model was created by following a similar approach, but by increasing the number of participants to 22 and subdividing the data sets into training, validation and test data sets at video level instead of at frame level. To reduce the influence of human bias on the models, an already existing, pre-annotated data set was used to train for the next models. The data set had to be relabelled to only make use of three distinct sentiment classes. Two ways of doing this were identified; thus, two more models were created. The first variation of the data set had a class imbalance leading to a model with somewhat skewed results. For the second variation, the classes were more evenly distributed, which was reflected in the performance of the model.

The overall results obtained from the study show that the proposed techniques produce models with accuracies that are comparable to models found in the literature, thereby indicating the usability of the proposed techniques. However, it is suggested that other types of neural

networks that process time-series data, such as long-short term memory neural networks, may be used to improve the results even further.

Keywords: affective computing, deep learning, multilayer perceptron, neural networks, sentiment analysis

Opsomming

Maatskappye vertrou gereeld op terugvoer van verbruikers om strategiese besluite te neem. Respondente versuim egter om hul eerlike antwoorde te lewer weens kwessies soos vooroordeel rakende sosiale wenslikheid. Dit kan veroorsaak word deur 'n aantal eksterne faktore, soos dat hulle probleme ondervind om hul eie gevoelens oor 'n onderwerp akkuraat uit te druk of 'n mening te hê wat nie ooreenstem met die norm van die samelewing nie. Nietemin word die akkuraatheid van die data uit sulke studies negatief beïnvloed, wat tot ongeldige resultate lei. Sentimentontleding bied 'n manier om die werklike opinies van kliënte en verbruikers op grond van teksdokumente, soos tweets en Facebook-inskrywings, te ontdek. Hierdie tekste kan egter dikwels dubbelsinnig en sonder emosie wees. Dit kan dus voordelig wees om affektiewe rekenaarverwerking in hierdie proses te inkorporeer om inligting te verkry uit gesigsuitdrukkings rakende die kliënt se mening. ’n Ander nuttige hulpmiddel wat hierdie proses kan vergemaklik, is diep neurale netwerke. In hierdie studie word 'n metode voorgestel om sentimentontleding op grond van die gesigsuitdrukkings van 'n persoon uit te voer.

Affektiewe rekenaarverwerking is gebruik om betekenisvolle statistieke of kenmerke uit die gesigte te onttrek, wat dan gegee word as invoer vir 'n diep multilaag perseptron neurale netwerk om die ooreenstemmende sentiment te klassifiseer. Vyf modelle is ontwikkel met behulp van verskillende datastelle om die geldigheid van hierdie benadering te toets. Vir die eerste twee modelle, wat gedien het as 'n loodsstudie, is 'n datastel wat bestaan het uit video's wat geneem is van nege deelnemers se gesigte, gebruik vir leer- en toetsdoeleindes. Die video's is verwerk om 42 affektiewe statistieke te onttrek wat as invoer gedien het vir die eerste model en ses emosies as invoer vir die tweede model. Die resultate van hierdie twee modelle het gewys dat dit beter was om van die 42 statistieke gebruik te maak in plaas van bloot die ses emosies om 'n model op te lei om sentimentontleding uit te voer. Die modelle kan egter oorleer wees as gevolg van die verdeling van die oefen-, validasie- en toetsdatastelle op datapuntvlak. 'n Derde model is geskep deur 'n soortgelyke benadering te volg, maar die aantal deelnemers is verhoog na 22 en die datastel is onderverdeel in oefen-, validasie- en toetsdatastelle op video-vlak in plaas van datapuntvlak. Om die invloed van menslike vooroordeel op die modelle te verminder, is 'n reeds bestaande voorafgeannoteerde datastel gebruik om die volgende modelle op te lei. Die datastel moes hermerk word om slegs van drie verskillende sentimentklasse gebruik te maak. Twee maniere om dit te doen is egter geïdentifiseer; dus is nog twee modelle geskep. Die eerste variasie van die datastel het 'n klaswanbalans gehad wat daartoe gelei het dat die model skewe resultate het. ’n Beter verspreiding van die klasse van die tweede opstelling is weerspieël in die model se resultate.

Die algehele resultate wat uit die studie verkry is, toon dat die voorgestelde tegnieke modelle lewer met akkuraatheid wat vergelykbaar is met modelle in die literatuur. Hiermee word die bruikbaarheid van die voorgestelde tegnieke aangedui. Daar word egter voorgestel dat ander soorte neurale netwerke wat tydreeksdata verwerk, soos lang-korttermyngeheue neurale netwerke, gebruik kan word om die resultate nog verder te verbeter.

Sleutelwoorde: affektiewe rekenaarverwerking, diep leer, multilaag perseptron, neurale netwerke, sentimentontleding

Conference contributions

Excerpts from the study have been presented at conferences as follows:

Performing visual sentiment analysis using a deep learning approach

N.J. Maree*, L. Drevin, J.V. du Toit, H.A. Kruger

(Abstract presented at the 48th ORSSA Annual Conference, Cape Town, South Africa, 16-19 September 2019)

Affective computing and deep learning to perform sentiment analysis

N.J. Maree*, L. Drevin, J.V. du Toit, H.A. Kruger

(Full paper presented at the SATNAC 2019 Conference, Ballito, South Africa, 1-4 September 2019)

See Annexure D for the full paper.

A Deep Learning Approach to Sentiment Analysis

J.V. du Toit*, N.J. Maree, L. Drevin, H.A. Kruger

(Abstract presented at the 30th European Conference on Operational Research, Dublin, Ireland, 23-26 June 2019)

Affective computing and deep learning to perform sentiment analysis in order to address response bias

N.J. Maree*, L. Drevin, J.V. du Toit, H.A. Kruger

(Abstract presented at the 47th ORSSA Annual Conference, Pretoria, South Africa, 16-19 September 2018)

*Presenting author

TABLE OF CONTENTS

List of abbreviations

AU – Action unit

BNB – Bernoulli naïve Bayesian

CERT – Computer expression recognition toolbox

CLM-Z – 3D constrained local model

CMU-MOSI – Carnegie Mellon University Multimodal Opinion Sentiment Intensity

CMU-MOSEI – Carnegie Mellon University Multimodal Opinion Sentiment and Emotion Intensity

CNN – Convolutional neural network

FACS – Facial action unit coding system

FCP – Facial characteristic point

GAVAM – Generalised adaptive view-based appearance model

HCI – Human-computer interaction

HMM – Hidden Markov model

ICT-MMMO – Institute for Creative Technology's multimodal movie opinion

LBP – Local binary patterns

LR – Logistic regression

LSTM – Long short-term memory

LSVC – Linear support vector classifier

MLP – Multilayer perceptron

MOUD – Multimodal Opinion Utterances Data set

NAS – Neural architecture search

NLP – Natural language processing

nu-SVR – *nu* support vector regression

POS – Parts-of-speech

ReLU – Rectified linear unit

RNN – Recurrent neural network

ROC – Receiver operating characteristics

SDK – Software development kit

SGD – Stochastic gradient descent

SVM – Support vector machine

List of figures

List of tables

List of equations

Chapter 1 Introduction and contextualisation

The secret of getting ahead is getting started

~Mark Twain

1.1. Introduction

When researchers make use of surveys in their studies, their results are highly dependent on the participants providing accurate responses (Gittelman *et al.*, 2015). To ensure that they do not cause the participant to lose interest in participating in the study, thereby causing a bias from the participant, the content within the survey should be carefully designed and developed. Even when the researchers have made provisions to ensure the data that they obtain are of high quality, external factors may still influence the participants. They often find that participants answer the questions within their survey in such a way that it differs from their actual opinions, or behaviours (Larson, 2019). Response bias occurs when answers provided to questions in a survey do not align with the participant's real attitude, behaviour, or other characteristics. This can be due to several reasons, including their answer conflicts with social norms, the participant wants to appear better to others, they want to feel better about themselves, or other external factors (Larson, 2019; Gittelman *et al.*, 2015).

The actions and behaviour of people are often influenced by the opinions of others (Farhadloo & Rolland, 2016). Furthermore, one's worldview, i.e. the way a person perceives reality, is moulded by the manner in which others see and evaluate the world. When faced with making a decision, individuals and organisations alike rely on the opinions of others. Organisations traditionally make use of customer satisfaction questionnaires that are used to gain insight into their customers' attitudes and opinions. However, a high cost is often associated with this means of gaining insight. Sometimes they do not have the resources needed to conduct such research. Since it is essential to the image of an organisation and revenue generation, it is necessary that they can get access to these insights in another way. Therefore, other methods that do not have a high cost, which are not resource-intensive, and can preferably be automated should be explored.

One such way that may be advantageous to both individuals and organisations is sentiment analysis (Farhadloo & Rolland, 2016). Sentiment analysis refers to methods used to detect and extract subjective information from a source (Mäntylä *et al.*, 2018). This information is usually in the form of attitudes, or opinions, and can be classified based on whether the person's opinion

about something is positive, neutral, or negative. The traditional focus of sentiment analysis is to extract information from sources consisting of natural language, i.e. text-based sources. However, social media has enabled people to voice their opinions using other data modalities which cannot be analysed using traditional sentiment analysis techniques based on natural language processing (NLP) (Poria *et al.*, 2016a). These modalities, including videos, images and audio files, such as podcasts, open new avenues for research that can be explored. Using videos as opposed to textual sources for sentiment analysis has the advantage of providing behavioural cues and facial expressions that can be used to identify the true emotions that the person in the video is feeling (Poria *et al.*, 2017a). However, with the use of videos, the problem of extracting the needed information from the videos arises.

People often struggle to explain or label their feelings. The field of affective computing can help with the identification of human emotions since it entails computing that "relates to, arises from, or deliberately influences emotions" (Picard, 1999). Affective computing makes use of facial expressions, gestures, posture, and other physiological cues to detect and interpret the users' current emotional state (Politou *et al.*, 2017). Sentiment analysis can incorporate affective computing in order to identify sentiments in images and videos, as most techniques used in the field of sentiment analysis are based on NLP. It was further found that little research has been done on sentiment analysis with affective computing based on facial expressions.

In this research project, the problem of performing sentiment analysis using videos was investigated. The role of affective computing to identify underlying emotions and facial expressions within videos that can provide insight into a participant's true opinions was explored. For conducting this research, a deep neural network trained with affective data features is proposed to analyse the sentiment expressed in videos (as opposed to text).

The remainder of this chapter is structured as follows. In Section 1.2, the research question is formulated, followed by the primary goal and secondary objectives of the study in Section 1.3. The goal of Section 1.4 is to provide a brief discussion on the research design, specifically the research paradigm and methods to be used. The ethical considerations that were taken into account during the performance of this study are briefly discussed, and the assigned ethics number is provided in Section 1.5. A summary of the purpose of each of the chapters is presented in Section 1.6. Finally, in Section 1.7, a summary of the information presented within this chapter is provided.

1.2. Research question

This study is conducted to perform sentiment analysis using affective computing specifically for the identification of emotions, based on a person's facial expressions. The extracted emotions were then presented as input to a deep artificial neural network to model sentiment. Consequently, the research question that is investigated can be stated as "How can affective computing be used with deep learning techniques to perform sentiment analysis on videos to mitigate feedback problems, such as response bias?".

1.3. Research goals

The primary aim of this research is to determine whether sentiment analysis can effectively be performed by using a deep learning model trained on metrics obtained, using affective computing techniques to mitigate general feedback problems. Therefore, it is proposed that a deep multilayer perceptron (MLP) neural network should be developed and trained, using a data set consisting of affective features for performing sentiment analysis.

The following secondary objectives have been set to facilitate the achievement of the primary aim:

1. Construct, or find a suitable data set consisting of video recordings of people's faces and annotate with the expressed sentiment;
2. Extract useful features relating to emotions from the video data set, using affective computing techniques;
3. Develop a deep MLP for classifying affective data into one of three sentiment categories, i.e. positive, neutral, or negative; and
4. Determine how well the deep MLP performs by evaluating it in terms of accuracy, error rate, precision, recall and F-score.

1.4. Research design

An overview of the research design, including the research paradigm and methods, is provided within this section.

1.4.1. Research paradigm

Paradigms refer to a set of shared assumptions about some aspect of the world (Oates, 2005). Each paradigm has its own ontological and epistemological assumptions, which describe what constitutes reality, and what one can define as knowledge, respectively (Gray, 2014). This study is conducted using the positivistic paradigm, as it follows an experimental design approach, and

data was analysed in a quantitative form. Moreover, the researcher must stay objective and uninvolved during the data acquisition phase, as it may influence the results of the study.

Researchers within the positivistic research paradigm view the world as physically and socially objective (Paré, 2004). It is also stated that the nature of the world can be perceived, characterised, and measured relatively easily. This paradigm is based on the epistemological assumption that *a priori* relationships exist within phenomena which can be identified and tested using hypothetico-deductive reasoning and analysis. Furthermore, researchers view themselves as being impartial observers, enabling them to evaluate actions and processes objectively. This paradigm has the following characteristics (Oates, 2005):

- The world, which exists apart from the human mind, can be examined, observed, and captured.
- New discoveries are made through modelling the world through observations and measurements.
- To prevent the researchers' views and values from influencing the outcome of a study, they must not participate during observation and must remain objective and impartial.
- Theories generated using the positivistic paradigm should be extensively tested to confirm or refute these theories.
- To analyse the results of a study objectively, some researchers prefer to make use of mathematical models and proofs.

While conducting this study, the applicable characteristics mentioned above were taken into consideration.

1.4.2. Research methods

This study follows an experimental design to construct a computational model for performing sentiment analysis. Initially, a single experiment was planned for this purpose. However, due to issues relating to the data sets, the number of experiments was increased to five. The data sets for the first three experiments that will be discussed were collected, using a systematic approach, that involved recording participants' facial expressions while they viewed media on a computer and then extracting the affective data in quantitative form using the Affectiva© emotion software development kit[1]. Participants in the data collection phase consisted of second-year, third-year, honours and master's students studying Computer Science and Information Systems. The third data set, used for the last two experiments, is the Carnegie Mellon

[1]https://knowledge.affectiva.com/docs/getting-started-with-the-emotion-sdk-for-windows#section-1-download-and-run-the-sdk-installer

University Multimodal Opinion Sentiment Intensity (CMU-MOSI) data set consisting of 2 199 segmented videos, which is popularly used in multimodal sentiment analysis studies (Zadeh, 2016b).

The data sets include metrics measuring the likelihood that a participant is experiencing certain feelings, such as joy, frustration or anger, as well as other metrics more specific to the way the participant contorts his or her face, such as a brow raise or a frown. There are 42 metrics in total that were extracted at around 0.033-second intervals, or 30 hertz, of the recorded videos. The affective data was then used to construct and train a deep MLP neural network. The purpose of the MLP was to classify the data into one of three sentiment categories, i.e. positive, neutral, or negative.

1.5. Ethical considerations

The research proposal was presented to the Faculty of Natural and Agricultural Sciences' ethics committee for ethical clearance. The study was approved as a minimal risk study and the ethics number NWU-00150-19-A9 was issued on 25 February 2019. The following ethical considerations were adhered to during the execution of the study:

- Ensure the anonymity and privacy of the participants in the study;
- prevent participants from feeling vulnerable by partaking in the study;
- only collect data from participants who provide their consent;
- allow the participants to withdraw their collected data at any time;
- do not expose participants to sensitive or controversial topics and content; and
- safely store the collected data on an off-line device.

In the event that any other ethical concerns arise during the execution of the study, the appropriate ethics committee will be contacted before proceeding. It should also be noted that the CMU-MOSI data set is available online (Zadeh, 2018) in the form of a software development kit (SDK) and may be used by anyone for research purposes with proper citation of the research paper by Zadeh *et al.* (2018a).

1.6. Outline of chapters

In this section, an overview of the remaining chapters is provided by briefly describing each chapter's purpose.

A literature review on sentiment analysis is presented in Chapter 2, defining sentiment analysis, followed by a discussion on techniques for performing sentiment analysis, and how it can be used in conjunction with affective computing.

Chapter 3 is devoted to exploring the concept of affective computing, how emotions, also known as affects, can be recognised, and the uses thereof in the existing literature.

In Chapter 4, background information on deep multilayer perceptron neural networks is provided. Firstly, a background discussion on the origins and functionality of machine learning is presented, followed by a discussion on deep learning and MLP neural networks.

The focus in Chapter 5 is to present the experimental design of the study, including the data acquisition techniques used, as well as the development and validation of the deep MLP neural networks. It further explores the results obtained from the experimental phase, along with an evaluation of the performance of the MLPs. A discussion of the insights gained from the results concludes this chapter.

In conclusion, a summary of the results and the insights acquired from the study, and an indication of how the research goals, as stipulated in Section 1.3, have been met is provided in Chapter 6. Additionally, the limitations and future work relating to this study are presented.

1.7. Summary

With the increase of opinions uploaded to the Internet in the form of videos, the need for analysing and identifying sentiment has arisen. Thus, the problem under investigation for this research is the use of videos as a data source for performing sentiment analysis. The intended approach is to extract features relating to emotions and facial expressions by using affective computing and training a deep MLP neural network for classifying the expressed sentiment.

In this chapter, the aim was to introduce the research problem, question, and goals. Additionally, the proposed methods and ethical considerations that had to be taken into account were discussed. Finally, an overview of each chapter was provided. In the next chapter, a literature review on sentiment analysis is provided.

Chapter 2 Sentiment analysis

Opinion is the medium between knowledge and ignorance

~Plato

2.1. Introduction

Sentiment analysis, also known as opinion mining, is traditionally a form of subjectivity analysis which aims to detect the contextual polarity of text (Micu *et al.*, 2017). The opinion expressed within the text can be grouped into three sentiment categories, i.e. a positive, negative, or neutral opinion. Sentiment refers to one's personal experience or one's attitude or feeling towards something (Farhadloo & Rolland, 2016). Human behaviour is often influenced by the opinions of others and therefore forms a central part of almost all human activities. However, organisations are also influenced by opinions, as they rely on the opinions of their customers when making decisions. With the rise of social media, large volumes of data have been made available in the public domain, which is challenging to sort through using manual processes. By implementing sentiment analysis techniques, organisations can obtain new insights into their brand awareness and their customers' behavioural patterns. Marketers can make use of these insights to provide their organisations with a competitive advantage. Rizwana and Kalpana (2018) state that organisations can make use of sentiment analysis for various other reasons. These include measuring return on investment, improving product quality and customer service, crisis management, lead generation, and increasing sales revenue. The concept of sentiment analysis and its applications will be explored within this chapter.

The structure of the remainder of this chapter is organised in the following manner. Section 2.2 defines the fundamental terms and tasks used within sentiment analysis literature. The purpose of Section 2.3 is to differentiate between the various levels of sentiment analysis techniques. A shift within the field has occurred and the use of other types of media, such as audio and visual sources, has become a focus of research. The aim of this study is to perform sentiment analysis by employing a deep neural network and affective computing techniques. In Section 2.4 and Section 2.5, respectively, related work for sentiment analysis in general and those studies that made use of affective computing are presented. A summary of the topics discussed in this chapter is provided in the concluding section, i.e. Section 2.6.

2.2. Basic concepts

The aim of sentiment analysis is to discover the underlying opinion expressed by a person. In this section, the aim is to provide an overview of the fundamentals of sentiment analysis by discussing different types of opinion, as well as the general tasks associated with sentiment analysis.

2.2.1. Defining opinion

According to Farhadloo and Rolland (2016), sentiment analysis is used to identify four components of a sentiment or opinion, i.e. the entity, the aspect, the opinion holder, and the aspect's opinion orientation. An additional component is also added when an opinion is being analysed to indicate the time at which the opinion was made. These five components are referred to as an opinion quintuple. The entity, also referred to as the target or object, is a product, service, person, event, organisation, or topic that has a hierarchy of components and attributes, known as the aspects of the entity (Liu, 2017; Liu & Zhang, 2012). The opinion holder, or contributor, is usually the author of the source being analysed, but the author may also refer to another person's opinion. An opinion contains two components: the entity and the sentiment on the target, or opinion orientation, expressed as positive, negative, neutral, or may even be expressed using different intensity levels.

Opinions can be classified along different dimensions, i.e. regular and comparative opinions, subjective and fact-implied opinions, first-person and non-first-person opinions, and meta-opinion.

2.2.1.1. Regular and comparative opinions

A regular opinion is a positive or negative attitude, emotion, or appraisal that an opinion holder has about an entity or an aspect of the entity, and has two subtypes, namely direct and indirect opinions. A direct opinion refers to an opinion that is directly expressed about an entity or an aspect of an entity, e.g. "The quality of the picture is excellent". An indirect opinion is indirectly expressed on an entity or entity aspect, based on a positive or negative effect on another entity, e.g. "This new computer allows me to finish my work, which used to take me five hours, in one hour". Conversely, a comparative opinion is an expression of the similarities or differences between multiple entities. It may also include the opinion holder's preference, based on some shared aspects of the entities, e.g. "Coke tastes better than Pepsi".

2.2.1.2. Subjective and fact-implied opinions

Though all opinions are inherently subjective because they express a person's subjective view, they do not always appear as subjective sentences within opinionated documents. People often make use of factual sentences to express their feelings, since facts can be desirable or undesirable. Therefore, opinions can be either subjective or fact-implied. Subjective opinions are regular or comparative opinions stated in a subjective manner, e.g. "I think Google's profit will go up next month". A fact-implied opinion is also a regular or comparative opinion implied in a factual manner, which expresses a desirable or undesirable fact. Two subtypes of fact-implied opinion exist, i.e. personal and non-personal fact-implied opinions. The former type refers to opinions expressed by a factual statement based on a person's personal experience, e.g. "I bought the mattress a week ago, and a valley has formed in the middle". In contrast, a non-personal fact-implied opinion does not express any personal opinion, experience, or evaluation and merely reports a fact, e.g. "Google's revenue went up by 30%".

2.2.1.3. First-person and non-first-person opinions

Sometimes it is necessary to differentiate between statements that express the opinions of the person self or that express the beliefs of a person regarding the opinions of others. These opinions can be either classified as first-person opinions or non-first-person opinions. A first-person opinion states the opinion holder's attitude towards an entity, e.g. "I think Google's profit will go up next month". When a person states someone else's opinion, it is known as a non-first-person opinion, e.g. "I think John likes Lenovo PCs".

2.2.1.4. Meta-opinions

The last class of opinions, referred to as meta-opinions, is opinions about opinions. The target of such an opinion is also an opinion and is usually contained in a subordinate clause. The subordinate clause can either state a fact with an implied opinion or a subjective opinion, e.g. "I am very happy that my daughter loves her new Ford car".

2.2.2. Sentiment analysis tasks

Taking the definition of an opinion, as well as the four components of which it is made up as stated in the previous section into account, the objective of sentiment analysis can be formally stated as follows (Liu, 2017; Farhadloo & Rolland, 2016):

> Given an opinion document, or collection of reviews, $D = \{d_1, d_2, \ldots, d_D\}$ all about an object, discover all the corresponding opinion quintuples expressed in that collection. Optionally, discover the reason and qualifier of the sentiment in each opinion quintuple, for more advanced analysis.

Sentiment analysis can be broken up into the following six main tasks to discover the opinion quintuples from the opinion document:

In task one, the aim is to extract and categorise the entity. This task finds and extracts all entity expressions within the collection D. If synonymous entities exist within the collection, they should be grouped together in order to indicate a unique entity. Similarly, the second task extracts and categorises all aspects of each entity to express unique aspects. During the execution of the third task, the unique opinion holders within the text are found and categorised. Note that there may be more than one opinion holder mentioned in the text. The time at which the opinion was made is extracted and standardised in the fourth task. Task five's purpose is to discover the opinion orientation regarding each of the aspects identified in the second task. Lastly, produce all the opinion quintuples for each document $d \in D$, based on the results from the above tasks, during the sixth task.

Two optional tasks can be added to address the second, more advanced, part of the objective of sentiment analysis, namely to identify the reason and qualifier of the sentiment: Firstly, find the reason why each opinion was expressed, and cluster these reasons according to each aspect or entity and opinion orientation. Then, discover and group qualifier expressions for each opinion according to each aspect or entity and each opinion orientation.

Note that the reason and qualifier are not always specified within the opinionated document. To illustrate the eight tasks discussed above, consider the following blog example and analysis results:

> Posted by: bigJohn Date: Sept. 15, 2011
>
> I bought a Samsung camera and my friends bought a Canon camera last Saturday. In the past week, we both used the cameras a lot. The photos from my Samy are not that great, and the battery life is short too. My friend was very happy with his camera and loves its picture quality. I want a camera that can take good photos. I am going to return it tomorrow.

The first task identifies three entities mentioned in the text, i.e. "Samsung", "Samy", and "Canon". It should further group "Samsung" and "Samy" together, since they refer to the same entity. During the second task, it should identify the following aspects, i.e. "picture", "photo", and "battery life", as well as their corresponding entities. Once again "picture" and "photo" are synonymous and should be grouped together. "bigJohn" and his friend are extracted from sentences three and four as the opinion holders, respectively, in the third task. The next task recognises that the statement was posted on 15 September 2011. The fifth task finds that the

opinion expressed about the Samsung camera's picture and battery life is negative, whilst the opinion about the Canon's picture, as well as the overall experience are positive. Finally, the last task is to generate the appropriate opinion quintuples expressed in the text. Task six generates the following four quintuples based on the above-mentioned results:

1. (Samsung, picture, negative, bigJohn, Sept-15-2011)
2. (Samsung, battery_life, negative, bigJohn, Sept-15-2011)
3. (Samsung, GENERAL[2], positive, bigJohn's_friend, Sept-15-2011)
4. (Samsung, picture, positive, bigJohn's_friend, Sept-15-2011)

By applying more advanced mining and analysis techniques, the reasons and qualifiers can also be found for each of the quintuples:

1. Reason for opinion: picture not clear
   Qualifier of opinion: night shots
2. Reason for opinion: short battery life
   Qualifier of opinion: unspecified
3. Reason for opinion: unspecified
   Qualifier of opinion: unspecified
4. Reason for opinion: unspecified
   Qualifier of opinion: unspecified

Through this simple example, the execution and expected results of the tasks of sentiment analysis are illustrated.

2.2.3. Sentiment identification

Although sentiment analysis consists of multiple tasks, the fifth task, i.e. the identification of the sentiment, can be viewed as the most important and needs to be explained further. The sentiment can be expressed in different forms, such as a rating scale where a numerical value is assigned to the aspect, or it can be classified into different categories, for example, positive, negative, and neutral (Farhadloo & Rolland, 2016). The process of identifying the sentiment can be done in three steps:

1. Extract the opinionated fragments: use parts-of-speech (POS) tags and syntactic structures, along with opinionated words and phrases to identify the fragments that are most likely opinionated.

[2] GENERAL is indicated in all-caps as it refers to the overall experience and not to an aspect extracted from the opinionated document.

2. For each opinionated fragment, identify the sentiment: make use of supervised, e.g. Naïve Bayes and Support Vector Machines (SVM), unsupervised methods, or by making use of lexicons for determining the polarity of each individual fragment.
3. Deduce the collective sentiment of all fragments: find the overarching sentiment expressed by the separate fragments.

Traditionally, there are two categories into which sentiment identification can be grouped, i.e. lexicon-based approaches, and machine learning approaches (Vyrva, 2016). An additional category can be added which is known as hybrid methods (Catal & Nangir, 2017). Lexicon-based methods determine the sentiment expressed within a text based on the total sum of positive and negative words contained in it (Abdi *et al.*, 2019). For this purpose, a sentiment lexicon is used, which is a set of words with positive and negative values assigned to them. With a machine learning approach, feature extraction needs to be done before a pre-trained classifier, such as an SVM or Naïve Bayes classifier, can perform sentiment analysis. Hybrid methods combine lexicon-based and machine learning methods to improve the accuracy of a text classification system. The strengths and weaknesses of the above-mentioned approaches are listed in Table 2.1 (Devika *et al.*, 2016; D'Andrea *et al.*, 2015).

2.3. Levels of sentiment analysis

The objective of sentiment analysis can be achieved at different levels of focus (Farhadloo & Rolland, 2016). In this section, the aim is to provide an overview of four different levels of sentiment analysis, i.e. document level, sentence level, aspect level, and entity level analysis.

2.3.1. Document level

The goal of sentiment analysis at document level is to determine the overall polarity or opinion orientation of an opinionated document (Zhang *et al.*, 2018). Pawar *et al.* (2016) state that document-level sentiment analysis can be formally defined as:

> Consider a document that contains opinionated sentences regarding an entity, then identify the orientation of the opinion expressed towards the entity. This can be presented through the opinion quintuple, where the feature is equivalent to the object, and the opinion holder, time, and entity is known.

It should be further noted that the general assumption is made that the opinionated document contains only an opinion related to a single targeted entity. Performing sentiment analysis at document level can be challenging, as not every sentence in the document may be opinionated, and therefore does not contribute to the sentiment of the document (Xu *et al.*, 2016).

Table 2.1: Strengths and weaknesses of existing approaches (Devika *et al.*, 2016; D'Andrea *et al.*, 2015)

Approaches	**Features/Techniques**	**Advantages**	**Disadvantages**
Machine learning	• Unigrams or n-grams, along with their occurrence frequency • Parts-of-speech information used for disambiguation and feature selection • Negations can potentially reverse the sentiment of words, e.g. not working	• Ability to adapt • Models can be trained for specific contexts and purposes. • No dictionary is necessary. • High accuracy of classification	• Cost to develop and label data set • Labelled data need to be available before techniques can be applied to new data • Models are usually domain-specific and will not work in another domain
Lexicon-based	• Manual construction of a lexicon • A corpus-based approach to produce opinion words • A dictionary-based approach using a small set of known opinion words and growing it by finding their synonyms and antonyms	• Wider term coverage • Labelled data and learning procedure are not required	• Lexicons contain only a finite number of words • A fixed sentiment orientation is assigned to words within the lexicon. • Powerful linguistic resources are needed

Table 2.1: Strengths and weaknesses of existing approaches (continued)

Approaches	Features/Techniques	Advantages	Disadvantages
Hybrid	• Sentiment lexicon constructed from public resources for ignition sentiment detection • Sentiment words as features in machine learning methods	• Lexicon/learning symbiosis • Lesser sensitivity to changes in the topic domain • Detection of sentiment at concept level	• Noisy reviews are often assigned a neutral score

2.3.2. Sentence level

Instead of assigning a single sentiment, or opinion quintuple, to an entire opinionated document, it is possible to perform sentiment analysis at a finer grain, i.e. sentence level. Thus, the sentiment of a single sentence is calculated (Zhang *et al.*, 2018). However, it should be noted that the assigned value is not merely the sum of the polarity of its constituent words (Mohammad, 2017). This task can be done through subjectivity classification and polarity classification. The first determines whether the sentence's content is subjective or objective, while the second is used to determine the opinion orientation of a subjective sentence. Syntactic and semantic information, such as part-of-speech tags and parse trees, can additionally be used to help with the task of classifying sentiment within sentences, as they are typically shorter than documents. It is not as important to know the polarity of a single sentence as it is to know the polarity of a specific aspect or feature of an entity (Varghese & Jayasree, 2013).

2.3.3. Aspect level

Sentiment analysis can also be performed at aspect level, meaning that specific aspects which the opinion holder is addressing are extracted and assigned a sentiment (Farhadloo & Rolland, 2016). This task is typically more challenging to perform than at document and sentence level (Beigi *et al.*, 2016). Aspect extraction can be done in one of two ways: automatic, or (semi) manual extraction. With automatic aspect extraction, the aspects within the opinionated document are unknown beforehand and should be automatically extracted, using supervised or unsupervised methods. Manual extraction of aspects requires *a priori* knowledge of which aspects need to be extracted, after which it can be extracted. Next, the sentiment of the aspect

needs to be identified, where the fragments contain references to the aspects identified in the previous step.

2.3.4. Utterance level

With the advent of video-based sentiment analysis, a new level has emerged, namely utterance level (Poria *et al.*, 2017b). An utterance is a speech unit that is bounded by breaths and pauses. Thus, when conducting sentiment analysis at this level, the aim is to tag each utterance in a video with the appropriate sentiment label. This is opposed to assigning a single label to the entire video. It provides the advantage of understanding the dynamics of the speaker's sentiment towards different aspects of the topic throughout his dialogue.

2.4. Survey on text-based sentiment analysis

Most work done in the field of sentiment analysis has focused on natural language (NLP) techniques to identify the opinion within textual documents. Studies within the field of NLP often make use of a multilevel representation learning approach to simulate the functions of the brain, known as deep learning (Abdi *et al.*, 2019; Sun *et al.*, 2017). These methods make use of discrete features automatically extracted from the text, such as word, phrase, and syntactic features. A recurrent neural network (RNN) is a widespread technique employed to solve NLP problems, as it can handle input sequences of variable length. Deep learning models show to be effective when addressing the task of sentiment analysis when trained using word vector representations (Araque *et al.*, 2017). This approach can be further augmented by including information from other sources, such as visual or audio information.

Jianqiang *et al.* (2018) proposed using word embeddings that were obtained using an unsupervised approach. For their experiments, they made use of five different data sets consisting of tweets from Twitter. They made use of word *N*-gram features, the polarity of words, vector representations of words and features related specifically to Twitter, e.g. number of hashtags as the input to a deep convolutional neural network (CNN). The resulting accuracy varied between 80.61% and 88% for each of the data sets. Compared to an SVM using a bag-of-words model, their proposed model more effectively identifies the context sentiment information, reduces data sparsity, and retains information about the word order.

Feng *et al.* (2018) proposed a deep learning approach to perform sentiment analysis consisting of two steps. The first step applied a deep CNN to words in sentences. Word vectors containing each word in the sentence, as well as the three words preceding and the three words following it, were used to build the feature vector. Furthermore, the parts-of-speech vectors and syntax vectors that were extracted from the word vectors were appended to the feature vector. In the

second step, the feature vector was given as input to a deep CNN. This experiment resulted in an accuracy of 87.58%.

A multiphase, hierarchical feature engineering methodology was developed by Ghiassi and Lee (2018) to extract a reusable feature set of Twitter data that can be used over multiple domains. To demonstrate the efficiency of their technique, they made use of an SVM and a dynamic architecture for neural networks (DAN2) applied to four different domains. Four classes were also identified for classification, i.e. strongly positive, mildly positive, mildly negative and strongly negative. They used vector representations of the tweets from each data set, along with a sentiment class label as input to the models. Accuracy rates ranging between 63.86% and 94.64% were obtained with DAN2 on the test data set. The SVM approach achieved a maximum accuracy of 93.5% and a minimum of 62%.

Symeonidis *et al.* (2018) performed sentiment analysis using two data sets, namely the sentiment strength Twitter (SS-Twitter) data set (which was re-annotated to only include three sentiment labels, i.e. positive, neutral, and negative) and the SemEval data set (SemEval-2019, 2019). They selected four machine learning and deep learning algorithms that are popularly applied to the problem, specifically Logistic Regression (LR), Bernoulli Naïve Bayes (BNB), Linear Support Vector Classifier (LSVC) and a CNN. Furthermore, 16 pre-processing techniques were applied to the data sets before being used in each of these algorithms. In Table 2.2, the resulting accuracies from their study are shown.

Another approach to sentiment analysis using deep learning models is to study the compositionality of the opinionated document. For this task, a recursive neural tensor network model can be used, as proposed by Socher *et al.* (2013) which outperformed other models trained for binary and fine-grained sentiment analysis. It makes use of word vectors and parse trees and computes the vectors for nodes higher up in the tree with a tensor-based composition function. Araque *et al.* (2017) proposed multiple techniques, using an ensemble of analysers in conjunction with a combination of manually crafted features and automatically extracted embedding features. They found that by combining models from different sources can improve their baseline model.

Uysal and Murphey (2017) performed a comparative study between sentiment analysis approaches based on feature selection and deep learning methods. For the former, three techniques were applied to select features, namely information gain (Uğuz, 2011), Gini index (Shang *et al.*, 2007) and a distinguishing feature selector (Uysal & Gunal, 2012). The latter method consisted of three deep learning models, i.e. CNN, long short-term memory (LSTM) and a long-term CNN, i.e. a hybrid of the previous two methods. To implement all these techniques,

four data sets were used. The deep learning models outperformed the feature selection techniques for all but one data set, where the information gains featured expanded with word embedding features outperformed all other methods.

Table 2.2: Accuracies (%) obtained by Symeonidis *et al*. (2018)

	SS-Twitter				SemEval			
Technique	LR	BNB	LSVC	CNN	LR	BNB	LSVC	CNN
0 (Baseline)	60.6	57.9	60.8	65.6	65.2	62.7	66.4	66.1
1	60.9	58.6	60.4	64.0	65.3	63.4	66.4	64.5
2	60.3	58.2	60.4	62.6	64.7	63.0	66.7	61.5
3	61.4	58.4	61.0	63.9	65.3	63.0	66.7	64.8
4	60.9	58.0	61.0	59.1	65.3	62.8	66.7	65.2
5	60.7	58.6	61.2	65.7	65.4	62.9	66.9	65.4
6	60.8	57.9	60.7	63.0	65.2	62.6	66.2	66.5
7	57.8	55.8	57.3	55.5	65.2	62.2	65.4	62.3
8	60.2	57.7	60.3	66.1	65.2	62.8	66.4	67.3
9	60.6	58.0	60.7	65.2	64.5	61.5	64.9	63.4
10	59.9	56.4	59.6	64.4	65.5	62.9	66.6	65.6
11	60.6	58.4	61.4	64.8	64.1	62.0	64.3	64.9
12	59.3	58.5	58.6	60.1	65.1	61.5	64.1	61.7
13	59.5	57.3	57.9	62.6	65.4	63.5	65.7	63.6
14	60.9	58.7	61.0	64.1	65.4	63.0	65.7	63.9
15	61.7	60.6	60.9	63.6	65.1	62.6	65.5	62.2
16	61.0	58.8	61.0	66.9	65.1	62.0	65.9	64.9

Kumar and Jaiswal (2017) extracted around 3 000 tweets from Twitter and 3 000 tumblogs from Tumblr to evaluate the capabilities and scope of sentiment analysis within these sources. The data was assessed, using six supervised techniques, i.e. Naïve Bayesian, SVM, multilayer perceptron neural network, decision tree, k-nearest neighbour, and fuzzy logic in Weka. The SVM outperformed the other techniques for both the data sets. Their findings furthermore suggest that results from Tumblr may be more improved and optimised as opposed to the data collected from Twitter.

2.5. Survey on multimodal sentiment analysis

Emotions play a significant role in humans' decision-making process, as it is an intrinsic component of their mental activity (Cambria *et al.*, 2019). The detection and interpretation of emotions using computing models are essential to specific fields of computer science, such as human-computer interaction, e-learning, security, user profiling, and sentiment analysis. The fields of affective computing and sentiment analysis both relate to the identification, interpretation, and generation of human emotions (Han *et al.*, 2019). As previously stated, sentiment analysis aims to identify opinions within the longer term, whilst affective computing tries to identify emotional states instantaneously. Though emotions and sentiment are closely related and often used interchangeably, there are some differences between them (Tian *et al.*, 2018). Specifically, sentiments are usually more directed towards a specific entity and are more stable and dispositional than emotions.

However, most sentiment analysis approaches that incorporate affective computing entail using text-based methods. With the rise of social media and advances within technology, consumers no longer only make use of text to voice their opinions about an entity (Poria *et al.*, 2017a). Reviews of products and services are increasingly uploaded to the Internet in the form of videos, which consumers make using their web cameras. Analysing these videos has the advantage of detecting behavioural cues and emotions from the reviewer which would not be detectable using only text. Studying sentiment analysis by taking other modalities where emotion can be detected into account, is part of the field of multimodal sentiment analysis.

The pioneering work done in the field of multimodal sentiment analysis was that by Morency *et al.* (2011). They drew inspiration from the field of audio-visual emotion detection and tried to combine the concept with text-based sentiment analysis. For their research, they compiled a data set of 47 YouTube videos that included real-world videos with a group of diverse people and ambient noises. They made use of three annotators to decide on the correct label for each video. Overall the annotators agreed in 78.7% of all the cases, and none of the videos had a complete disagreement. The features extracted from the videos were processed using a Hidden Markov Model (HMM). The text-only HMM achieved a 43.1% accuracy; the visual-only achieved 44.9% and the audio-only 40.8%. Nevertheless, when these modalities were combined, it achieved 54.3%, indicating the potential strength of exploring other modalities for performing sentiment analysis.

A study done in 2018 focused on making use of facial expressions and body gestures to identify the sentiment expressed within videos (Junior & dos Santos, 2018). Histogram of Oriented Gradients (HOG) was used to extract faces from the videos in the Carnegie Mellon University Multimodal Opinion Sentiment Intensity (CMU-MOSI or sometimes simply MOSI), Multimodal

Opinion Utterances Data set (MOUD) and YouTube data sets. At the same time, the body gestures are identified using a combination of HOG and Motion History Imaging. Before these features are provided as input to a linear kernel version SVM, it is passed through principal component analysis to reduce the dimensionality of the features. The features are also firstly encoded with the Fisher Vector (FV) and Vector of Locally Aggregated Descriptors (VLAD) methods. The accuracies for the different encodings for each of the data sets are summarised in Table 2.3. These results show a much higher accuracy than obtained in previous related studies where the visual modality was also under investigation. However, most of the mentioned studies' main focus was on combining the modalities of text, audio and visuals, most having a minimal emphasis on the latter. In contrast, Junior and Dos Santos (2018) focused solely on this modality. Thus, their study's results could show much higher accuracies.

Table 2.3: Summarised results of Junior and Dos Santos (2018)

Method	**CMU-MOSI**	**MOUD**	**YouTube Data set**
VLAD	77%	**93%**	57%
FV	**92%**	76%	**77%**

Tian *et al.* (2018) studied the application of multitask learning to both unimodal, and multimodal sentiment analysis, using the CMU-MOSI database (Zadeh *et al.*, 2018b). They also found that the model trained on verbal (textual) data performed the best, followed by vocal data, and lastly the visual data. Furthermore, the models trained using multitask learning performed better for each of the data modes.

Poria *et al.* (2017c) performed multimodal sentiment analysis fusing audio, visual and textual data together by using the data set developed by Morency *et al.* (2011). Audio features, such as pitch and voice intensity, were extracted using the openSMILE feature extraction tool[3] at a 30Hz frame rate. The facial recognition library 3D Constrained Local Model (CLM-Z) (Baltrušaitis *et al.*, 2012) was used to extract facial characteristic points (FCPs) along with facial expression features obtained using the generalised adaptive view-based appearance model (GAVAM) (Saragih *et al.*, 2009). Lastly, the textual features were generated using a convolutional neural network (CNN). Each of these modalities was then separately given as input to a support vector machine. Their proposed solution outperformed that of Morency *et al.* (2011) by reaching an accuracy of 74.49%, 75.22%, and 79.14%, for each of the respective modalities. This once again showed that textual data delivered the best performance. It should be noted that all three

[3] https://www.audeering.com/opensmile/

of Poria *et al.*'s (2017c) unimodal methods outperformed the solutions proposed in other studies.

In another study done by Poria *et al.* (2017b), a multimodal approach using long short-term memory (LSTM) neural networks with a data set consisting of videos at utterance level was proposed. They made use of a CNN and word2vec to extract textual features from the transcripts of the videos and openSMILE for audio features. They obtained visual features from a 3-D CNN. They further made use of MOUD, CMU-MOSI data sets to perform sentiment analysis. When all three modalities were combined, they achieved the highest accuracy, i.e. 80.3% and 68.11% for each of the respective data sets. However, in terms of the unimodal LSTMs, the text-based model had the highest accuracy at 78.12% and 52.17% correspondingly. The visual neural network scored the lowest with 55.8% on the CMU-MOSI data set and 48.58% on the MOUD data set.

Poria *et al.* (2016b) also proposed the use of a temporal CNN with multiple kernel learning, where images at time t and $t + 1$ were combined into a single image. This model was further combined with an RNN to improve the results. Both the MOUD and the Interactive Emotional Dyadic Motion Capture (IEMOCAP) data sets were utilised. The authors used the CLM to find the outline of each frame's face, which was then used as input to their model. Once more, openSMILE was used for audio feature extraction and a CNN was used to extract textual data. Controversially, it was decided to translate the original transcripts of the MOUD data set from Spanish to English. The results obtained from the IEMOCAP data set showed a different result from most other studies in terms of the accuracy of the visual analysis. In this study, the visual modality performed far better than both the textual and audio modalities.

Nu-support vector regression (*nu*-SVR) models, where *nu* refers to a parameter that replaces the epsilon parameter in an *epsilon*-SVR, trained on the CMU-MOSI data set, were studied by Zadeh *et al.* (2016a). The models made use of five-fold cross-validation, and the performance of the regressor was calculated with a mean absolute error and correlation. Their approach was similar to other approaches mentioned above, as they also trained different models, using distinct modalities, as well as in combination with each other. In this case, they only focused on visual and transcripts of the dialogue. Once again, the text-based model outperformed the visual model, with correlation values of 0.46 and 0.36, respectively. When combined, the correlation increased to 0.53.

A data set consisting of segments of American news programs recorded between 13 August 2013, and 25 December 2013, was created by Ellis *et al.* (2014). They extracted audio features with openSMILE, similar to Poria *et al.* (2017c) and Poria *et al.* (2017b). For the visual features,

they focused on extracting the sentiment from the face that is busy speaking, as news programs often split the screen showing two faces. Dense Local Binary Patterns (LBP) were extracted from the identified face, resulting in 100 59-dimensional LBP histograms for each facial image. Other visual features used for training the SVM were obtained, using a Block-Based-Bag-of-Words. The results for the visual sentiment classification were extremely low, ranging from 31.47% to 44.41% with the different feature sets when all the speakers were combined. Nevertheless, the authors found an interesting pattern within the data set: when a model was created for each separate speaker, the models performed much better, with the best accuracy being 85.71%. This suggests that each speaker has unique patterns when expressing his/her sentiments. Another unexpected result was that the text classification also performed rather poorly compared to some of the other studies, as well as against their own audio- and visual-based models, suggesting that transcripts within the news domain are somewhat ambiguous.

Pérez-Rosas *et al.* (2013) performed multi-modal sentiment analysis at utterance level by using a manually collected data set consisting of 80 Spanish product opinions from YouTube. They used linguistic features in the form of a bag-of-words representation, 28 acoustic and 40 visual features in combinations of one, two or all three modalities, as input to an SVM. The goal was to classify each utterance as positive or negative, ignoring the neutral class. It obtained an accuracy of 70.94%, 64.85% and 67.31% for each distinct modality, respectively.

The Institute for Creative Technology's Multi-Modal Movie Opinion (ICT-MMMO) data set was created by Wöllmer *et al.* (2013) from social media and contained videos of people reviewing movies. The data set consists of 370 review videos of individual people speaking towards a camera. To detect sentiment within this data set, they made use of a bi-directional LSTM for classification, based on audio and visual features, and an SVM for linguistic features. The models trained on the separate modalities obtained 64.4% and 61.2% accuracy for audio and visual. The accuracy ranged from 59.6% to 73.0% for the models trained on the linguistic features as they made use of multiple feature sets for this model.

Other studies worth mentioning on performing sentiment analysis on videos include Sun *et al.* (2016) who made use of a deep CNN to identify the affective regions on still images in order to predict sentiment. This study, however, focused on the sentiment evoked by said images, and not the emotional expressions of people within them. Wang *et al.* (2017) compared a CNN and select-additive learning (SAL) CNN, using three data sets containing multimodal data. They found that SAL methods improve the generalisability of models across all three modalities. Zadeh *et al.* (2017a) showed that their solution to perform sentiment analysis with a tensor fusion network delivered state-of-the-art performance compared to other known solutions. Specifically, their model based on visual data was compared to that of Kahou *et al.* (2015) and

Byeon and Kwak (2014). Sharma *et al.* (2018) extracted some features from a Twitter image data set. The features were then fed to a CNN model to be classified as positive, negative, or neutral. Their model, consisting of three convolutional layers and a max pool layer, achieved an accuracy of 67.77%.

2.6. Summary

From the literature discussed above, it can be concluded that multimodal sentiment analysis can be effectively used to determine a person's sentiment. By detecting and combining affective information through different modalities, the task of sentiment analysis can be significantly improved. Nevertheless, studies on the unimodal analysis of each distinct modality can be of benefit to this field, as the current accuracy of models based on a single modality is much lower than when combined and can be improved. Techniques used to identify sentiment expressed in text or transcriptions of the speech were significantly higher than in the other modalities. This can be ascribed to NLP-based analysis that has been studied for far longer than speech and facial expression analysis to detect sentiment. This study, therefore, focuses on the visual modality by analysing affects related to the face. It can also be concluded that very little work has been done using deep learning approaches; therefore, its use for visual sentiment analysis at frame level will be investigated in this study.

The aim of sentiment analysis is to discover the opinion of a person stated in an opinionated document. Traditionally, it is done using natural language processing techniques, but recently other modalities have been studied. In this chapter, the context around sentiment analysis covering the fundamental concepts and tasks related to it, as well as related literature were presented. In the next chapter, affective computing to identify emotions and related features based on the facial expressions of a person will be discussed.

Chapter 3 Affective computing using facial expressions

The question is not whether intelligent machines can have any emotions, but whether machines can be intelligent without any emotions.

~Marvin Minsky

3.1. Introduction

Emotions function as an aid to humans in everyday decision making, learning, communication and situation awareness in their environments (Poria *et al.*, 2017a). However, computers were not designed to be aware of human emotions. Only in the past two to three decades, researchers have been exploring the possibility of giving computers the ability to recognise, interpret and express emotions. Affective computing, as coined by Picard in 1995, is used to describe the branch of artificial intelligence concerned with designing computer systems and devices that have the ability of detecting, interpreting, communicating, responding to, and eliciting human emotions (Schwark, 2015; Picard, 1995). It spans the fields of computer science, psychology, social science and cognitive science. Understanding emotions is also key to the process of emulating human intelligence (Cambria *et al.*, 2017). The task of emotion detection is also closely related to polarity detection; thus, it can be used to extract people's sentiments from online social data.

The aim in this chapter is to provide background on this field where emotions and computers intertwine, specifically on the use of facial expression analysis. Affective computing is explored, whereafter background on emotions is provided in Section 3.2 by giving a definition of emotions and briefly discussing models of emotion. Humans can perceive and express their emotions in a variety of ways, including through facial expressions, speech, body language, and other physiological reactions. In Section 3.3, affect detection, based on facial features, is presented, along with relevant literature on the topic. Finally, the chapter's content is summarised in Section 3.4.

3.2. Background on emotions and affective computing

In this section, the aim is to define emotion, also referred to as affects, and to introduce popular models of emotion found in the literature. The section is concluded by providing background on the use of emotions with computer systems.

3.2.1. Defining emotion

Emotions are part of humans' everyday lives and communication, yet there seems to be no universally accepted definition for this term. One such attempt at defining it states that emotions are generally short-lived, intense internal mental states that represent a valenced or evaluative reaction to external stimuli (Oliver *et al.*, 2020). Ekkekakis and Zenko (2016) provide another definition that involves both the cognitive appraisal, as well as the physiological response. According to them, emotional states arise due to the appraisal of a situation that may influence the individual's well-being. It further leads to an affective reaction in the form of a physiological signature or behavioural expression pattern, as well as corresponding actions and coping efforts. Oliver *et al.* (2020) identify five components present in all emotions, i.e. a cognitive evaluation of a situation, arousal, a subjective feeling state, the motivation, and a motor expression.

Theories on human emotion can be subdivided into three classes based on their cause, namely physiological, neurological and cognitive (Sreeja & Mahalakshmi, 2017). Physiological emotion theories, such as the James-Lange theory (James, 1884), suggest that emotions have their origin from within the human body itself because of physiological changes. Neurological-based theories claim that emotional responses are triggered by activity within the brain. Emotion theories, for example, the Schachter-Singer theory (Schachter & Singer, 1962), based on cognitive studies, propose that emotions are formed by thoughts and other mental activities.

3.2.2. Models of emotion

There are typically two models used to describe emotions, i.e. categorical and dimensional (Mennig *et al.*, 2019; Sreeja & Mahalakshmi, 2017). In the former model, emotions are grouped into distinct labelled categories from which the emotion that best describes the conveyed feeling should be selected. In other words, the emotion is assigned a label. Studies surrounding this model were pioneered by Darwin (1872), suggesting that emotions are universal among humans and animals, as they express emotions through similar behaviour.

This claim was further supported by Ekman and Friesen (1971) who studied the universality of emotions amongst people of different cultures. They stated that emotions could generally easily be recognised by observing the facial expressions of the subject, even if the individuals' cultures differ. Ekman (1999) later found that there are six basic universally distinctive emotions, namely anger, fear, enjoyment (or happiness), sadness, disgust and surprise, shown in Figure 3.1 as used in his research. However, Ekman admitted that other emotions might exist apart from the ones he identified. These six emotions have formed the basis of most research in the field of affective computing. Matsumoto (1992) added a seventh emotion, contempt, to this model

because it did not include a facial expression to describe the expression made when someone shows disrespect to another.

Figure 3.1 Six basic emotions proposed by Ekman (1999)

Conversely, emotions represented by dimensional emotional models use a set of quantitative measures based on multidimensional scaling (Mennig *et al.*, 2019; Sreeja & Mahalakshmi, 2017). Two- or three-dimensional spaces are often used to describe and position the emotions. Measurements that are typically used in a two-dimensional space are valence and arousal, for example, the Circumplex Model of Affect proposed by Russell (1980), as shown in Figure 3.2, whereas a three-dimensional space adds power as a third measurement, such as the Pleasure, Arousal, Dominance (PAD) emotional state model (Mehrabian & Russell, 1974). Valence refers to how positive emotions are; arousal represents the excitement associated with the emotions and power indicates the influence or sense of control the person has over the emotion. The advantage that dimensional models hold is that they can capture emotions at a much finer level of granularity.

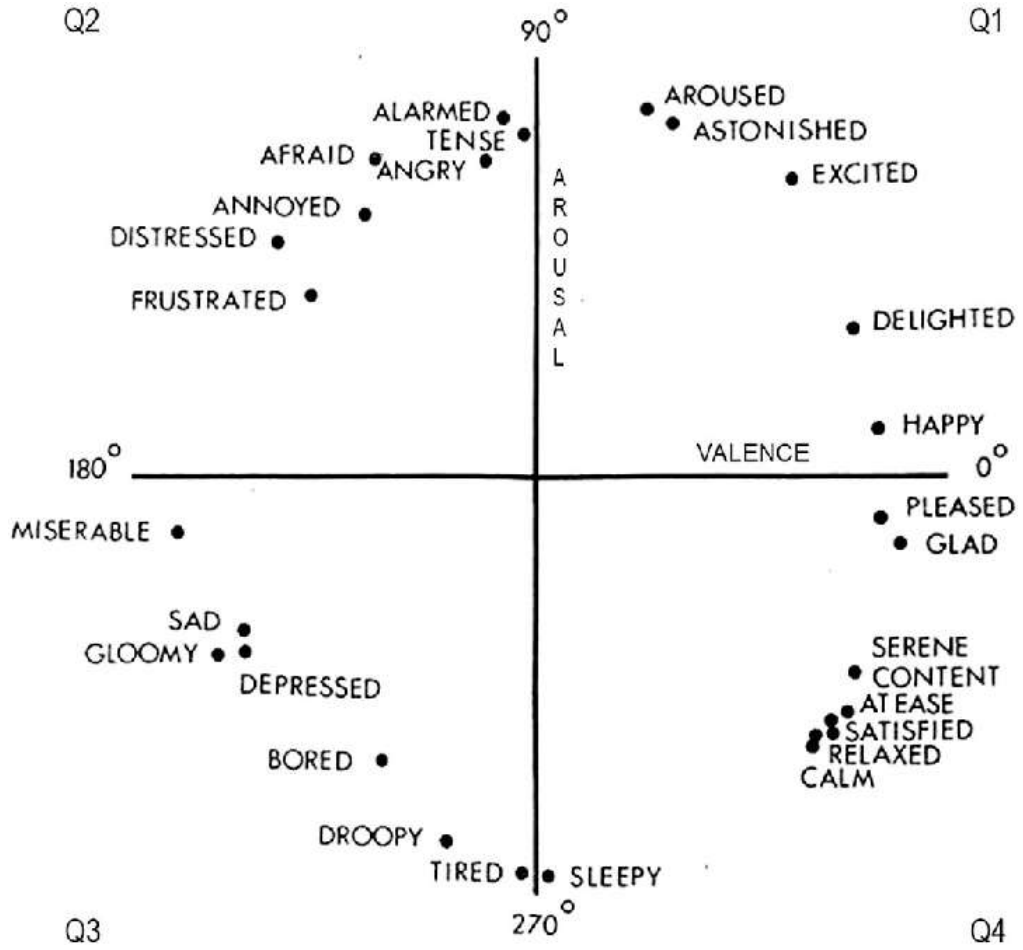


Figure 3.2: Russell's circumplex model of affect with 28 affect words (Russell, 1980)

These models are not always mutually exclusive, as in the case of the Wheel of Emotion proposed by Plutchik (1980). A visualisation of this model is shown in Figure 3.3. Similar to the categorical model proposed by Ekman, Plutchik proposes that there are eight basic emotions, i.e. anger, anticipation, joy, trust, fear, surprise, sadness and disgust. It suggested that any emotion that can be expressed is related to one of the eight emotions. Moreover, each one of these emotions has a polar opposite included in the list. Joy can be considered as opposed to sadness, fear as opposed to anger, anticipation and surprise, as well as disgust as the opposite of trust. These emotions can also be combined to form other emotions, e.g. joy and trust can be combined to form love, and optimism is formed by joining joy and anticipation. Lastly, each emotion can be experienced at different intensities according to where the middle of the circle represents high intensities and the outer layer shows lower intensities. As an example, surprise can range from distraction (low intensity) to amazement (high intensity).

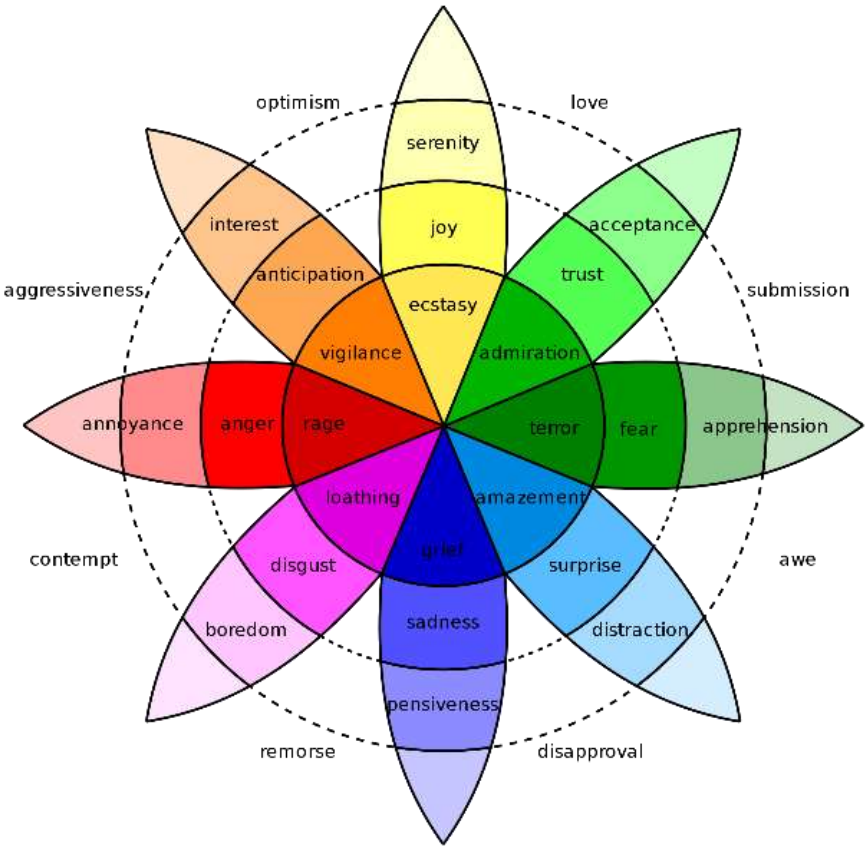


Figure 3.3: Plutchik's wheel of emotion (Plutchik, 1980)

3.2.3. Emotions and computer systems

The field of designing and developing emotion-aware computer systems, known as affective computing, has only been under study since 1995 (D'Mello *et al.*, 2018; Picard, 1995). Much of the work done in this field draws from affective science, as the research done on emotions and its influence on humans forms the basis for affective computing. Furthermore, it is supported by the fields of digital signal processing and machine learning. Research within the field spans across theories of how affects can influence the interaction between humans and technology, improving the understanding of affect detection and generation, as well as designing, implementing and evaluating affect-aware systems (Calvo *et al.*, 2015).

Yannakakis (2018) suggests a four-phase process cycle for systems that collect affective information and adapts to it. During the first phase, emotions are elicited from the user or participant. This can be done using elements, such as videos, sounds, pictures or games. Some form of a bodily response will follow the elicitation which can be detected and modelled. Equipment, such as a camera, microphone, gaze tracker or other physiological sensors, can be used to capture the reactions. The collected data should then be transformed to reduce noise

and identify specific attributes that are of importance. Once this is done, labels can be assigned to the data so that the data can be modelled, using machine learning techniques, such as supervised learning. In the final phase, the affective loop is closed by the system adapting to the state of the user or performing some function or procedure based on the detected affective state. In the next section, the application of affective computing for emotion detection using facial expressions is discussed.

3.3. Facial expression analysis

The face plays an important role in conveying a person's emotions in a social context (Girard *et al.*, 2015). Facial expression analysis has many applications, ranging from cars that can detect drowsiness to an emotional response in marketing and smile detection in consumer cameras. In this section, the focus is on presenting the generic process followed by facial expression analysis systems and existing emotion detection systems utilising facial expressions. Some of the criticism and challenges within the literature against such systems are also presented.

3.3.1. Processing facial expressions

Most facial expression recognition systems follow the same generic processing sequence, but variations may exist (Valstar, 2015; Jiang *et al.*, 2014). The face is firstly detected and registered, or pre-processed, followed by feature extraction and classification or regression of the expression.

Since images and videos may contain noisy data which should not be taken into consideration during the feature extraction phase, the subject's face should be isolated from the image. A popular approach to this problem is the Viola-Jones face detector proposed by Viola and Jones (2001) because of its speed and reliability. However, merely detecting a subject's face within an image will not be sufficient to move on to extracting features. Faces in images can differ due to dynamic differences, such as the head's orientation and illumination, as well as static differences between different sexes and ethnographies. These differences should not be taken into account when analysing the facial expressions. Therefore, during the face registration phase, the identified face is scaled and rotated to remove the differences that do not relate to the facial expressions of the subject.

The feature extraction phase is crucial to the successful analysis of facial expressions, though, in theory, classifications can be made without this phase. By extracting features, pixel data is transformed into a high-level representation of the shape, motion, colours and textures, thereby reducing the dimensionality of the problem and ignoring irrelevant aspects. If the previous phase failed to correct certain aspects of the image, such as face alignment and illuminations,

this phase might provide robustness against it. Feature extraction can be approached by using either geometry- or appearance-based methods. Geometry-based approaches make use of measurements of the differences in the position of certain parts of the face, such as eyebrows (Joy & Prasad, 2016). Facial expressions can, therefore, be determined based on the movement of these different facial points. Appearance-based methods try to identify changes in the texture of the face, such as wrinkles, bulges, and other changes that can be caused by facial movements.

The last phase of the facial expression analysis process is to classify the facial expression based on the feature representation, using machine learning techniques, such as supervised and unsupervised learning. These models can be trained to classify the features in terms of discrete emotions, Facial Action Coding System (FACS) action units (AUs) or even dimensional affects.

3.3.2. Survey on facial emotion detection

Facial expressions can be used to gain insight into a person's present state of mind and understand his/her emotions and sentiments (Poria *et al.*, 2017a). Chen *et al.* (2018) state that two streams of research on facial expression analysis exist. The first is oriented towards identifying facial actions, whilst the second directly tries to identify emotion without taking these facial actions into account. Emotions can be measured based on the way a person moves his/her face (Stöckli *et al.*, 2018). This can be done by manually coding the facial expressions through computer-based classification algorithms or facial electromyography activity. The latter method refers to measuring the electrical changes within the facial muscles by using biosensors placed on the face. It is, however, not able to clearly classify distinct emotions with this method. Therefore, only manual and computer-based techniques will be presented.

According to McDuff *et al.* (2019), Chen *et al.* (2018) and Jiang *et al.* (2014), FACS is currently the best-known system that is widely used to manually code humans' facial actions. The system was developed by Ekman and Friesen (1976) and was improved upon in 2002 (Ekman & Friesen, 2002). They identified 27 atomic facial muscle movements or actions, referred to as action units (AU) which can be combined to describe every possible facial expression a human can have, but does not infer emotions. Nine of these AUs are in the upper face and 18 in the lower face. The FACS' AU codes, along with their descriptors and muscle groups involved with each action, are listed in Table 3.1. Additionally, 14 descriptors for measuring head position and movements, as well as nine for eye position and movements are included in the system, along with 28 other descriptors. Examples of further, more grossly defined action units are contained within Table 3.2.

Table 3.1: Main action units (Ekman & Friesen, 1976)

AU number	Descriptor	Muscular basis
1	Inner brow raiser	Frontalis, Pars Medialis
2	Outer brow raiser	Frontalis, Pars Lateralis
4	Brow lowerer	Depressor Glabellae, Depressor Supercilli, Corrugator
5	Upper lid raiser	Levator Palpebrae Superioris
6	Cheek raiser	Orbicularis Oculi, Pars Orbitalis
7	Lid tightener	Orbicularis Oculi, Pars Palebralis
9	Nose wrinkler	Levator Labii Superioris, Alaeque Nasi
10	Upper lip raiser	Levator Labii Superioris, Caput Infraorbitalis
11	Nasolabial fold deepener	Zygomatic Minor
12	Lip corner puller	Zygomatic Major
13	Cheek puffer	Caninus
14	Dimpler	Buccinator
15	Lip corner depressor	Triangularis
16	Lower lip depressor	Depressor Labii
17	Chin raiser	Mentalis
18	Lip puckerer	Incisivii Labii Superioris, Incisivii Labii Inferioris
20	Lip stretcher	Risorius
22	Lip funneler	Orbicularis Oris
23	Lip tightener	Orbicularis Oris
24	Lip pressor	Orbicularis Oris
25	Lips part	Depressor Labii, or Relaxation of Mentalis or Orbicularis Oris
26	Jaw drop	Masetter, Temporal and Internal Pterygoid Relaxed
27	Mouth stretch	Pterygoids, Digastric
28	Lip suck	Orbicularis Oris

Table 3.2: Examples of more grossly defined action units (Ekman & Friesen, 1976)

AU number	Descriptor
8	Lips toward each other
19	Tongue out
21	Neck tightener
29	Jaw thrust
30	Jaw sideways
31	Jaw clencher
32	Lip bite
33	Blow
34	Puff
35	Cheek suck
36	Tongue bulge
37	Lip wipe
38	Nostril dilator
39	Nostril compressor
43	Eyes closure
45	Blink
46	Wink

McDuff *et al.* (2019) created a data set known as AM-FED+, which is an improvement on their original AM-FED data set, as presented in McDuff *et al.* (2013). It consists of 1 044 webcam videos containing facial responses, which were recorded “in-the-wild” over the internet of which 545 were manually coded for 11 of the FACS AUs. In addition, 34 automatically detected facial landmark points (shown in Figure 3.4), the baseline performance of detection algorithms for eight actions and self-reported responses of the familiarity and liking of the viewed videos are included in the data set. The authors provided two sets of baseline performance metrics for automated AU recognition using the AM-FED+ data set. The first made use of support vector machine (SVM) classifiers with approximated radial basis function kernels. Images collected in other studies were used to train the classifiers and were person-independent. For the testing phase, the AM-FED+ data set was employed. It obtained an average receiver operating characteristic of 0.766, indicating that facial actions can be accurately detected, but that it still may be improved. The second baseline used a standard open-source facial analysis toolbox, OpenFace. This toolbox has been pre-trained on another data set, therefore, McDuff *et al.*’s data set was only used for testing. Once again, their results suggested that there is still room to improve.

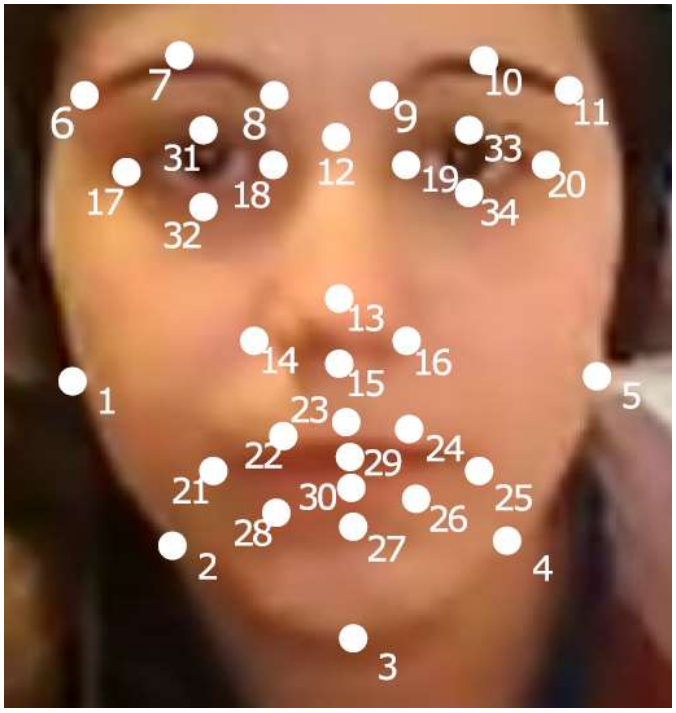


Figure 3.4: Location of automatically detected facial landmarks (McDuff *et al.,* 2019)

OpenFace 2.0 is a computer vision and machine learning tool for facial behaviour analysis (Baltrušaitis *et al.*, 2018), and is an extension to the original OpenFace toolkit proposed by Baltrušaitis *et al.* (2016). One of the aims of the OpenFace team is to provide an open-source tool that is freely available to researchers. The authors claimed that their solution extends state-of-the-art facial expression detection algorithms and that it is capable of real-time performance without relying on a GPU. It is further capable of detecting and tracking facial landmarks, estimating head pose and eye gaze, as well as classifying facial expressions. Facial landmark detection and tracking were done using a custom implementation of the Convolutional Experts Constrained Local Model proposed by Zadeh *et al.* (2017b) and the Multi-task Convolutional Neural Network for face detection by Zhang *et al.* (2016). The latter method provides the ability to detect profile or highly occluded faces. Furthermore, a convolutional neural network (CNN) was implemented to prevent tracking from drifting by reporting whether it has failed based on the detected landmarks. Head pose estimation is done by solving the perspective *n*-point problem proposed by Hesch and Roumeliotis (2011) once the facial landmarks are detected. A Constrained Local Neural Field landmark detector (Baltrusaitis *et al.*, 2013) was implemented to estimate eye gaze. Finally, facial expression recognition is done using a linear kernel SVM using AUs.

An approach to captioning images based on the emotions of the human faces in them was proposed by Nezami *et al.* (2018). A facial expression recognition model has been developed to extract features from images containing human faces. Firstly, the faces were detected using a convolutional neural network face detection algorithm, after which it was converted to a grayscale image and resized. They then trained a VGGNet[4] model to extract the probabilities of each emotion from the faces. Lastly, a long short-term memory network was used to generate the captions for each image.

In 2013, Kahou *et al.* (2013) submitted a winning technique to the 2013 Emotion Recognition in the Wild Challenge, which they later extended (Kahou *et al.*, 2016). The latter paper discussed the approach in more detail and presented newer results. They were tasked with assigning one of seven emotion labels to a set of short video clips. A multimodal method was followed using deep learning techniques. For visual features, a CNN was used, whilst the audio stream was handled by a deep belief network. Furthermore, a *K*-means bag-of-mouth model was used to extract visual features around the mouth, and a relational auto-encoder for addressing spatio-temporal aspects in the video was employed. Different strategies were followed to combine the separate models' results. The first involved averaging the predictions together, which yielded an accuracy of 37.17%. They also aggregated the results, using an SVM and multilayer perceptron (MLP), resulting in an accuracy of 42.17%. However, they obtained their highest performance at 47.98% when using a random search for weighting models.

McDuff *et al.* (2016) proposed a system for automated facial coding consisting of four main components, i.e. face and facial landmark detection, face texture feature extraction, facial action classification and emotion expression modelling. The system makes use of the Viola-Jones face detection algorithm, after which 34 facial landmarks are identified inside each facial bounding box. Bounding boxes that have a confidence level less than a threshold are ignored. Each image region is then used to extract histogram of oriented gradient features. An SVM is then trained on 10 000 manually coded images with a corresponding score of 0-100 for each facial action. Finally, seven distinct emotions, i.e. disgust, fear, joy, sadness, surprise, anger and contempt, are modelled, based on the EMFACS system (Friesen & Ekman, 1983), which is an abbreviated version of FACS that only focuses on the muscles associated with emotional expressions. This system proposed by McDuff *et al.* is known as the AFFDEX software development kit (SDK) and is owned by Affectiva©.

[4] The VGGNet was invented by the Visual Geometry Group (VGG) from the University of Oxford. It received the second place for the ImageNet Large Scale Visual Recognition Competition 2014 in the classification task.

Yu and Zhang (2015) aimed to classify static images into the seven basic emotion categories. Their proposed method consisted of an ensemble of techniques, i.e. the joint cascade detection and alignment detector (Chen *et al.*, 2014), the deep CNN-based detector (Zhang & Zhang, 2014) and Mixtures of Trees (Zhu & Ramanan, 2012) to detect faces within the images. The faces were then resized and converted to grayscale. Finally, the facial expressions were classified using an ensemble of CNNs.

The Computer Expression Recognition Toolbox (CERT) was introduced by Littlewort *et al.* (2011), claiming it to be the first publicly-available fully automatic real-time system that recognises FACS AUs with state-of-the-art accuracy. It can detect 3-D head orientation, the intensity of smiles, ten facial feature points, the six universal emotions and 19 FAC AUs. The Viola-Jones algorithm is used to detect faces with a boosting algorithm, GentleBoost (Friedman *et al.*, 2000) and WaldBoost (Sochman & Matas, 2005), for automatic cascade threshold selection. Next, ten facial features are detected which are then used during the phase registration phase to re-estimate the face patch, which is in turn transformed into a feature vector by convolving it. The feature vector presented as input to a separate linear SVM for each AU and the intensities thereof are given as frame-by-frame output.

3.3.3. Criticism and challenges

Barrett *et al.* (2019) identified three shortcomings in the scientific research of how emotions can be perceived based on a person's facial expressions, i.e. limited reliability, lack of specificity and limited generalisability. The first pertains to the fact that there is no common set of facial movements which can be reliably used to distinguish emotions of the same category. A lack of specificity refers to a specific configuration of facial movements or actions that cannot be uniquely mapped to an exact emotion. In terms of generalisability, they claim that insufficient research has been done to document the effects of context and culture on emotions. Martinez (2019) holds that the interpretation of facial expressions without information on the surrounding context can decrease the accuracy of emotion recognition. Further problems include biasing results due to limitations in the selection process of the stimulus to evoke a specific emotion and the focus of research is usually based on English emotion categories with little focus on emotions that are described in other languages.

According to Schwark (2015), most affective computing models make use of stereotypical and acted emotional responses that may not be similar to how a person would typically express his/her emotions. These systems also often neglect to take personal and cultural differences into account; thus, it can result in a system that does not adapt well enough to make a significant impact on the user. He further quotes Picard (2003) who raised two criticisms of

affective computing. With the former, she stated that there is a vast number of modalities available for emotion expression and that some of them cannot be accessed or measured effectively and others are too non-differentiated. This view is shared by Lee and Norman (2016) who claim that such systems should include various sensors to detect these modalities. The second criticism pertains to the variability of emotions from person to person. Different people express their emotions differently; it can therefore be challenging to identify the emotional state of an individual accurately.

3.4. Summary

Facial expressions are generally used by humans to express their inner feelings to the outside world. Though humans experience emotions every day, it proves difficult to define. Hence, a general definition found in literature was provided, along with an overview of different models that may be used to describe emotions. Affects can be detected by computer systems using sensors, such as cameras; however, these systems still need to interpret or classify them in some way. Thus, a general introduction into the field of affective computing with regard to emotion recognition using facial expressions was presented.

For this study, the Affectiva© emotion software development kit (SDK) will be used to identify affects based on the six emotions identified by Ekman (1999) and the AUs (Ekman & Friesen, 2002). In the following chapter, artificial neural networks with a focus on deep neural networks will be examined.

Chapter 4 Deep multilayer perceptron neural networks

A baby learns to crawl, walk and then run. We are in the crawling stage when it comes to applying machine learning.

~Dave Waters

4.1. Introduction

Artificial neural networks are structures consisting of “densely interconnected adaptive simple processing elements”, known as nodes or neurons that can perform large parallel computations suitable for knowledge representation and data processing (Basheer & Hajmeer, 2000). These structures are ideal for solving complex problems, due to capabilities inspired by biological neural networks, such as fault and failure tolerance, high parallelism, robustness, non-linearity, learning, and ability to generalise.

In this chapter, an overview of deep learning concepts is provided. Before considering the concept of deep learning, artificial neural networks, upon which deep learning models are built, are introduced. Firstly, the biological inspiration for artificial neural networks is discussed in Section 4.2. This is followed by an explanation of the structure and functioning of artificial neural networks in Section 4.3. A brief discussion of deep learning is provided in Section 4.4 by providing definitions for it, describing the different categories of deep learning, and discussing deep multilayer perceptron (MLP) neural networks. The automated architecture search strategy, known as neural architecture search, is presented in Section 4.5 followed by a description of some performance measures which can be used to evaluate the efficiency of deep neural networks in Section 4.6. The chapter is concluded in Section 4.7, with a summary of the work discussed below.

4.2. Biological origins

Research in artificial neural networks draws its inspiration from biological neural networks, as can be found in the brain of both humans and animals. Therefore, it is necessary to have a basic understanding of how these structures work before considering artificial neural networks. The human brain consists of roughly 10^{11}, or 100 billion neurons, and each of these has approximately 10^{4} connections (Du & Swamy, 2013). A depiction of neurons is shown in Figure 4.1 and can be considered as the hardware used by the brain to perform sophisticated processing tasks in parallel.

Only the three major functional units relevant to artificial neural networks are considered in the simplified explanation of the functioning of a biological neuron. These functional units are known as the dendrites, cell body, and axon (Basheer & Hajmeer, 2000). The cell body, also known as the soma, is the central unit of the neuron and consists of a nucleus (which contains information on hereditary traits), plasma (containing the molecular equipment needed to produce material for the neuron), and several tree-like protrusions, known as dendrites. The dendrites receive electrical impulses as input, which is then passed to the cell body to be processed. Once it has been processed, a new output signal consisting of a set of impulses is sent down the neuron's axon to the pre-synaptic membrane. Synapses are the microscopic gaps between two consecutive neurons (Eluyode & Akomolafe, 2013). Depending on the strength of the signals arriving at the membrane, a certain amount of a chemical, known as a neurotransmitter, is released. The neurotransmitter diffuses toward the dendrites of the subsequent neuron. Once the signal reaches a certain threshold, it enters through the dendrite's membrane and the process of passing along the signal through the cell body and axon is repeated. The intensity of the signal being passed through this neuron is dependent on the intensity of the signals of each of the feeding neurons, the synaptic strength, and the threshold of the receiving neuron. The signal strength may either inhibit or stimulate the firing of the neuron.

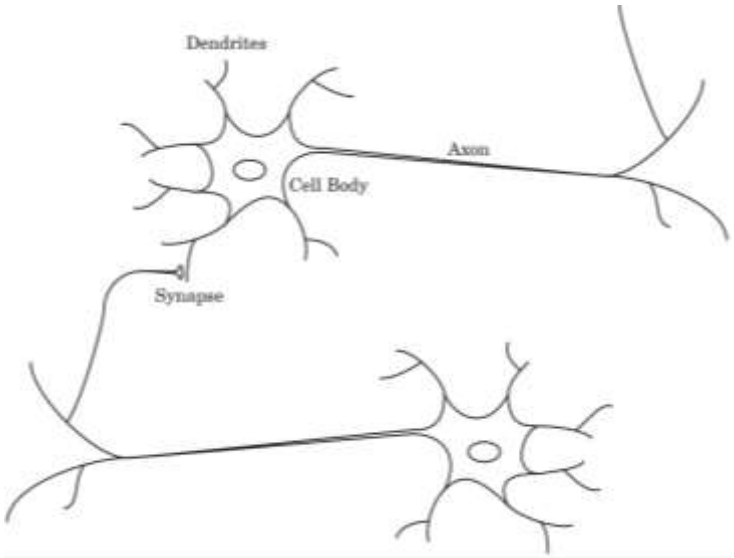


Figure 4.1: Simplified biological neural network (Hagan *et al.*, 2014)

Engineers and computer scientists use this simplified neuron model to understand how information is processed by neurons and apply it to design software that can adapt and learn (Coolen, 1998). These artificial neurons, just like biological neurons, are not pre-programmed, but are instead trained. Moreover, engineers and computer scientists study parallel information processing, as performed by biological neural networks, and apply it by building hardware that can also operate in parallel.

4.3. Artificial neural networks fundamentals

In this section, the aim is to present a better comprehension of the analogy between biological and artificial neural networks. Therefore, an explanation of the fundamental concepts of artificial neural networks is provided in order to understand the basic building blocks of deep neural networks.

4.3.1. Artificial neurons

Artificial neural networks consist of certain components similar to those of a biological neuron, i.e. connections between nodes, connection weights, and a threshold (Basheer & Hajmeer, 2000). An artificial neuron that has similar features to the biological neuron, as shown in Figure 4.1, is depicted in Figure 4.2.

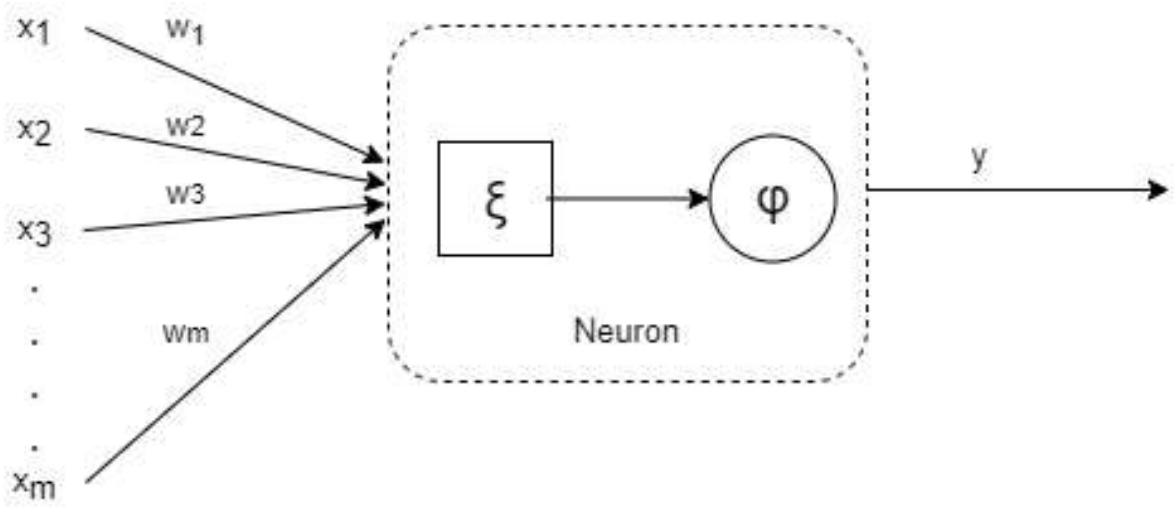


Figure 4.2: Artificial neuron (Azam, 2000; Basheer & Hajmeer, 2000)

The artificial neuron receives stimuli from the environment as input in the form of a vector of values $x_1, x_2, \dots, x_m$ (Hagan *et al.*, 2014; Azam, 2000; Basheer & Hajmeer, 2000). These values are combined in order to form a net input (ξ), by multiplying it with another vector of weight values $w_1, w_2, \dots, w_m$, and adding a bias (b). This can be explained mathematically as follows:

$$\xi = \sum_{i=1}^{m} w_i x_i + b, \tag{4.1}$$

where each value w_i represents the synapse or connection strength between two consecutive nodes and enhances the net signal. These values can indicate either an inhibitory, excitatory or no connection. When the weight value is negative, it inhibits the neuron activity. Should the value of the weight be positive, it stimulates, or excites, the neuron. If the weight value is equal to zero, it does not affect the neuron. The bias is treated as an additional input node with a value of 1 (i.e. $x_0 = 1$), and a connection weight of value b.

The value of ξ is then passed over a threshold gate (φ), also known as an activation or transfer function, before being transmitted as an output signal (y) to another neuron. This can be formally stated as

$$y = \varphi(\xi) = \varphi(\sum_{i=1}^{m} w_i x_i + b). \quad (4.2)$$

4.3.2. Activation functions

The purpose of an activation function is to limit the allowable amplitude range of the output produced by an artificial neuron to some finite value, as well as to simulate the nonlinearity of complex systems (Khalafi & Mirvakili, 2011; Haykin, 2009). Different activation functions that are widely used, as well as their definitions and functional plots, are shown in Table 4.1 (Ding *et al.*, 2018; Hagan *et al.*, 2014). Though different activation functions are contained in Table 4.1, only the *rectified linear unit* (*ReLU*) and *softmax* activation functions are discussed, as these are the functions that are used in the construction of the deep MLP for the study.

Table 4.1: Activation functions (Ding *et al.*, 2018; Hagan *et al.*, 2014)

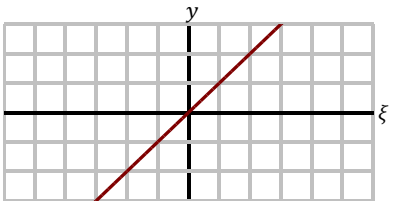
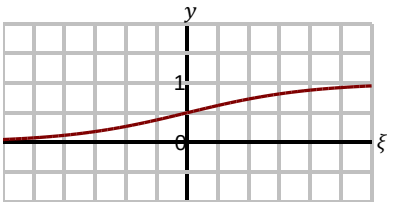
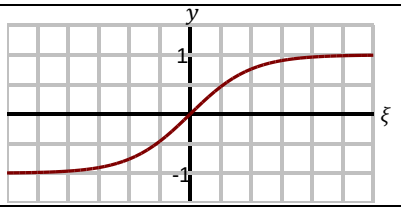
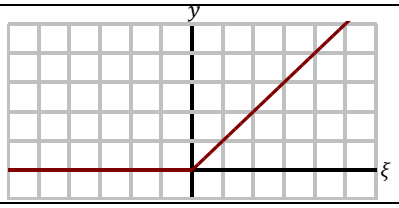
Activation function	**Definition**	**Functional plot**
Linear	$y = \xi$	
Logistic sigmoid	$y = \frac{1}{1 + e^{\xi}}$	
Hyperbolic tangent sigmoid	$y = \frac{e^{\xi} - e^{-\xi}}{e^{\xi} + e^{-\xi}}$	

Table 4.1: Activation functions (continued)

Rectified linear unit (*ReLU*)	$y = \max(0, \xi) = \begin{cases} 0, if\ \xi \leq 0 \\ \xi, if\ \xi > 0 \end{cases}$	
Softmax	$y_i = \frac{e^{\xi_i}}{\sum_{j=1}^{n} e^{\xi_j}}\ for\ i = 1, \ldots, n$	No functional plot

4.3.2.1. Rectified linear unit

Neuroscientists found that neurons in the outer layer of the cerebrum are not often at full saturation capacity (Ding *et al.*, 2018). Additionally, it was found that the number of neurons that can be active simultaneously is about one to four per cent of the neurons inside the brain. However, up to 50% of the neurons in an artificial neural network can be activated at the same time, when applying the *logistic sigmoid* or *hyperbolic tangent sigmoid* function. This is not only inconsistent with the above-mentioned findings, but can lead to additional difficulties while training the artificial neural network.

The *ReLU* activation function can help in obtaining comparable sparsity levels using artificial neurons, compared to neurons found within the brain (Ding *et al.*, 2018). This will then lead to an improvement in the efficiency of the artificial neural network. It is seen as a breakthrough in training supervised deep neural networks and is defined in Table 4.1. Its first derivative function is given by

$$\frac{\partial y}{\partial x} = \begin{cases} 1, \text{if } \xi \geq 0 \\ 0, \text{if } \xi < 0 \end{cases}. \tag{4.3}$$

Due to being similar to linear units, a *ReLU* function is easily optimised (Goodfellow *et al.*, 2016). The major difference between it and a linear unit function is that while linear units output values across its entire domain, this is only true for half of a *ReLU*'s domain, as the other half outputs zeroes. This leads to a consistent and large gradient when the first derivative is calculated for a *ReLU*. More specifically, the first derivative has a value of 1 everywhere that it is active, as shown in (4.3), and the second derivative is 0 across most of its domain. This is beneficial, as it makes the *ReLU* function far more useful than other activation functions that would typically introduce second-order effects in the learning process. It offers several other benefits, including the following (Ding *et al.*, 2018):

- Since it does not make use of an exponential function, and therefore does not need to be computed in the nodes, it offers cheaper activation function computations than sigmoid and hyperbolic tangent activation functions.
- As opposed to using saturating activation functions, such as the *logistic sigmoid* or *hyperbolic tangent sigmoid* functions, neural networks using the *ReLU* activation function converge much faster in terms of training time with gradient descent training.
- Due to having derivatives that are the constant 1, it avoids getting trapped into local optimisation, thereby resolving the vanishing gradient effect (Hanin, 2018) that occurs when sigmoid and hyperbolic tangent activation functions are used.
- Supervised learning neural networks using it can achieve their best performance without any pre-training by using unsupervised learning.

Although the *ReLU* has several benefits as listed above, and is recommended for use with MLP deep neural networks, it still has some disadvantages (Goodfellow *et al.*, 2016). Due to the derivative being equal to 0 when $\xi < 0$, it is left-hard-saturated (Ding *et al.*, 2018). This may lead to the relative weights not being updated and consequently cause some of the nodes within the artificial neural network never being activated. As the average output of the nodes is identically positive, using it can also cause a bias shift for the nodes in the next layer of the neural network.

4.3.2.2. Softmax

The *softmax* activation function is widely used as the output activation function of a neural network that makes classifications, as a result of its simplicity and probabilistic interpretation (Goodfellow *et al.*, 2016; Liu *et al.*, 2016). This is due to its ability to represent the probability distribution over n classes or possible values. However, this does not mean that it cannot be used within the artificial neural network itself. When the artificial neural network must be able to select an option from n different options, the *softmax* activation function may be used. The *softmax* activation function is defined as follows:

$$y_i = \frac{e^{\xi_i}}{\sum_{j=1}^{n} e^{\xi_j}}, \text{ for } i = 1, \dots, n, \tag{4.4}$$

where y_i denotes the output of node i.

The *softmax* activation function ensures that all the outputs generated by a layer within an artificial neural network are values between zero and one, as well as that these values add up to one (Graves & Schmidhuber, 2005). It assigns a probability, based on all the inputs provided to the network up to the specific point where the *softmax* is calculated.

4.3.3. Drop-out rate

Combining different machine learning models and averaging their predictions often improves the performance of these models (Srivastava, 2013). However, this technique becomes unfeasible when training large neural networks, as they are often difficult to train and quite slow. Drop-out, the action of dropping out or removing units in a neural network, may be used to address both the aforementioned concerns. This action is only performed during the forward pass of the training phase of the model; thus, all units are used during testing (Pham *et al.*, 2014). According to Dahl *et al.* (2013), making use of drop-out during training can be considered as training and combining multiple models that differ in architecture. Each unit has a probability of p to be dropped out, where p typically has a value of 0.5. However, as a higher drop-out rate or probability can potentially lead to higher error rates and slower convergence, a smaller value may be used.

4.3.4. Properties of artificial neural networks

Artificial neural networks possess some properties that make them applicable for solving specific problems (Du & Swamy, 2013; Azam, 2000). These properties are briefly discussed below:

1. *Generalisation*: An artificial neural network with continuous activation functions can provide similar outputs as long as the data points received as inputs are also similar. Therefore, artificial neural networks can easily generalise from the cases it has already learned to new cases presented to it.
2. *Robustness and fault tolerance*: Neural networks can process data that is imprecise, fuzzy, or noisy.
3. *Graceful degradation*: Missing or distorted data will lead to only a slight diminishing of the artificial neural network's performance. This is because of the scenarios needing classification being similar to those that were used to train the model. The performance degradation is in proportion to the inaccuracy within the data presented to it.
4. *Adaptation and learning*: Artificial neural networks' knowledge is indirectly encoded into it. The artificial neural network also tries to maintain this knowledge when the conditions under which it operates change.
5. *Parallelism*: Connecting weights linked to most or all of the artificial neurons can easily be changed simultaneously. This gives the added advantage of being able to distribute tasks that are computationally expensive to a large number of artificial neurons.
6. *General-purpose nonlinear nature*: Neural networks perform like a black box.

4.3.5. Categories of neural networks

Most classification and regression problems can easily be approached through the use of neural networks. The task of classification aims to determine the class to which an entity belongs, based on its attribute values (Kirchner *et al.*, 2015). On the other hand, solving regression problems involve using the predictor values to predict numerical values, whilst learning a model that minimises the loss function (Jagielski *et al.*, 2018). Neural network architectures can be classified into several categories based on their intended use, i.e. neural networks for supervised learning, neural networks for unsupervised learning, semi-supervised learning, self-supervised learning, and hybrid neural networks (Goodfellow *et al.*, 2016; Deng & Yu, 2014). Though this study makes use of a supervised learning architecture, each of these categories is briefly discussed in this section.

4.3.5.1. Supervised learning

Supervised learning is the most common category of learning algorithms (LeCun *et al.*, 2015). There are a few steps involved in this type of learning. Data is collected and sorted into categories which are indicated by adding the appropriate label for each input vector. During the training phase, an input vector is shown to the model, which in turn outputs a vector of scores containing a value for each of the categories. Ideally, the category that best fits the shown vector will have the highest score. An objective function is applied to determine the error of the obtained output scores compared to the desired output scores. This is followed by the model adjusting its parameters, or weights, with the goal of reducing this error.

4.3.5.2. Unsupervised learning

In contrast to supervised learning algorithms which make use of labelled data sets for training, unsupervised learning algorithms do not make use of any such information (Deng & Yu, 2014). The techniques used within these types of learning algorithm are typically aimed at capturing high-order correlation within the observed data. More informally, unsupervised learning refers to those techniques within machine learning that do not require a human to label the data set (Goodfellow *et al.*, 2016). Unsupervised learning is generally focused on estimating density, de-noising data, and grouping data into groups that are related to each other (known as clustering).

4.3.5.3. Semi-supervised learning

Semi-supervised learning is a combination of both supervised and unsupervised learning (Chapelle *et al.*, 2006). The data set used with this type of learning can be divided into two parts, i.e. the unlabelled and labelled data points. The goal of semi-supervised learning is to learn a representation of the data points that are from the same class and have similar properties (Goodfellow *et al.*, 2016). The algorithm needs to compensate for the lack of labelled data (Chebli *et al.*, 2018). A more accurate model can be constructed using the unlabelled data

for modifying the suggested result obtained from the provided labelled data. This also results in reducing the need for manual labour for annotating the data points.

4.3.5.4. Self-supervised learning

Self-supervised learning is a subset of unsupervised learning methods (Jing & Tian, 2019). This method does not need any annotations on the data made by humans, as it generates its own pseudo labels for the data. The pseudo label is generated making use of either attributes from the original data source, or other traditional hand-designed methods. Hence, the computer can learn to teach itself to do supervised tasks (Mundhenk *et al.*, 2018). The self-supervised neural network may perform just as well as supervised networks if the representations that were learned are of good quality.

4.3.5.5. Hybrid learning

Hybrid learning neural networks are often set up in such a way that the results of unsupervised learning neural networks are used to assist with the goal of discrimination of the data (Deng & Yu, 2014). This can be done by optimising supervised learning neural networks. Furthermore, criteria for supervised learning can be used to estimate parameters in unsupervised learning neural networks.

4.3.6. A simple neural network example

In this section, an example of a shallow artificial neural network consisting of an input layer, one hidden layer, and an output layer to illustrate the basic functioning of such a network is included. The presented network makes use of the *ReLU* and s*oftmax* activation functions. For simplicity, the bias and weight values for the presented problem only consist of integer values. Consider the following description of Fisher's (1936) Iris data set:

> The Iris data set consists of 150 records describing three species of the Iris flower, i.e. *Iris setosa, Iris virginica* and *Iris versicolor*. Each class has 50 records that comprise the following four attributes measured in centimetres: petal length and width, as well as sepal length and width.

Since each record of the data set has four attributes, the sample artificial neural network needs to have four input nodes to represent these attributes. Three output nodes are used to indicate the three classes, respectively *Iris setosa, Iris virginica* and *Iris versicolor*, along with five artificial neurons in the hidden layer. This architecture is shown in Figure 4.3. The *ReLU* activation function is used within the hidden layer, whilst the *softmax* activation function is used in the output layer.

To illustrate how this artificial neural network works, assume the following weight matrix ($\boldsymbol{W}_1$) indicates the weights assigned to each input value from input node i to hidden node j:

$$\boldsymbol{W}_1 = \begin{bmatrix} 1 & 2 & 1 & 2 \\ 2 & 1 & 2 & -1 \\ 1 & 2 & -2 & 1 \\ -3 & 2 & 1 & 3 \\ 1 & 1 & 1 & 1 \end{bmatrix}, \tag{4.5}$$

and $\boldsymbol{W}_2$ indicates the weight values between each hidden node j and output node k:

$$\boldsymbol{W}_2 = \begin{bmatrix} 2 & 1 & 5 & 0 & 2 \\ 2 & 3 & 1 & 2 & 1 \\ 2 & 1 & 2 & 6 & 3 \end{bmatrix}. \tag{4.6}$$

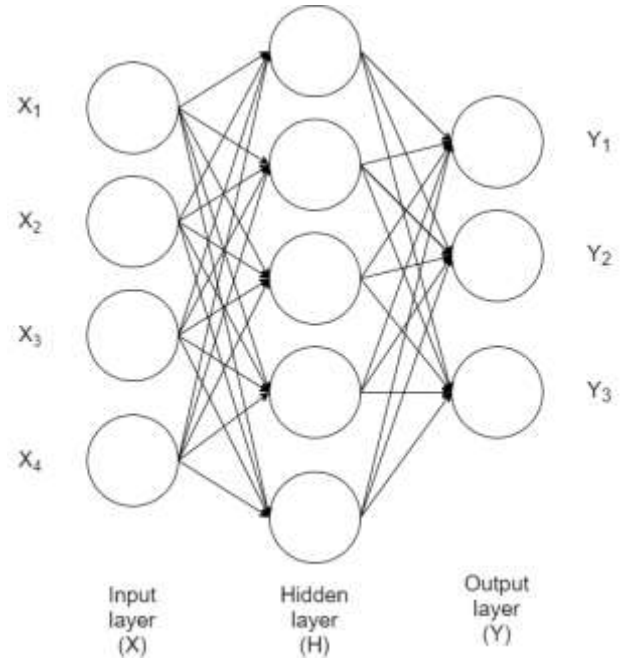


Figure 4.3: Architecture of simple artificial neural network for classifying the Iris data set

The biases of the hidden layer are represented by

$$\boldsymbol{b}_h = \begin{bmatrix} -1 \\ 1 \\ 3 \\ 2 \\ 0 \end{bmatrix}, \tag{4.7}$$

and the biases for the output layer by

$$\boldsymbol{b}_y = \begin{bmatrix} 1 \\ 2 \\ 2 \end{bmatrix}. \tag{4.8}$$

Given the following vector, representing an *Iris setosa*, as input to the artificial neural network

$$x = \begin{bmatrix} 5.1 \\ 3.5 \\ 1.4 \\ 0.2 \end{bmatrix}, \tag{4.9}$$

the net input for the first hidden neuron (h_1) can be calculated as

$$\begin{aligned} \xi &= \sum_{i=1}^{m} w_i x_i + b \\ &= \begin{bmatrix} 1 & 2 & 1 & 2 \end{bmatrix} \begin{bmatrix} 5.1 \\ 3.5 \\ 1.4 \\ 0.2 \end{bmatrix} + (-1) \\ &= (1)(5.1) + (2)(3.5) + (1)(1.4) + (2)(0.2) + \\ &(-1) \\ &= 12.9. \end{aligned} \tag{4.10}$$

Following the same formula, the net input values for the other three hidden neurons are calculated and result in, respectively 0, 5, and 8. Using the *ReLU* activation function the output of h_1 is calculated as follows:

$$y_{h_1} = \max(0, 12.9) = 12.9. \tag{4.11}$$

Repeating these calculations for each of the hidden neurons results in the following vector:

$$y_h = \begin{bmatrix} 12.9 \\ 17.3 \\ 12.5 \\ 0 \\ 10.2 \end{bmatrix}. \tag{4.12}$$

This vector is then used as the input to the output layer. Thus, the net inputs as calculated using (4.1) are 127, 102.4 and 100.7, respectively for the three output nodes. The output values of the output nodes are calculated using the *softmax* activation function given in (4.4). Thus, the output value of the first output node (y_1) is calculated as follows:

$$\begin{aligned} y_1 &= \frac{e^{\xi_i}}{\sum_{j=1}^{2} e^{\xi_j}} \\ &= \frac{e^{127}}{e^{127} + e^{102.4} + e^{100.7}} \\ &= 0.999999999975497, \end{aligned} \tag{4.13}$$

and the output values for the second and third output nodes (y_2 and y_3) are respectively

$$y_2 = \frac{e^{\xi_i}}{\sum_{j=1}^{2} e^{\xi_j}}$$
$$= \frac{e^{102.4}}{e^{127}+e^{102.4}+e^{100.7}}$$
$$= 0.00000000002071838, \quad (4.14)$$

and

$$y_3 = \frac{e^{\xi_i}}{\sum_{j=1}^{2} e^{\xi_j}}$$
$$= \frac{e^{100.7}}{e^{127}+e^{102.4}+e^{100.7}}$$
$$= 0.00000000000378491, \quad (4.15)$$

Therefore, the artificial neural network predicts that the outcome of the given input in (4.9) is most likely an *Iris setosa*.

When using a vector of the form

$$x = \begin{bmatrix} 7.0 \\ 3.2 \\ 4.7 \\ 1.4 \end{bmatrix}, \quad (4.16)$$

as input, indicating an *Iris versicolor*, the resulting vector for the hidden layer's output is

$$y_h = \begin{bmatrix} 19.9 \\ 26.2 \\ 8.4 \\ 0 \\ 16.3 \end{bmatrix}. \quad (4.17)$$

Therefore, the net input values for the three respective output nodes are 141.6, 145.1 and 133.7. The output of the neural network is then calculated as

$$y_1 = \frac{e^{\xi_i}}{\sum_{j=1}^{2} e^{\xi_j}}$$
$$= \frac{e^{141.6}}{e^{141.6}+e^{145.1}+e^{133.7}}$$
$$= 0.0293119122, \quad (4.18)$$
$$y_2 = \frac{e^{\xi_i}}{\sum_{j=1}^{2} e^{\xi_j}}$$
$$= \frac{e^{145.1}}{e^{141.6}+e^{145.1}+e^{133.7}}$$
$$= 0.9706772206, \quad (4.19)$$

$$y_3 = \frac{e^{\xi_i}}{\sum_{j=1}^{2} e^{\xi_j}}$$
$$= \frac{e^{133.7}}{e^{141.6}+e^{145.1}+e^{133.7}}$$
$$= 0.0000108672, \tag{4.20}$$

which shows that the neural network correctly classifies it as an *Iris versicolor*.

Lastly, given the following vector that represents an *Iris virginica* as input*:*

$$x = \begin{bmatrix} 6.3 \\ 3.3 \\ 6.0 \\ 2.5 \end{bmatrix}, \tag{4.21}$$

the output vector of the hidden layer is thus,

$$y_h = \begin{bmatrix} 22.9 \\ 26.4 \\ 6.4 \\ 3.2 \\ 18.1 \end{bmatrix}. \tag{4.22}$$

This provides the following net input values to the output layer: 141.4, 157.9 and 160.5. After calculating the output of the neural network, it can be seen that *Iris virginica* is also correctly classified, as follows:

$$y_1 = \frac{e^{\xi_i}}{\sum_{j=1}^{2} e^{\xi_j}}$$
$$= \frac{e^{141.4}}{e^{141.4}+e^{157.9}+e^{160.5}}$$
$$= 0.00000000472, \tag{4.23}$$
$$y_2 = \frac{e^{\xi_i}}{\sum_{j=1}^{2} e^{\xi_j}}$$
$$= \frac{e^{157.9}}{e^{141.4}+e^{157.9}+e^{160.5}}$$
$$= 0.06913842002, \tag{4.24}$$

and

$$y_3 = \frac{e^{\xi_i}}{\sum_{j=1}^{2} e^{\xi_j}}$$
$$= \frac{e^{160.5}}{e^{141.4}+e^{157.9}+e^{160.5}}$$
$$= 0.93086157526, \tag{4.25}$$

In Section 4.3, an overview of the fundamental concepts relating to artificial neural networks was provided. In the next section, a brief discussion of how these concepts can be applied to create deep neural networks is included.

4.4. Deep learning

Thus far, the biological origins, which inspired the work in machine learning and artificial neural networks, have been discussed in this chapter, followed by an explanation of the building blocks of neural networks, i.e. artificial neurons, which are based on the working of biological neurons. In this section, deep learning, before considering a typical deep learning architecture, known as a multilayer perceptron (MLP) neural network, is explained.

4.4.1. Background and definitions

Machine learning is a subfield of artificial intelligence which focuses on the construction of computer programs that can automatically adapt based on its experience (Jordan & Mitchell, 2015). It has a broad field of applications, including, but not limited to, computer vision, speech recognition, natural language processing, and robotics. Until recently, research within the field of machine learning mainly used shallow artificial neural networks, consisting of at most two hidden layers, and an input layer (Deng & Yu, 2014). These shallow models proved to be effective in solving simple and well-constrained problems. However, difficulties arose when they were applied to problems that had higher levels of complexity involved, such as processing human speech, language, and natural images and scenes. The processing of raw natural data by applying techniques that made use of shallow artificial neural networks was somewhat limited (LeCun *et al.*, 2015). To design a machine learning system for extracting and transforming the raw input data into an internal representation that could easily be used by the classifier to recognise and correctly classify patterns in the input, required considerable domain expertise and careful engineering.

The field of deep learning emerged from the field of machine learning and artificial neural network research in 2006 (Deng & Yu, 2014). It drew inspiration from the deep architectures used for the extraction of complex structures and the building of internal representations based on rich sensory inputs, as used by human information processing mechanisms. Deep learning models can learn complex functions because of the ability to transform a representation at one (lower) level into a higher abstracted representation (LeCun *et al.*, 2015). This transformation starts from the raw input data and ensures that aspects that are relevant to the classification problem are highlighted, and other irrelevant aspects are suppressed. Gradient-based optimisation algorithms, such as the backpropagation algorithm explained in Section 4.4.2.2,

are used in deep learning models in order to adjust its parameters of the network based on the error rate of the output (Jordan & Mitchell, 2015).

Five closely related definitions describing deep learning with deep architectures for signal and information processing are provided by Deng and Yu (2014). Before presenting these definitions, it should be noted that all of them have two key aspects in common. First of all, these models consist of multiple layers of non-linear information processing units or nodes. Secondly, supervised or unsupervised learning methods can be used at each of the consecutively higher levels of abstracted layers for feature representation. The definitions are as follows:

1. A class of machine learning techniques that exploits many layers of non-linear information processing units for supervised or unsupervised feature extraction and transformation, and for pattern analysis and classification.
2. A sub-field within machine learning that is based on algorithms for learning multiple levels of representation in order to model complex relationships among data. Higher-level features and concepts are thus defined in terms of lower-level ones, and such a hierarchy of features is called a deep architecture.
3. A sub-field of machine learning that is based on learning several levels of representations, corresponding to a hierarchy of features or factors or concepts, where higher-level concepts are defined from lower-level ones and the same lower-level concepts can help to define many higher-level concepts. Deep learning is part of a broader family of machine learning methods based on learning representations. An observation (e.g., an image) can be represented in many ways (e.g., a vector of pixels), but some representations make it easier to learn tasks of interest (e.g., is this the image of a human face?) from examples, and research in this area attempts to define what makes better representations and how to learn them.
4. Deep learning is a set of algorithms in machine learning that attempts to learn in multiple levels, corresponding to different levels of abstraction. It typically uses artificial neural networks. The levels in these learned statistical models correspond to distinct levels of concepts, where higher-level concepts are defined from lower-level ones, and the same lower-level concepts can help to define many higher-level concepts.
5. Deep learning is a new area of machine learning research, which has been introduced with the objective of moving machine learning closer to one of its original goals: artificial intelligence. Deep learning is about learning multiple levels

of representation and abstraction that help to make sense of data, such as images, sound, and text.

From the above, deep learning can be summarised as a subfield of machine learning that makes use of neural networks consisting of many levels of abstraction for learning and modelling of the complex relationships between data, thereby solving classification or regression problems.

4.4.2. Deep neural networks

A discussion on how several artificial neurons can be connected to form deep neural networks, more specifically deep MLP neural networks, is provided within this section.

4.4.2.1. Multilayer perceptron neural network architecture

The MLP neural network, or the deep feedforward network, often used to model a deep neural network, is based on Rosenblatt's original Perceptron model of 1958 (Goodfellow *et al.*, 2016; Ramchoun *et al.*, 2016; Rosenblatt, 1958). It consists of an input and output layer, as well as one or more hidden layers in between these two layers. The goal of such a network is to map an input x to category y, thus $y = f(x, \theta)$, where the value of the parameters represented by θ is learned as a result of the function approximation. The term feedforward is used to describe this type of neural network, since the output from nodes moves in from nodes in the lower layers to nodes in the upper layers and the nodes within the same layer are not interconnected. Additionally, it does not contain any connections between the nodes in which the output of the network is fed back into the model itself. The information flows in a single direction and in a sequential manner through the different hidden layers, through the consecutive nodes, towards the output y. A generic architecture of an MLP is depicted in Figure 4.4, where x_m indicates the m^{th} input, y_n indicates the n^{th} output, h_i indicates the i^{th} hidden layer and W_j indicates the j^{th} weight matrix of the deep neural network.

The number of nodes in the input layer is equivalent to the number of measurements within the input pattern (Ramchoun *et al.*, 2016). On the other hand, the number of categories or classes used to classify the pattern is represented by the same number of nodes within the output layer of the MLP. Each of the hidden layers between the input layer and the output layer may differ in terms of the number of nodes contained in each layer and the transfer function used (Hagan *et al.*, 2014). Determining the number of hidden layers, as well as the number of nodes within the different hidden layers, presents a problem known as the architecture selection problem. The number of hidden nodes may be determined by making use of optimisation of the network architecture, using adequate hyperparameters for the task of classification or regression during the training process.

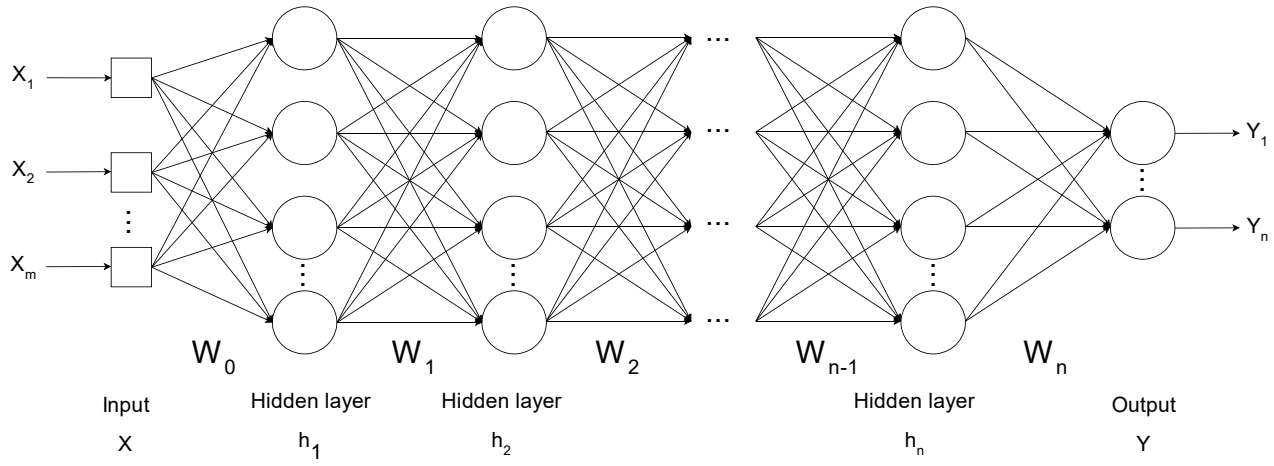


Figure 4.4: Generic architecture of an MLP (Ramchoun *et al.*, 2016)

4.4.2.2. Backpropagation

The aim of the training phase is to allow the MLP to learn (Ramchoun *et al.*, 2016); that is, adjusting connection weights to minimise the error between the output produced by the MLP and the desired output. To do this, the backpropagation algorithm is popularly used. Though this algorithm was first contained in Paul Werbos's thesis in 1974 (Werbos, 1974), it was not widely used with neural networks until the mid-1980s (Hagan *et al.*, 2014).

The algorithm receives a set of input values, indicating the desired behaviour, and outcome, of the MLP. This set can be depicted as follows:

$$\{\boldsymbol{p}_1, \boldsymbol{t}_1\}, \{\boldsymbol{p}_2, \boldsymbol{t}_2\}, \dots, \{\boldsymbol{p}_Q, \boldsymbol{t}_Q\}, \tag{4.26}$$

where $\boldsymbol{p}_q$ refers to the input value, and $\boldsymbol{t}_q$ refers to the matching target output. As each of these inputs is applied to the artificial neural network, its parameters are adjusted accordingly, based on the *mean square error* calculated, using the expected output value compared to the actual output. Suppose the output of one layer, which serves as the input of the subsequent layer, can be described as:

$$\boldsymbol{a}^{m+1} = \boldsymbol{f}^{m+1}(\boldsymbol{W}^{m+1}\,\boldsymbol{a}^m + \boldsymbol{b}^{m+1}) \text{ for } m = 0, 1, \dots, M-1, \tag{4.27}$$

where M indicates the number of layers within the MLP, $\boldsymbol{W}^{m+1}$ and $\boldsymbol{b}^{m+1}$ refer to the weight vector and bias of layer $m+1$, respectively. The first layer of the MLP will receive inputs from some external stimuli, and is the starting point for (4.27):

$$\boldsymbol{a}^0 = \boldsymbol{p}. \tag{4.28}$$

The output of the MLP is equivalent to the last layer's output and is represented by

$$\boldsymbol{a} = \boldsymbol{a}^M. \tag{4.29}$$

From this information, the *mean square error* can be calculated, using the following equation:

$$F(\boldsymbol{x}) = E[e^2] = E[(t-a)^2], \tag{4.30}$$

where x refers to the vector of weights and biases of the MLP, and E refers to the expected value. This can be further generalised when the MLP has multiple outputs to

$$F(\boldsymbol{x}) = E[\boldsymbol{e}^T\boldsymbol{e}] = E[(\boldsymbol{t}-\boldsymbol{a})^T(\boldsymbol{t}-\boldsymbol{a})]. \tag{4.31}$$

The *mean square error* can be approximated by replacing the expected value for the squared error with the squared error at iteration k, resulting in the following equation:

$$\hat{F}(\boldsymbol{x}) = \big(\boldsymbol{t}(k)-\boldsymbol{a}(k)\big)^T\big(\boldsymbol{t}(k)-\boldsymbol{a}(k)\big) = \boldsymbol{e}^T(k)\boldsymbol{e}(k). \tag{4.32}$$

To calculate the stochastic gradient descent, in order to adjust the weights ($w_{i,j}^m$) and biases (b_i^m), using the *mean square error,* the following algorithm is used:

$$w_{i,j}^m(k+1) = w_{i,j}^m(k) - \alpha\frac{\partial\hat{F}}{\partial w_{i,j}^m}, \tag{4.33}$$

$$b_i^m(k+1) = b_i^m(k) - \alpha\frac{\partial\hat{F}}{\partial b_i^m}, \tag{4.34}$$

where α is the learning rate. The derivatives used in (4.33) and (4.34) can be calculated, using the chain rule as follows:

$$\frac{\partial\hat{F}}{\partial w_{i,j}^m} = \frac{\partial\hat{F}}{\partial n_i^m}\times\frac{\partial n_i^m}{\partial w_{i,j}^m}, \tag{4.35}$$

$$\frac{\partial\hat{F}}{\partial b_i^m} = \frac{\partial\hat{F}}{\partial n_i^m}\times\frac{\partial n_i^m}{\partial b_i^m}, \tag{4.36}$$

where n_i^m, or the net input for layer m and its first derivatives are, respectively, defined by

$$n_i^m = \sum_{j=1}^{S^{m-1}} w_{i,j}^m\, a_j^m + b_i^m, \tag{4.37}$$

$$\frac{\partial n_i^m}{\partial w_{i,j}^m} = a_j^{m-1}, \tag{4.38}$$

$$\frac{\partial n_i^m}{\partial b_i^m} = 1. \tag{4.39}$$

Using this information, the sensitivity of $\hat{F}$ to the changes in the ith element of the net input at layer, m can be defined as

$$s_i^m \equiv \frac{\partial \hat{F}}{\partial n_i^m}, \tag{4.40}$$

which leads to the simplification of (4.35) and (4.36) as follows:

$$\frac{\partial \hat{F}}{\partial w_{i,j}^m} = s_i^m a_j^{m-1}, \tag{4.41}$$

$$\frac{\partial \hat{F}}{\partial b_i^m} = s_i^m. \tag{4.42}$$

The stochastic gradient descent algorithm can be approximated as

$$w_{i,j}^m(k+1) = w_{i,j}^m(k) - \alpha s_i^m a_j^{m-1}, \tag{4.43}$$

$$b_i^m(k+1) = b_i^m(k) - \alpha s_i^m. \tag{4.44}$$

These equations are better expressed in matrix form as follows:

$$\boldsymbol{W}^m(k+1) = \boldsymbol{W}^m(k) - \alpha \boldsymbol{s}^m (\boldsymbol{a}^{m-1})^T, \tag{4.45}$$

$$\boldsymbol{b}^m(k+1) = \boldsymbol{b}^m(k) - \alpha \boldsymbol{s}^m, \tag{4.46}$$

where

$$\boldsymbol{s}^m \equiv \frac{\partial \hat{F}}{\partial \boldsymbol{n}^m} = \begin{bmatrix} \frac{\partial \hat{F}}{\partial n_1^m} \\ \frac{\partial \hat{F}}{\partial n_2^m} \\ \vdots \\ \frac{\partial \hat{F}}{\partial n_{S^m}^m} \end{bmatrix}. \tag{4.47}$$

However, the sensitivities ($\boldsymbol{s}^m$) should be computed and backpropagated before the gradient can be calculated. The sensitivity at layer m is calculated using the sensitivity at layer $m+1$. The following Jacobian matrix is used to derive the sensitivities' recurrence relationships:

$$\frac{\partial \boldsymbol{n}^{m+1}}{\partial \boldsymbol{n}^m} \equiv \begin{bmatrix} \frac{\partial n_1^{m+1}}{\partial n_1^m} & \frac{\partial n_1^{m+1}}{\partial n_2^m} & \cdots & \frac{\partial n_1^{m+1}}{\partial n_{S^m}^m} \\ \frac{\partial n_2^{m+1}}{\partial n_1^m} & \frac{\partial n_2^{m+1}}{\partial n_2^m} & \cdots & \frac{\partial n_2^{m+1}}{\partial n_{S^m}^m} \\ \vdots & \vdots & & \vdots \\ \frac{\partial n_{S^{m+1}}^{m+1}}{\partial n_1^m} & \frac{\partial n_{S^{m+1}}^{m+1}}{\partial n_2^m} & \cdots & \frac{\partial n_{S^{m+1}}^{m+1}}{\partial n_{S^m}^m} \end{bmatrix}, \tag{4.48}$$

where the i, j element is calculated by

$$\frac{\partial n_i^{m+1}}{\partial n_j^m} = \frac{\partial\left(\sum_{l=1}^{S^m} w_{i,l}^{m+1} a_l^m + b_i^{m+1}\right)}{\partial n_j^m}$$
$$= w_{i,j}^{m+1} \frac{\partial a_j^m}{\partial n_j^m}$$
$$= w_{i,j}^{m+1} \frac{\partial f^m\left(n_j^m\right)}{\partial n_j^m}$$
$$= w_{i,j}^{m+1} \dot{f}^m\left(n_j^m\right), \tag{4.49}$$

and

$$\dot{f}^m\left(n_j^m\right) = \frac{\partial \dot{f}^m\left(n_j^m\right)}{\partial n_j^m}. \tag{4.50}$$

Thus, the Jacobian matrix can be simplified to

$$\frac{\partial \boldsymbol{n}^{m+1}}{\partial \boldsymbol{n}^m} = \boldsymbol{W}^{m+1} \dot{\boldsymbol{F}}^m(\boldsymbol{n}^m), \tag{4.51}$$

where

$$\dot{\boldsymbol{F}}^m(\boldsymbol{n}^m) = \begin{bmatrix} \dot{f}^m(n_1^m) & 0 & \cdots & 0 \\ 0 & \dot{f}^m(n_2^m) & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & \dot{f}^m\left(n_{S^m}^m\right) \end{bmatrix}. \tag{4.52}$$

The recurrence relation of the sensitivity can be written in matrix form using the chain rule as

$$\boldsymbol{s}^m = \frac{\partial \widehat{\boldsymbol{F}}}{\partial \boldsymbol{n}^m}$$
$$= \left(\frac{\partial \boldsymbol{n}^{m+1}}{\partial \boldsymbol{n}^m}\right)^T \frac{\partial \widehat{\boldsymbol{F}}}{\partial \boldsymbol{n}^{m+1}}$$
$$= \dot{\boldsymbol{F}}^m(\boldsymbol{n}^m)(\boldsymbol{W}^{m+1})^T \frac{\partial \widehat{\boldsymbol{F}}}{\partial \boldsymbol{n}^{m+1}}$$
$$= \dot{\boldsymbol{F}}^m(\boldsymbol{n}^m)(\boldsymbol{W}^{m+1})^T \boldsymbol{s}^{m+1}. \tag{4.53}$$

These sensitivities can then be sent or propagated, backwards through each layer of the artificial neural network, starting at the last layer to the first layer. The term backpropagation is used to describe this process and is defined as

$$\boldsymbol{s}^M \to \boldsymbol{s}^{M-1} \to \cdots \to \boldsymbol{s}^2 \to \boldsymbol{s}^1. \tag{4.54}$$

The last part needed to complete the backpropagation algorithm, the starting point ($\boldsymbol{s}^M$), should be obtained for the recurrence relation

$$
\begin{aligned}
S_i^M &= \frac{\partial \widehat{\boldsymbol{F}}}{\partial n_i^M} \\
&= \frac{\partial (\boldsymbol{t}-\boldsymbol{a})^T (\boldsymbol{t}-\boldsymbol{a})}{\partial n_i^M} \\
&= \frac{\partial \sum_{j=1}^{S^M} (t_j - a_j)^2}{\partial n_i^M} \\
&= -2(t_i - a_i) \frac{\partial a_i}{\partial n_i^M}.
\end{aligned} \tag{4.55}
$$

It can be written as

$$
s_i^M = -2(t_i - a_i)\dot{f}^M\left(n_i^M\right), \tag{4.56}
$$

because

$$
\frac{\partial a_i}{\partial n_i^M} = \frac{\partial a_i^M}{\partial n_i^M} = \frac{\partial f^M(n_i^M)}{\partial n_i^M} = \dot{f}^M\left(n_i^M\right). \tag{4.57}
$$

Finally, (4.56) can be rewritten in matrix form as

$$
s^M = -2\dot{\boldsymbol{F}}^M(\boldsymbol{n}^M)(\boldsymbol{t} - \boldsymbol{a}). \tag{4.58}
$$

The algorithm can be summarised into the following three steps, which were discussed in this section:

1. Propagate the input forward through the MLP, using (4.27), (4.28), and (4.29),
2. Propagate the sensitivities backwards through the MLP, as (4.53) and (4.58), and
3. Adjust the MLP's weights and biases by using the approximate stochastic gradient descent rule as provided in (4.45) and (4.46).

In the next section, an automated method for selecting an appropriate neural network architecture is presented.

4.5. Neural architecture search

Automated machine learning (AutoML) refers to the automation of machine learning model selection, hyperparameter optimisation, and model search (Guyon *et al.*, 2015). The process of automating neural network architecture engineering, known as Neural Architecture Search (NAS), is a subfield of AutoML and has created models that sometimes outperform those that were manually designed (Elsken *et al.*, 2019). An NAS method typically consists of three dimensions, i.e. search space, search strategy and performance estimation strategy. An abstraction of such a method is illustrated in Figure 4.5.

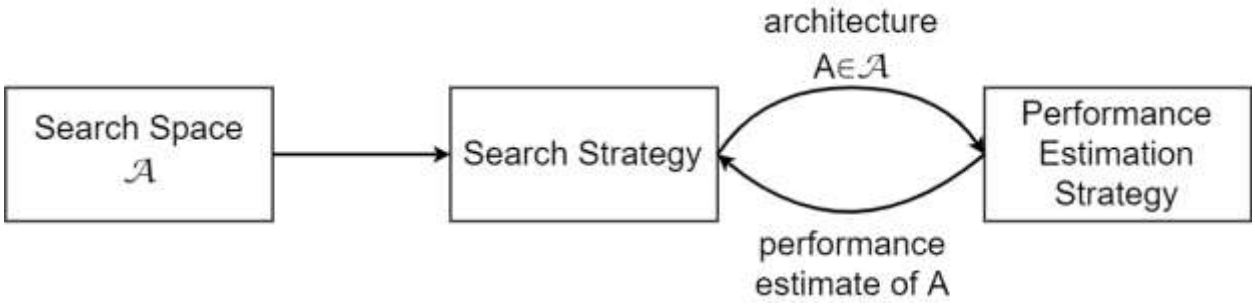


Figure 4.5: Abstract illustration of neural architecture search methods (Elsken *et al.*, 2019)

The search space ($\mathcal{A}$) defines all architectures that can be presented in principle. Its size can be decreased by including prior knowledge on architectures of similar tasks, but this introduces a human bias. A multilayer perceptron neural network and other chain-like neural networks' search spaces are parametrised by the maximum number of hidden layers (possibly unbounded), the operation of each layer and the hyperparameters associated with the operation. The difficulty of the architecture optimisation problem is mostly determined by the choice of the search space. However, the problem remains non-continuous and consists of many dimensions.

A search strategy is employed to explore the search space and find an architecture $A \in \mathcal{A}$, which is then passed to the performance estimation strategy. Architectures that perform well should preferably be found quickly and premature convergence to a region where suboptimal architectures exist should be avoided. Methods, such as random search, Bayesian optimisation, evolutionary methods, reinforcement learning, and gradient-based methods can be used to search for an appropriate architecture within the search space. In a study done by Real *et al.* (2019), a reinforcement learning, an evolutionary and a random search approach were compared. They found that the latter method was outperformed by both the former two approaches. Furthermore, the evolutionary method produced models with higher accuracies during the earlier stages of the process than the other two methods. Therefore, for the search strategy used in this study, an adapted version of a regularised evolution approach proposed by Real *et al.* (2019) was implemented to construct and select an appropriate architecture in each of the experiments. This method is summarised in Algorithm 4.1.

Algorithm 4.1: Regularised evolution search strategy

$population \leftarrow empty\ queue$
$history \leftarrow empty\ list$
while $|population| < P$ **do**
 $model.arch \leftarrow RANDOMARCHITECTURE()$
 $model.accuracy \leftarrow TRAINANDEVAL(model.arch)$
 add $model$ to right of $population$
 add $model$ to $history$
end while
while $|history| < C$ **do**
 $sample \leftarrow empty\ list$
 while $|sample| < S$ **do**
  $candidate \leftarrow$ distinct random element from $population$
  add $candidate$ to $sample$
 end while
 $parent \leftarrow$ highest accuracy model in $sample$
 $child.arch \leftarrow \text{MUTATE}(parent.arch)$
 $child.accuracy \leftarrow \text{TRAINANDEVAL}(child.arch)$
 add $child$ to right of $population$
 add $child$ to $history$
 remove $dead$ from the left of $population$
 discard $dead$
end while
return highest accuracy model in $history$

The method stores a population of models that had already been trained throughout the experiment. At the start of the experiment, P models with random architectures, based on the search space described above, are added to the population. Then C cycles are performed to mutate the population and add them to the history list. In each cycle, S random candidates are selected from the population. The candidate with the highest accuracy is then selected, mutated and trained, thereby creating a child model. A mutation modifies the selected architecture in a simple and randomised manner. This was done by randomising one or more of the architecture's hyperparameters. The child model is then added to the population and history. Finally, the population is adjusted to remove the oldest model from it. The performance estimator strategy was kept simple by trying to maximise the validation loss of the model. To ensure that the NAS method makes efficient use of the computing resources, the generated models were configured to stop early when the accuracy of the model started to converge during training.

Performance estimation strategies are implemented to estimate how an architecture A performs (Elsken *et al.*, 2019). The strategy then returns this estimation to the search strategy, which aims to maximise the performance measure. The most common performance measure that is used simply examines the accuracy obtained by the architecture on a validation data set after it has been trained. However, it can be time-intensive when using NAS. Thus, methods for speeding up the process have been studied, namely lower fidelity estimates, learning curve extrapolation, weight inheritance and one-shot models. In Table 4.2, an overview of studies on performance strategies is presented.

Table 4.2: Overview of studies on performance estimation strategies (Elsken *et al*., 2019)

Speed-up method	**Methods for improving the performance estimation strategy**	**References**
Lower fidelity estimates	Using subsets of the data set, downscaled models, and fewer epochs reduce training time.	Li *et al.* (2017), Zoph and Le (2016), Zela *et al.* (2018), Falkner *et al.* (2018), Real *et al.* (2019), Runge *et al.* (2018)
Learning curve extrapolation	Extrapolating performance from running only a few epochs decreases the time needed to train the model.	Swersky *et al.* (2014), Domhan *et al.* (2015), Klein *et al.* (2016), Baker *et al.* (2018)
Weight inheritance	Models inherit the weights from a parent model. Therefore, it is not necessary to train the models from scratch.	Real *et al.* (2017), Elsken *et al.* (2018b), Elsken *et al.* (2018a), Cai *et al.* (2018a); Cai *et al.* (2018b)
One-shot models	Weights are shared across different architectures that are subgraphs of the one-shot model.	Saxena and Verbeek (2016), Pham *et al.* (2018), Bender *et al.* (2018), Liu *et al.* (2019), Cai *et al.* (2018c), Xie *et al.* (2018)

Lower fidelity estimates refer to methods that train on a smaller subset of the data set, using fewer epochs, fewer filters or lower quality data, i.e. low-resolution images. However, these methods often underestimate the performance measure. Learning curve extrapolation methods train the model with fewer epochs, after which those models that are predicted to perform poorly when trained for a longer period are discarded. Weight inheritance methods involve initialising new models with the same weights of previously trained models. This allows the architecture to be changed without modifying the function that is represented by the model. Still, this can lead to models with extremely complex architectures. The last method, i.e. one-shot models, views all models as part of the same supergraph, sharing some of the edges. The edges that are shared are also assigned the same weights. Nevertheless, this method also underestimates the actual performance. Hence, it seems that learning curve extrapolation is the fastest and most accurate method for performance evaluation.

4.6. Performance evaluation measures

Flach (2019) states that making use of only one aggregated measurement to evaluate the performance of a machine learning model is inadequate. This is further supported in the literature, as Tripathy *et al.* (2016), Jiao and Du (2016), Gokgoz and Subasi (2015), Powers (2011) and Sokolova and Lapalme (2009) all make use of or suggest making use of various other performance metrics. These performance measures include the following:

- Accuracy;
- Precision;
- Recall or sensitivity; and
- F-measure, also sometimes referred to as the F_1-measure.

These methods make use of eight commonly used count values, i.e. real positives (RP), real negatives (RN), predictive positives (PP), predictive negatives (PN), true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN), where

$$RP = TP + FN, \tag{4.59}$$

$$RN = TN + FP, \tag{4.60}$$

$$PP = TP + FP, \tag{4.61}$$

$$PN = TN + FN, \tag{4.62}$$

$$\begin{aligned} n &= TP + TN + FP + FN \\ &= RP + RN \\ &= PP + PN. \end{aligned} \tag{4.63}$$

These counts can be visually represented using a confusion matrix, as shown in Figure 4.6. However, when making use of multiple classes, the confusion matrix should be adjusted to be similar to Figure 4.7 (Krüger, 2016). Similarly, the equations used to calculate the above-mentioned performance metrics have to be adapted to take the multiple classes into account. Thus, when defining the performance metrics below, both the single class and multiple class definitions will be provided (Sokolova & Lapalme, 2009). The multiclass equations, except for accuracy and the error rate, are further divided into micro- (μ) and macro-averaging (M) measures. All the classes contribute equal amounts to the specific measure when using macro-averaging; in other words, it averages the measures calculated for each of the classes. Consequently, such measures are biased towards classes containing fewer samples. In contrast, micro-averages are biased towards classes having more samples, since it gives equal weight to each sample in the data set.

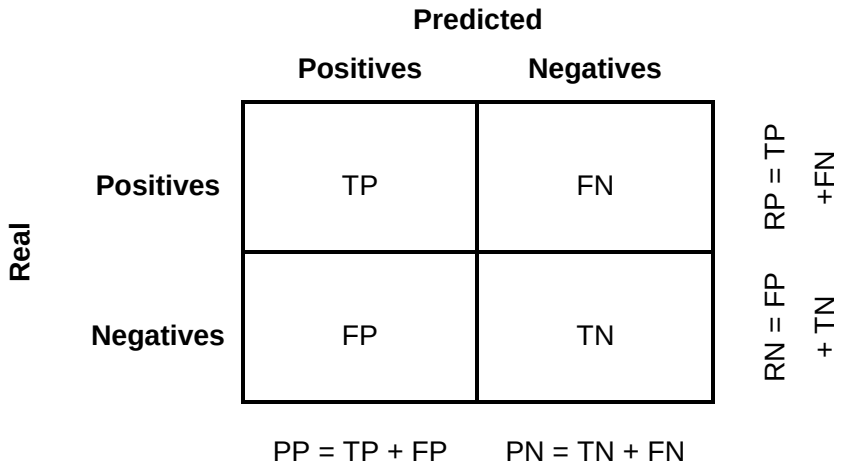


Figure 4.6: Confusion matrix for binary classification (Jiao & Du, 2016)

The performance measures mentioned above are all based on the values represented by the confusion matrix. Each of the measures is discussed below, and a definition for each is given in both the binary and multiclass classification form. Accuracy, the most commonly used measure, is applied to determine how well the model can classify both positive and negative samples correctly. This is done by calculating the ratio of correctly classified samples, both positive and negative, to the total number of samples. Therefore, it can be formally defined as follows:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}. \tag{4.64}$$

$$Average\ accuracy = \left(\sum_{i=1}^{n} \frac{tp_i+tn_i}{tp_i+tn_i+fp_i+fn_i}\right) \div n, \tag{4.65}$$

where n indicates the number of classes.

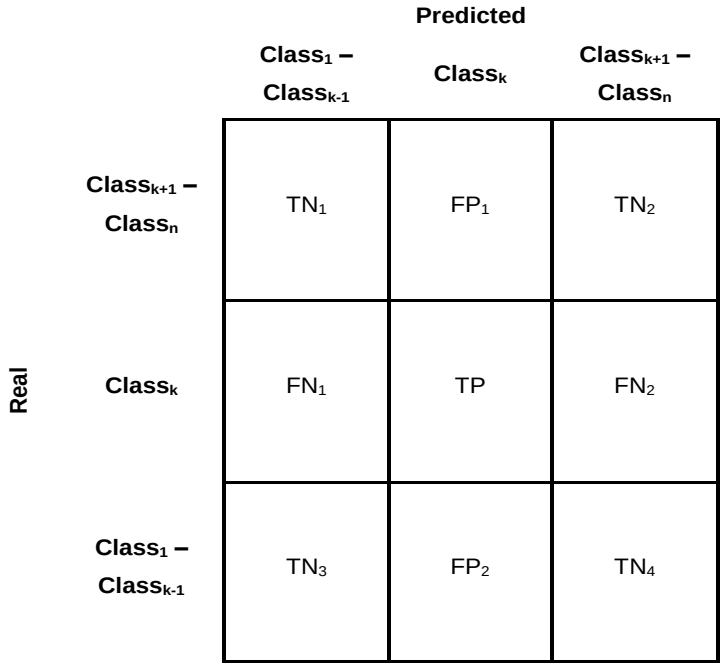


Figure 4.7: Multiclass confusion matrix (Krüger, 2016)

The error rate is used to indicate the frequency of errors that occurred during the prediction phase. It is given as

$$Error\ rate = \frac{FP+FN}{TP+TN+FP+FN}. \tag{4.66}$$

$$Average\ error\ rate = \left(\sum_{i=1}^{n} \frac{fp_i+fn_i}{tp_i+tn_i+fp_i+fn_i}\right) \div n, \tag{4.67}$$

Using the precision measure, insight can be gained into the proportion of correctly classified true positives in relation to the total number of predicted positives. Thus, its definition is given as

$$Precision = \frac{TP}{TP+FP}. \tag{4.68}$$

$$Precision_{\mu} = \frac{\sum_{i=1}^{n} tp_i}{\sum_{i=1}^{n} tp_i+fp_i}, \tag{4.69}$$

$$Precision_{M} = \left(\sum_{i=1}^{n} \frac{tp_i}{tp_i+fp_i}\right) \div n. \tag{4.70}$$

To measure the ratio of samples that were labelled as positive in comparison with all the truly positive samples, the recall measure can be used. Therefore, this measure indicates the model's completeness. It can be defined as follows:

$$Recall = \frac{TP}{TP+FN}, \tag{4.71}$$

$$Recall_{\mu} = \frac{\sum_{i=1}^{n} tp_i}{\sum_{i=1}^{n} tp_i+fn_i}, \tag{4.72}$$

and

$$Recall_M = \left(\sum_{i=1}^{n} \frac{tp_i}{tp_i + fn_i}\right) \div n. \quad (4.73)$$

The F-measure is often referred to as the “harmonic mean of precision and recall” and provides a measurement of how accurately a model performed on a test. This measure is defined as follows (Sasaki, 2007):

$$F = \frac{(\beta^2+1)\times Precision \times Recall}{\beta^2 \times Precision + Recall}, \quad (4.74)$$

where $0 \leq \beta \leq +\infty$,

$$F_\mu = \frac{(\beta^2+1)\times Precision_\mu \times Recall_\mu}{\beta^2 \times Precision_\mu + Recall_\mu}, \quad (4.75)$$

and

$$F_M = \frac{(\beta^2+1)\times Precision_M \times Recall_M}{\beta^2 \times Precision_M + Recall_M}. \quad (4.76)$$

Where β is used to control the weighting between precision and recall. If β is equal to 1, then F becomes the harmonic mean of precision and recall, as both measures have the same weight. However, F becomes more recall-oriented when β is greater than 1 and it becomes precision-oriented if β is less than 1.

4.7. Summary

Deep neural networks are useful for learning complex functions that are difficult to approximate using conventional solutions. For the experimental phase of this study, a deep MLP neural network will be constructed in order to classify collected affective data with the goal of modelling sentiment. In this chapter, background information on the topic of deep neural networks was provided. A discussion on the procedures used for gathering data, as well as developing and evaluating the deep learning MLP, is presented in Chapter 5.

Chapter 5 Experimental results

The true method of knowledge is experiment.

~William Blake

5.1. Introduction

The previous four chapters provided the context of the study and an overview of the background of the techniques that were applied within this study. The purpose of Chapter 5 is to discuss how the appropriate techniques were implemented in the development of a deep multilayer perceptron (MLP) neural network that performs sentiment analysis based on affective data. The experimental design of this study consisted of four parts, i.e. data acquisition, data pre-processing, the development and testing of a deep MLP. In Section 5.2, the data acquisition and data pre-processing techniques used in the experiments are presented. This is followed by a discussion on the automated method that was implemented to construct and select the best neural network architectures during the experiments. A pilot study was performed to test the feasibility of the study and to determine whether only relying on the six emotions identified by Ekman (1999) would create an accurate model. Therefore, two experiments were performed using a data set consisting of video recordings of nine participants reading text passages: one that creates a model with six inputs and one with 42 inputs. Respectively, these experiments are further referred to as Emotion6 and Metric42. The setup of and knowledge gained from the pilot study are considered in Section 5.3.

A follow-up experiment, described in Section 5.4, was conducted to extend the collected data set and improve on the results of the previous two experiments. This experiment, referred to as Advert22, made use of a data set collected by recording 22 participants who were watching video advertisements. However, a bias might have been introduced in this data set by the way the videos were selected. Therefore, two more experiments that made use of the pre-labelled Carnegie Mellon University Multimodal Opinion Sentiment Intensity (CMU-MOSI) data set, respectively CMU-MOSI1 and CMU-MOSI2, were conducted to limit this bias further. Both these experiments changed the continuous labels, ranging from -3 to +3, provided within the data set to three discrete labels, namely -1, 0 and +1. The former changed all the labels less than 0 to -1; all the values larger than 0 to +1 and left all 0s as 0. The latter grouped the labels -1 to -3 together under the new annotation -1; labels +1 to +3 as +1 and the remaining labels as 0. These two experiments are presented in Section 5.5.

The abovementioned experiments made use of the same neural architecture search (NAS) algorithm, as presented in Section 4.5, to find the best architecture for the deep learning

models. Each of the resulting models is then evaluated using the performance measures discussed in the previous chapter. Conclusions about the results obtained from the five experiments are presented alongside each experiment. An overarching discussion of the results obtained from all the experiments is presented in Section 5.6. Finally, the content of the chapter is summarised in Section 5.7.

It should be noted that in the discussion sections, positive classifications and predictions refer to correct classifications made by the model. Similarly, negative classifications and predictions refer to data points classified that were incorrect. These terms should not be confused with the positive and negative sentiment classes, which are the annotations assigned to data points.

5.2. Overview of experimental design

Standardised techniques were used across all five of the experiments to pre-process the data sets and to select an appropriate architecture for the deep MLP models. These techniques are discussed below.

5.2.1. Data acquisition

For the first three experiments, i.e. Emotion6, Metric42 and Advert22, data was acquired by recording participants while they were either reading or viewing content on their computers. The former two recorded nine participants reading three text passages, selected to evoke specific sentiments from them. In the latter, 22 participants were recorded while being shown three video advertisements to evoke the sentiments. Each data set was annotated using the expected sentiment from the text passages or video advertisements. The CMU-MOSI1 and CMU-MOSI2 experiments made use of the already existing CMU-MOSI data set, discussed in Chapter 2. This data set already contained continuous labels ranging between -3 and +3. As the goal of the study is to classify the sentiment into one of three sentiment classes, i.e. positive, neutral or negative, the labels had to be adapted accordingly, as discussed in Section 5.5.

5.2.2. Data pre-processing

The raw data that were used in the five experiments consisted of videos of people's faces expressing their opinions towards different entities, products, and advertisements. To use this data for this study, it was necessary to transform it to reflect affective data in a quantitative form. Affectiva©'s emotion software development kit (SDK) was used for this purpose, as its classifier has been trained on over 7.5 million faces of people originating from 87 countries, making it the world's largest emotion database (Affectiva, s.a.). Affectiva© provides 42 affective metrics representing emotions, facial expressions or emojis as listed in Table 5.1. Each metric's score

indicates the degree of confidence the SDK expresses that the user is showing a specific emotion, expression or emoji. The values range from 0, indicating no expression, to 100, representing that the expression is fully present. Valence is the only exception to this range, as its values range from -100 up to 100.

Table 5.1: Extracted metrics using the Affectiva© emotion SDK (Affectiva, 2017)

Emotions		
Metric	**Description**	**Example**
1. Anger	Having strong displeasure towards something, usually accompanied by a strong feeling of antagonism	
2. Contempt	Lacking respect for something	
3. Disgust	Aversion to something highly repugnant	
4. Fear	A strongly unpleasant emotion arising from the awareness or anticipation of danger	

Table 5.1: Extracted metrics using the Affectiva© emotion SDK (continued)

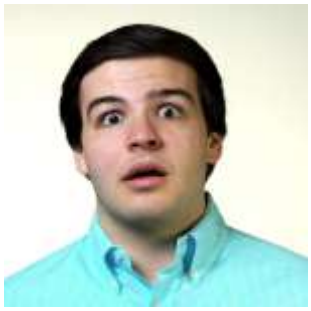
Metric	Description	Example
5. Joy	Expressing an emotion that is elicited by success, well-being, good fortune or by the achievement of one's desires	
6. Sadness	Expression of grief or unhappiness	
7. Surprise	A reaction to something unexpected or unusual	
Facial expressions		
8. Attention	A measure relying on the person's head orientation indicating his or her focus	
9. Brow furrow	The movement of both the eyebrows closer together and to a lower position	

Table 5.1: Extracted metrics using the Affectiva© emotion SDK (continued)

Metric	Description	Example
10. Brow raise	An upward movement of both eyebrows	
11. Cheek raise	Cheeks lifted to a higher position, often includes wrinkles at the eye corners known as crow's feet	
12. Chin raise	Moving the chin and lower lip upwards	
13. Dimpler	Tightening of the lip corners along with pulling them inwards	
14. Eye closure	Closing both eyelids	

Table 5.1: Extracted metrics using the Affectiva© emotion SDK (continued)

Metric	Description	Example
15. Eye widen	Exposing the entire iris by raising the upper lid	
16. Inner brow raise	Raising the inner corners of the eyebrows	
17. Jaw drop	Pulling the jaw downwards	
18. Lid tighten	Tightening the eyelids and narrowing the opening of the eyes	
19. Lip corner depressor	Dropping the lip corners downwards	

Table 5.1: Extracted metrics using the Affectiva© emotion SDK (continued)

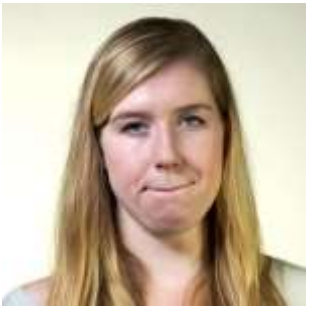
Metric	Description	Example
20. Lip press	Pressing the lips together without pushing up the chin boss	
21. Lip pucker	Pushing the lips forward	
22. Lip stretch	Pulling the lips back laterally	
23. Lip suck	Pull of the lips and the adjacent skin into the mouth	
24. Mouth open	Dropping the lower lip	

Table 5.1: Extracted metrics using the Affectiva© emotion SDK (continued)

Metric	Description	Example
25. Nose wrinkle	Pulling the skin of the nose upwards, causing wrinkles to appear on the sides of and across the nose	
26. Smile	Pulling the lip corners outwards and upwards towards the ears	
27. Smirk	Pulling either the left or the right lip corner upwards and outwards	
28. Upper lip raise	Upward movement of the upper lip	
Emojis		
29. Laughing	Opened mouth with closed eyes	

Table 5.1: Extracted metrics using the Affectiva© emotion SDK (continued)

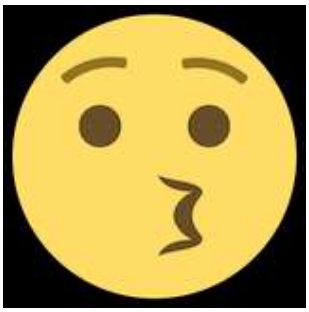
Metric	Description	Example
30. Smiley	Opened mouth with both eyes open and smiling	
31. Relaxed	Smiling with both eyes opened (with a closed mouth)	
32. Wink	A single closed eye	
33. Kissing	Puckered lips with both eyes opened	
34. Stuck out tongue	Visible tongue with both eyes open	

Table 5.1: Extracted metrics using the Affectiva© emotion SDK (continued)

Metric	Description	Example
35. Stuck out tongue and winking eye	Visible tongue with a single eye closed	
36. Scream	Open mouth with raised eyebrows	
37. Flushed	Widened eyes and raised eyebrows	
38. Smirk	Left or right lip corner pulled upwards and outwards	
39. Disappointed	Both lips pulled downward	

Table 5.1: Extracted metrics using the Affectiva© emotion SDK (continued)

Metric	Description	Example
40. Rage	Pressed and tightened lips with furrowed brows	
Other		
41. Engagement	A measure of the person's expressiveness based on facial muscle activation	No visualisation
42. Valence	An indication of the positive or negative nature of the person's experience	No visualisation

An application was created that implemented the Affectiva© emotion SDK to process the videos in each of the data sets for the five experiments. The 42 metrics were obtained for frames extracted, using a frame rate of 60 frames per second from each of the videos in the data sets.

The data points were then annotated to indicate the sentiment class, i.e. positive, neutral or negative, that is represented by it, as discussed in each experiment below. Finally, the data points were shuffled and divided into the training, validation and test data sets, using a ratio of 70%, 20% and 10%, respectively.

5.2.3. Selection of the neural network architecture

The Google Colaboratory platform, or simply Google Colab, was used for finding and selecting the respective architectures for each experiment. It is a cloud-based Jupyter notebook environment that requires minimal to no setup. This system consists of two 2.30GHz Intel(R) Xeon(R) CPUs with 13.3GB RAM and a Tensor Processing Unit. Furthermore, Keras version 2.2.5 with a Tensorflow version 1.15.0 backend and Python 3 were used to implement the search algorithm.

The neural architecture search method, specifically the regularised evolution search strategy, as discussed in Chapter 4, was used to automate the construction of the deep MLP architecture. This method can be summarised into the following steps (Elsken *et al.*, 2019b):

1. Determine the search space boundaries which define all the possible architectures that may be explored;
2. Implement a search strategy to find an architecture with the best performance from the search space;
3. Estimate the performance of the selected architecture and return the performance measure to the search strategy;
4. Repeat Step 2 and Step 3 until a specified number of cycles (C) has been completed; and
5. Select the architecture with the best performance measure.

Several restrictions were identified for the search space of this study to ensure that it could be adequately explored within a time- and resource-limited environment. Firstly, all hidden layers had to use the *rectified linear unit* (*ReLU*) activation function, as it can achieve sparsity levels similar to neurons within the human brain, and it is not as computationally intensive as other activation functions (Ding *et al.*, 2018). The output layer used the *softmax* activation function to estimate the probability that the given data points can be classified as one of the three sentiment classes (positive, neutral and negative). To manage the complexity of the generated architectures, the number of hidden layers that can exist in a model was limited to 10 layers. A lower bound of three layers was also implemented to ensure that deep models were created, as less than three layers can be considered as shallow. Similarly, each hidden layer could only contain between 10 and 256 nodes[5]. A drop-out layer followed each hidden layer with a drop-out rate ranging between 0.2 and 0.5. The batch size used for training was also limited to a value between powers of two between 32 and 4 096. Lastly, the learning rate was set to 0.01.

The evolutionary algorithm, presented in Chapter 4, Algorithm 4.1, was implemented as the search strategy to identify, train and evaluate the deep neural networks used in each experiment. It initially constructed 200 deep MLP models to add to the *population* queue, by randomising the number of layers and nodes within each layer, as well as the drop-out rate of the models and the batch size used for training the model. For each of these models, accuracy was calculated based on how well the model classified the validation data set. This initialisation was followed by 1 800 cycles. During each of these cycles, 50 distinct models were selected at random from the population as samples. The best performing model from this sample was then selected and mutated, i.e. its architecture was adapted. In this phase, either one or two of the model's hyperparameters, as mentioned above, were adjusted. The limitations set out in the space boundary definition were also applied to this phase. Once the mutation had been made,

[5] Additional experiments were performed having the maximum number of nodes set to both higher and lower values, but setting it to 256 nodes yielded the best results.

the new model was trained, evaluated and added to both the *history* list and *population* queue. Subsequently, the oldest model in the population was removed.

The algorithm was run until the storage capacity of 15GB on Google Drive, where the models were stored, was exceeded. After the storage capacity was exceeded, the model with the highest accuracy on the validation data set was selected to be evaluated on the test data set for each of the five experiments. These experiments are discussed next.

5.3. Experiments using the data set with nine participants

At the onset of this study, a pilot study consisting of two experiments was performed to determine the feasibility of applying the identified techniques, i.e. affective computing for facial expression recognition and deep learning using an MLP, to the problem of sentiment analysis. In this section, an overview of these two experiments is provided.

5.3.1. Data set description

A group of nine students studying towards their honours or master's degree in Computer Science was used to generate a data set consisting of videos containing their facial expressions. The faces of the participants were recorded while they read three text passages. Each passage was chosen with the intent of prompting a specific sentiment from the participants. A list of jokes was used to elicit a positive sentiment, and a neutral news article was selected to prompt a neutral sentiment. To evoke a negative sentiment from the participants, without exposing them to controversial or harmful content, a news article on the penalties one can face if caught illegally downloading content online was presented to them. Examples of the facial reactions of the participants toward each of the text passages are shown in Figure 5.1.

The videos were then pre-processed as discussed in Section 5.2.1, resulting in 132 261 data points with 42 features each, which were annotated based on the intended elicited sentiment of the text passage. For example, if the intended sentiment of the video was positive, a label indicating that the sentiment was positive was assigned to the data point. If the intended sentiment was neutral, a neutral label was used and similarly for an intended negative sentiment it was annotated as negative. In Table 5.2, the distribution of the classes among the videos and data points are shown. This data set was then randomised and divided into training (70%), validation (20%), and testing (10%) subsets at frame level to be used during training.

(a) Positive

(b) Neutral

(c) Negative

Figure 5.1: Example: facial reactions to each of the three text passages

Table 5.2: Distribution of sentiment classes in the pilot study data set

Category	Videos	Data points
Positive	9	41934
Neutral	9	54873
Negative	9	35454
Total	27	132261

These data points were then used during the modelling, training and selection of the deep MLP neural network, presented next. The first variation of the data set is further referred to as Metric42, as it made use of the 42 metrics provided by the Affectiva© emotion SDK. Since the second variation only uses the six emotions proposed by Ekman, it is called Emotion6.

5.3.2. Selected architectures and results

Two deep MLP models, differing in the number of input nodes, were constructed to classify the data set that was discussed in the previous section. This was done to determine whether an MLP would perform better when trained using only the six universal emotions that Ekman

(1999) proposed or all 42 metrics extracted using Affectiva. Both architectures of the models were selected using the regularised evolution search strategy.

5.3.2.1. Metric42 model

After 24699.329 minutes, a total of 2000 models was generated. The best-performing model found using the neural architecture search algorithm for this data set, consisted of an input layer with 42 input nodes and an output layer with three nodes using the *softmax* activation function. Eight hidden layers implementing the *ReLU* activation function were used, as visualised in Figure 5.2 with each layer's corresponding drop-out rates shown in Table 5.3. The model was trained using a batch size of 4 096 with 100 epochs and ran for 8.697 minutes.

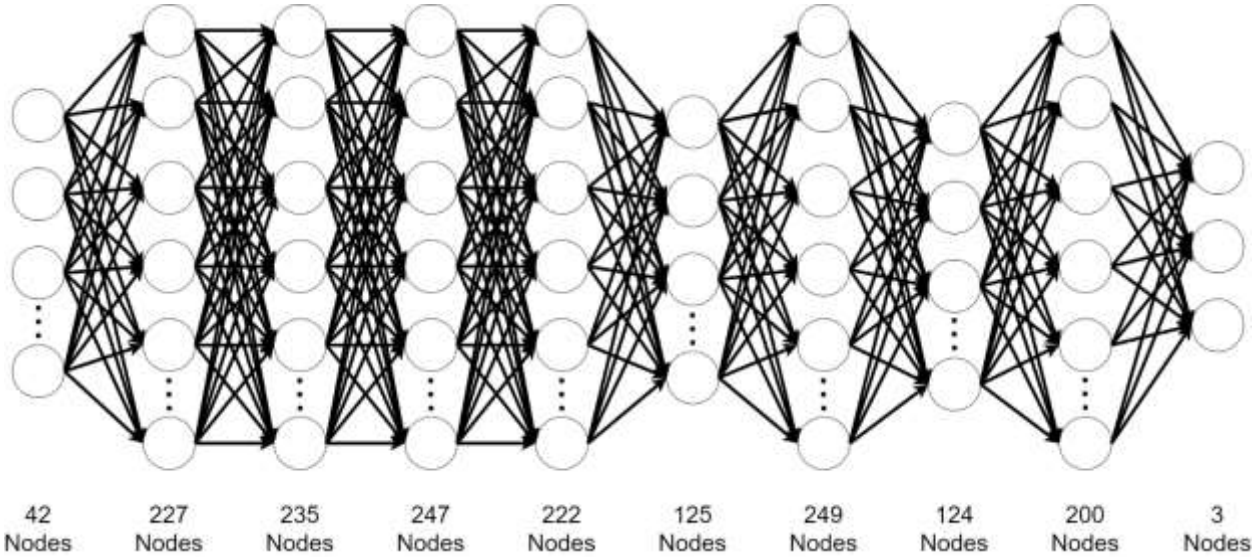


Figure 5.2: MLP architecture for the Metric42 model

Table 5.3: Summary of the drop-out rates for each layer of the Metric42 model

Hidden layer number	Drop-out rate
1	0.220
2	0.214
3	0.210
4	0.316
5	0.376
6	0.488
7	0.432
8	0.288

A validation accuracy and loss of 99.221% and 0.033 were respectively obtained during the training phase of this model. During the testing phase, it scored a test accuracy of 99.251% and a loss value of 0.039. The performance metrics obtained by evaluating the model using the test data set are presented in the confusion matrix in Table 5.4 and further interpreted by calculating the performance measures shown in Table 5.5.

Table 5.4: Confusion matrix for the Metric42 model

		Predicted sentiment		
		Positive	**Neutral**	**Negative**
Actual sentiment	**Positive**	8404	19	20
	Neutral	47	10870	28
	Negative	38	46	6980

Table 5.5: Performance measures for Metric42

	Class-specific measures			**Macro-Averaging (M)**	**Micro-averaging (μ)**
	Positive	**Neutral**	**Negative**		
Accuracy (%)	99.531	99.471	99.501	99.501	N/A
Error rate (%)	0.469	0.529	0.499	0.499	N/A
Precision (%)	98.999	99.406	99.317	99.240	99.251
Recall (%)	99.538	99.315	98.811	99.221	99.251
F-score (%)	99.268	99.360	99.063	99.231	99.251

5.3.2.2. Emotion6 model

The second model was constructed using the same collected data set as with the Metric42 model, but only taking the six emotions identified by Ekman (1999) into account. This was done to determine whether only the six emotions could be used to model sentiment. Similar to the first model, it also had an output layer consisting of three nodes using the *softmax* activation function, but only six input nodes representing each of the six previously mentioned emotions. This model, as presented in Figure 5.3, also contained six hidden layers. A summary of the drop-out rates for each of the layers of this model is presented in Table 5.6. The model was trained using 55 epochs and a batch size of 4 096 with a runtime of 1.642 minutes.

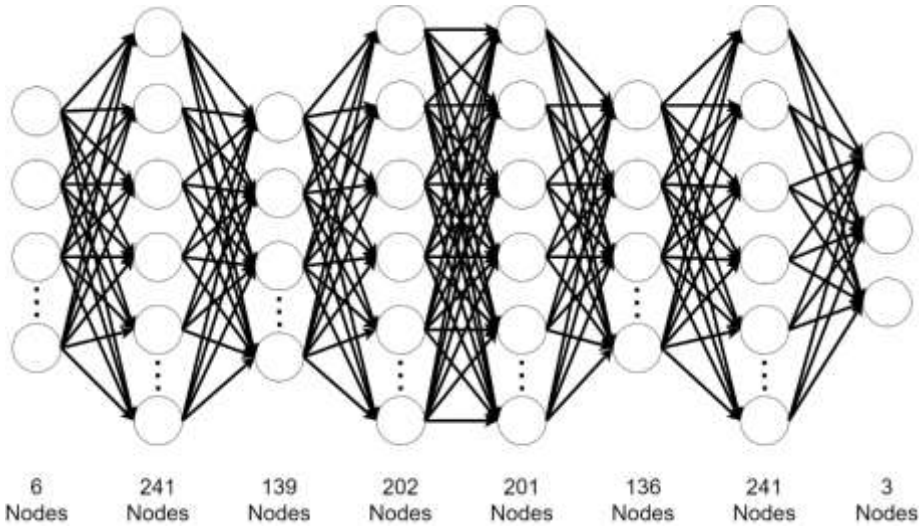


Figure 5.3: MLP architecture for the Emotion6 model

Table 5.6: Summary of the drop-out rates for each layer of the Emotion6 model

Hidden layer number	Drop-out rate
1	0.214
2	0.212
3	0.214
4	0.247
5	0.367
6	0.229

The selected model obtained a 65.643% validation accuracy and 0.725 validation loss, as well as test accuracy and loss of 66.766% and 0.717, respectively. In Table 5.7, the performance metrics obtained from evaluating this model's performance using the test data set are summarised in a confusion matrix. It is also further interpreted in Table 5.8 by providing performance measures.

Table 5.7: Confusion matrix for the Emotion6 model

		Predicted sentiment		
		Positive	**Neutral**	**Negative**
Actual sentiment	**Positive**	5889	2212	269
	Neutral	587	9862	471
	Negative	532	4720	1910

Table 5.8: Performance measures for the Emotion6 model

	Class-specific measures			Macro-Averaging (M)	Micro-averaging (μ)
	Positive	**Neutral**	**Negative**		
Accuracy (%)	86.390	69.794	77.348	77.844	N/A
Error rate (%)	13.610	30.206	22.652	22.156	N/A
Precision (%)	84.033	58.723	72.075	71.610	66.766
Recall (%)	70.358	90.311	26.669	62.446	66.766
F-score (%)	76.590	71.170	38.932	66.715	66.766

5.3.3. Discussion

These first two experiments, which were run in parallel, were intended as a pilot study with a two-fold purpose. The first was to determine the feasibility of continuing with the study, while the second purpose was to determine whether a model trained using data on the six emotions identified by Ekman (1999) would give adequate results.

5.3.3.1. Metric42 model evaluation

Inspecting the performance measures listed in Table 5.5, showed that the Metric42 model reached values for the accuracy, precision, recall and F-score that are all close to 100%. It was able to correctly identify the vast majority of examples into the correct sentiment class, with merely 198 out of 26452 data points being predicted incorrectly. Though high values are desired for the performance measures of a model, these near-perfect values obtained by this model seemed unusual. This observation is indicative that the model may have overfitted during training. One possible cause of this may relate to the number of data points available in the data set. Since there were only nine participants present in this data set, there may have been limited variation within the data set. Another cause relates to how the data set was split into the training, validation and test subsets. The original data set was randomised and then split into the three subsets at frame level. Sequential frames taken milliseconds apart would be very similar or even identical to one another and would end up in different data sets. Consequently, the model would be able to accurately classify these data points, as it would already be familiar with these similar data points.

5.3.3.2. Emotion6 model evaluation

The second experiment in the pilot study was used to determine whether sufficient results could be obtained by using only the six emotions identified by Ekman. Hence, the same data set with nine participants was used with a minor adjustment made to it, i.e. the 36 additional metrics produced by Affectiva© were removed, leaving only the data on the six emotions.

The best model was slightly less complex than the best Metric42 model. Regardless, this less complex model obtained a high accuracy, indicating that it made the correct classification for the majority of the data points in the test data set. Similarly, it obtained a macro precision of 71.610%, which indicates that at the class level, the data points that were classified as positive predictions were mostly correct. The micro precision value gives more emphasis to the class that contains the most data points, in this case, the neutral sentiment class. In contrast, the macro precision is more biased towards the class with the lowest number of data points, which in this case is the negative sentiment class. Thus, the micro-average precision is 6% lower than the macro-average precision.

The macro-recall value was around 9% lower, which suggests that the model had difficulty in identifying all data points that should have been predicted positively. The micro-average values depict a similar situation, i.e. at frame level, this model could identify more than half the positive classifications, but not as well as the Metric42 model could. This conclusion is also supported by both the micro- and macro-F-scores.

5.3.3.3. Conclusions of the pilot study

While the Metric42 and Emotion6 models, which formed part of a pilot study, did contain flaws in how the training, validation and test data sets were compiled, it did also serve its purpose to determine whether using the proposed techniques were feasible. The results obtained from these models showed that using the data extracted from Affectiva© to train an MLP model to perform sentiment analysis can lead to viable results. Additionally, though the Emotion6 model did obtain high accuracy, its other performance measures were much lower. Specifically, by comparing the F-scores of both models, it is clear that the Metric42 model outperformed the Emotion6 model, since it can both correctly classify the majority of the instances and misclassifies the minimum number of instances. This result indicated that using only the six emotions identified by Ekman (1999) to train an MLP model would not deliver the same level of results as a model trained on the 42 metrics extracted by Affectiva.

The collected data set that was used for training the Metric42 and Emotion6 models only contained the reactions of nine participants. As it was only used as part of a pilot study, its size was adequate to determine the feasibility of the proposed techniques. However, to ensure that future experiments made use of a data set that contained a wider variety of reactions, the number of participants needed to be increased. Moreover, by dividing the data set at frame level led to the models overfitting when being trained.

Thus, a few adjustments needed to be made to the data collection and processing techniques of the next experiment. Firstly, the number of participants needed to be increased to expand the data set. Next, the possible bias that may be introduced by the researcher during the annotation process had to be minimised. The method of dividing the data set into training, validation and test data sets also needed to be revised to be done at video level instead of at frame level. Finally, the data sets used in the next experiments would contain the data extracted about all 42 Affectiva© metrics.

5.4. Experiment using a data set with 22 participants

To address the issues identified during the first experiment, a second experiment was executed. The data set for this experiment was improved by including more participants and changing the format of the material that the participants viewed. It was then annotated using the sentiment experienced by the participants and not according to the sentiment that was expected from viewing the material. Furthermore, to prevent overfitting during training the model, the training, validation, and test data sets were split at video level instead of at frame level. The collected data set, architecture and results for the second experiment are described in this section.

5.4.1. Data set description

A group of 25 students in the second-, third-, and honours year of their BSc in Information Technology studies was initially used in the data collection process. A web application was created using HTML5, JavaScript, and CSS to record the reactions of participants with a webcam while they watched three video advertisements for technology-based products. Screenshots of the web application can be seen in Annexure A. Video advertisements for technology-based products were selected for the following reasons:

1. In the case where participants had to read text passages, the duration of video recordings differed vastly, as it was dependent on the participant's reading pace. However, by using videos the duration, and thereby the number of collected frames, can be made more constant, as the participants viewed videos of the same length;
2. It can be assumed that technology-based products would typically resonate with students studying Information Technology;
3. Emotional appeal is often used in advertisements (Mogaji *et al.*, 2018; Li *et al.*, 2018);
4. Video advertisements are widely available online; and
5. Similar studies by McDuff *et al.* (2014a, 2014b) showed that video advertisements are good media to evoke and measure participants' sentiment.

Each of the videos used during the data collection phase was chosen to elicit a specific sentiment reaction from the participants, i.e. positive, neutral, and negative, and had approximately the same duration. The videos that were used were determined by searching for advertisements with keywords such as “good tech advertisements”, “positive advertisements” for the positive video, “bad advertisements”, “negative advertisements” for the negative video, and “neutral advertisements” for the neutral video. Web articles which contained references to the above-mentioned keywords and, consequently, the chosen videos, were then further inspected to determine the overall reaction of the audiences who were originally initially targeted by the advertisements. Only those advertisements where the original audience had a reaction corresponding to the sentiment classes that were to be evoked during the experiment were selected.

The video selected for arousing a positive sentiment was an Amazon advertisement where they showcased their Alexa virtual assistant while making fun of it (Amazon, 2019). An Amazon employee is shown telling a visitor about their failed attempts to incorporate Alexa into other products, such as a toothbrush, a dog collar that understands the dogs’ commands, a hot tub, and even in a satellite. All these products had laughable consequences, including the dog ordering dog food and gravy, and the satellite switching the power off on Earth, instead of on the satellite.

A Facebook advertisement was used for the neutral sentiment video (Facebook, 2014). In this advertisement, a neutral, almost bored-sounding, voice lists the benefits and functional characteristics of chairs. For example, chairs are used by people to sit down and take a break, and larger chairs can accommodate multiple persons so they can sit down together and share stories. It goes further to list things that are used by people to connect to each other and claims that Facebook can be used for the same reason.

The last sentiment, i.e. negative, was evoked using an advertisement for the Sony PlayStation 2 (Playstation, 2007). The monochrome advertisement starts with a man walking into a hallway where unexplainable things happen, such as an incomprehensible voice over an intercom, his head floating off of his body, and a dismembered arm coming out of his mouth. He finally enters a room with a coach where he sees himself, a mummy, a duck, and the aforementioned arm sitting on a coach. The advertisement ends with the duck saying, “Welcome to the third place”, leaving the viewer with an eerie feeling.

At the start of the recording session, the participants were asked to provide consent to be recorded. However, they were not informed at this time about which metrics from the resulting videos would be used for the study. This was to prevent them from possibly becoming self-

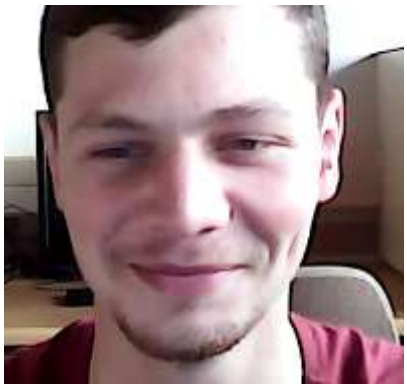
aware about their facial expressions and respond in a manner in which they usually would not. The video advertisements were then played in the order negative, neutral, positive while videos of the faces of the participants watching them were recorded. After watching the three selected video advertisements, the participants were asked to indicate how they experienced each video, in order to verify whether the intended reactions were successfully aroused. The responses of the participants for each video are summarised in Table 5.9. The majority, i.e. 59.1%, of the participants, agreed that they negatively experienced the video that was selected to evoke a negative sentiment. Similarly, 86.4% of the participants had a positive sentiment towards the positive sentiment video. However, their experience of the video that was selected to evoke a neutral sentiment was not as expected. Only 27.3% of the participants reacted neutrally to this video, with a majority of 59.1% reacting positively.

Table 5.9: Summary of participant responses

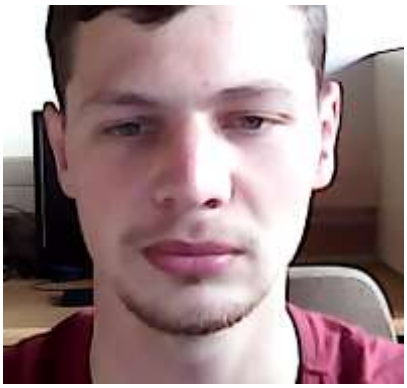
		Responded sentiment		
		Positive	**Neutral**	**Negative**
Expected sentiment	**Negative video**	2	7	13
	Neutral video	13	6	3
	Positive video	19	2	1

At the end of the session, the participants were informed about which metrics would be extracted from the videos and were given the option to review the videos recorded of them. Should they no longer have wished to submit these videos, they were removed from the data set. An example of the reactions of one of the participants is shown in Figure 5.4.

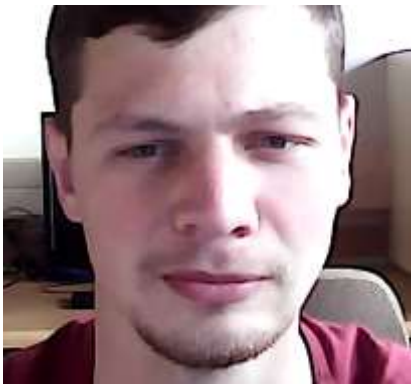
The collected videos were pre-processed, as explained in Section 5.2.1, to create a quantitative data set. Two of the students were, however, left out because the faces of the participants were unclear and could not be processed by the SDK. A third participant did not provide his opinion and was also left out of the final data set. The set of videos was then divided into training (70%), validation (20%), and testing (10%) subsets at video level before being randomised to prevent the models form overfitting during training. Each data point was annotated based on the self-reported responses of the respective candidate. The distribution of the sentiment classes in this data set is summarised in Table 5.10.



(a) Positive



(b) Neutral



(c) Negative

Figure 5.4: Example: reactions to each of the three video advertisements

Table 5.10: Distribution of sentiment classes in the 22 participants collected data set

Category	**Videos**	**Data points**
Positive	34	91666
Neutral	15	33706
Negative	17	34533
Total	66	159905

This data set, therefore, expands and improves upon the data set discussed in the previous section by, firstly, increasing the number of participants present in the recorded videos to increase variation within the data set. Next, the annotations were based purely on what the participants experienced. As a result, any bias that might have been previously introduced by the researcher was kept to a minimum. Lastly, the data set was divided into training, validation, and testing subsets at video level before being randomised.

This data set will further be referred to as Advert22, derived from the fact that the data set consists of 22 participants watching video advertisements. An extract of the data set can be seen in Annexure B.

5.4.2. Selected architecture and results for Advert22 model

A total of 1 109 models were generated in a time of 6 326.76 minutes. The deep MLP architecture selected by the neural architecture search for the data set described above consisted of an input layer with 42 nodes, eight hidden layers and an output layer with three nodes, as visualised in Figure 5.5. A summary of the drop-out rate used for each of the hidden layers is provided in Table 5.11. The model was trained, using a batch size of 4 096 and stopped training after 51 epochs due to its validation loss, reaching a plateau. This process took place over just 2.412 minutes.

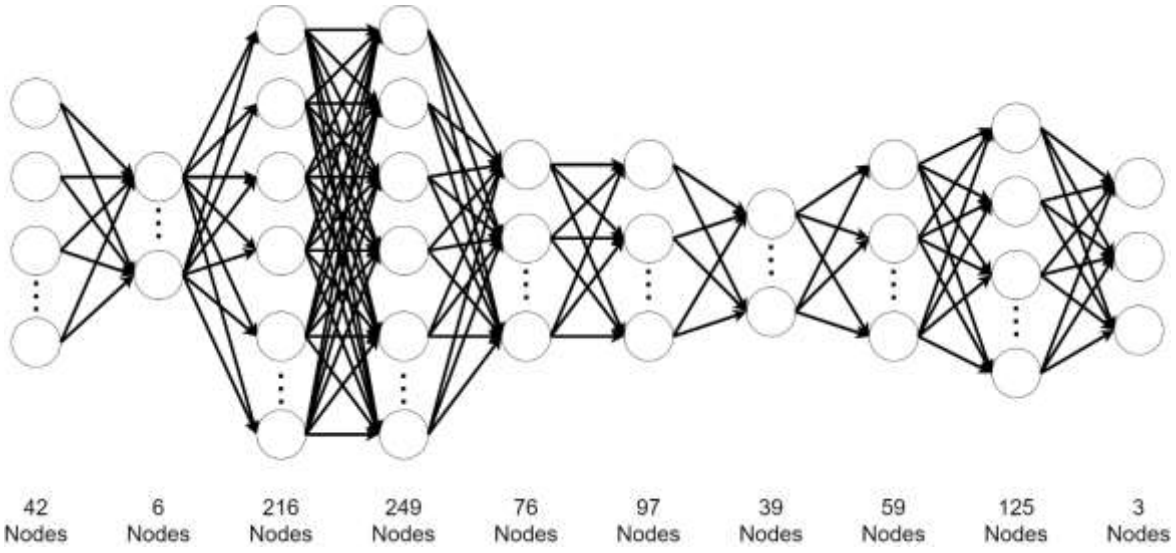


Figure 5.5: MLP architecture for the Advert22 model

Table 5.11: Summary of the drop-out rates for each layer of the Advert22 model

Hidden layer number	Drop-out rate
1	0.401
2	0.472
3	0.263
4	0.414
5	0.468
6	0.202
7	0.277
8	0.211

During training, this model achieved a validation accuracy of 60.207% and a validation loss of 0.016. The model was then assessed by providing it with the test data set to classify each of the data points contained in it. Test accuracy and loss of 73.978% and 1.143, were obtained, respectively. In Table 5.12, a confusion matrix is provided to offer the results obtained from analysing the test data set.

The micro- and macro-averaging performance measures (discussed in the previous chapter), as well as the class-specific accuracies and error rates which can be calculated using the confusion matrix, are presented in Table 5.13.

Table 5.12: Confusion matrix for the Advert22 model

		Predicted sentiment		
		Positive	**Neutral**	**Negative**
Actual sentiment	**Positive**	12591	170	190
	Neutral	2432	197	95
	Negative	1533	143	184

Table 5.13: Performance measures for the Advert22 model

	Class-specific measures			**Macro-averaging (M)**	**Micro-averaging (μ)**
	Positive	**Neutral**	**Negative**		
Accuracy (%)	75.335	83.804	88.817	82.652	N/A
Error rate (%)	24.665	16.196	11.183	17.348	N/A
Precision (%)	76.051	38.627	39.232	51.304	73.978
Recall (%)	97.220	7.232	9.892	38.115	73.978
F-score (%)	85.342	12.183	15.801	43.737	73.978

5.4.3. Discussion

This experiment built on the findings reported from the pilot study, and some improvements were made to the data collection and pre-processing stages.

5.4.3.1. Advert22 model evaluation

The selected Advert22 model is less complex than the models used in the previous two experiments. Based on the values obtained for the micro-average performance measures and the accuracy of the model, it seems that this model performs well overall. However, it is

misleading, as can be seen upon inspecting the macro-average performance measures. These values are more than 20% less than the micro-average values. Therefore, it suggests that the classes containing fewer data points performed much worse than those that contained more data points. The low precision and recall suggest only a small number of the predictions made by the model to be positive were indeed correct. In addition, it was not able to correctly identify all the data points that should have been classified as positive. Examining the performance of the individual sentiment classes clarifies why this was the case.

The positive sentiment class, which contained the majority of data points in the data set, had a high precision rate and an even higher recall rate. Thus, the model had almost no problem classifying data points belonging to this sentiment class. However, the neutral and negative sentiment classes had recall rates of less than 10% and precision rates of less than 40%, thereby causing the poor macro-average performance measures. These results also lead to the conclusion that this model would not be efficient to perform the task of sentiment analysis, as it is prone to misclassify two of the three sentiment classes.

5.4.3.2. Conclusions for the experiment

The data set used to create the Advert22 model was set up in such a way as to address the deficiencies of the one used in the pilot study, namely the number of participants was increased from nine to 22. Once again, it has proven challenging to get enough participants to create a data set of adequate size. From the limited number of people who were targeted and who indicated their interest in participating, an even smaller number showed up to the data acquisition session. The issues mentioned above led to the following problems with the data set:

1. The size of the data set could not be increased substantially compared to the previously collected data set;
2. Diversity within the group stayed mostly the same, i.e. a majority of male students studying towards the same degree; and
3. Relying on self-reported responses for annotating the data points in the data set, caused an imbalance in the number of data points per class.

Additionally, the training, validation and test data sets were divided at video level and the data points were labelled, using the responses of the participants. By these means, the model did not overfit during training and was not influenced by bias introduced by the researcher. The macro-average performance measures were low, indicating that the model had difficulty in classifying data points belonging to classes with fewer data points. Still, it did perform well overall, as

indicated by the average and micro-average values. These results indicated that the proposed techniques are usable for predicting sentiment, based on the facial expressions of a person.

It was therefore decided to make use of an already-existing data set which is widely used in the literature, instead. Its data points also needed to be pre-annotated in order to reduce human bias in the model; it needed to contain a larger number of videos with a diverse group of subjects or participants.

5.5. Experiments using the CMU-MOSI data set

The final experiments were performed, using the Carnegie Mellon University Multimodal Opinion Sentiment Intensity (CMU-MOSI) data set (Zadeh, 2016b) to address the inadequacies of the data sets used in the previous two experiments. The reasons for choosing this data set are as follows:

1. The number of unique people appearing in the videos is increased to 89 instead of only 22 participants in the previous data set;
2. A large number of videos with different labels, i.e. 2 199 segmented videos, are included in the data set;
3. Diversity in the people who appear in the videos is higher than in the previously collected data sets;
4. Labels were assigned to the videos by five independent workers. This process minimised the chance of introducing bias from either the researcher or the participants to the model; and
5. Literature indicates that it has been widely used for creating models that perform multimodal sentiment analysis.

In this section, the experiment using this data set is considered.

5.5.1. Data set description

The CMU-MOSI data set was presented by Zadeh *et al.* (2016b) who claimed it to be the first data set that can be used for multimodal sentiment intensity analysis research. As was seen in Chapter 2, Section 2.5, the data set is also widely used within this field (Junior & dos Santos, 2018; Tian *et al.*, 2018; Zadeh *et al.*, 2018b; Poria *et al.*, 2017b; Zadeh *et al.*, 2016a). The data set consists of self-recorded YouTube videos of people expressing their opinions on a variety of subjects using diverse setups. The speakers, who came from different ethnicities, had ages that ranged between 20 and 30 years old and 41 females and 48 males were included. A total of 93 full videos from 89 distinct speakers varying in length from two to five minutes were compiled.

The videos were segmented at utterance level resulting in 2 199 segments that contained a single expression of the subject's sentiment. Each segment was then manually annotated by five workers for the sentiment expressed by the speaker. Each annotator was given the simple instruction phrased as "How would you rate the sentiment expressed in this video segment? (Please note that you may or may not agree with what the speaker says. It is imperative that you only rate the sentiment stated by the speaker, not your opinion.)". They then had to select from eight choices, i.e. strongly positive (labelled as +3), positive (+2), weakly positive (+1), neutral (0), weakly negative (-1), negative (-2), strongly negative (-3), and uncertain. The responses of the annotators were then averaged before being assigned to the corresponding video segment.

The aim of this study was to classify the data into one of three sentiment classes, i.e. positive, neutral or negative. Hence, the labels had to be adapted to only include the three classes. To do this, all the original labels that had values greater than zero were relabelled as positive, all the values less than zero were changed to negative and the remaining values, i.e. those values equal to zero, were annotated as neutral. Two of the segmented videos were left out of the final data set, as their annotations were missing. In Table 5.14, a summary of the total number of data points, as well as videos that can be grouped into each category, is shown for the CMU-MOSI data set. The data set processed using this approach will further on be referred to as CMU-MOSI1. An example of each class for this data set, i.e. positive, neutral and negative sentiment, is presented in Figure 5.6.

As shown in Table 5.14, the number of data points is unevenly distributed among the three sentiment classes. Specifically, the neutral class had 109 383 data points less than the positive class, and 132 067 data points less than the negative class. Therefore, a second approach to annotating the data set was used to distribute the data points more evenly among the classes, to determine whether doing so could have a positive effect on the prediction accuracy of the model. With this approach, all labels having a value less than -1 were annotated as negative, and all labels having a value greater than +1 were annotated as positive. All values ranging between -1 and +1 received a label of neutral. The distribution of the data points when divided using this approach is summarised in Table 5.15. The data set resulting from this approach will further be referred to as CMU-MOSI2.

(a) Positive

(b) Neutral

(c) Negative

Figure 5.6: Example: sentiment displayed by subjects in CMU-MOSI data set videos

Table 5.14: Distribution of sentiment classes in CMU-MOSI1

Category	**Video segments**	**Data points**
Positive	1079	119073
Neutral	96	9690
Negative	1023	141757
Total	2198	264628

Table 5.15: Distribution of sentiment classes in CMU-MOSI2

Category	**Video segments**	**Data points**
Positive	719	74478
Neutral	854	104103
Negative	625	86047
Total	2198	264628

A deep MLP model was developed, using the neural architecture search algorithm discussed in Section 5.2.3. Both the CMU-MOSI1 and CMU-MOSI2 data sets were divided into training (70%), validation (20%) and testing (10%) data sets at video level, which were then used to train, evaluate, and test each model.

To summarise, by using the CMU-MOSI data set, several improvements were made from the previous experiments. Firstly, the number of unique faces present in the data set was increased immensely. The variety of backgrounds, as well as the number of males and females, present among these people also improved significantly as opposed to previously having a data set consisting of mostly male Computer Science students. An increase in the number of extracted frames was also achieved. Additionally, having the data set pre-annotated by independent workers limits the human bias introduced from either the participants or the researcher. Lastly, this data set has been widely used in the field of multimodal sentiment analysis, showing its applicability to the problem.

5.5.2. Selected architecture and results

The two data sets described above were used to train two separate models, as presented in this section.

5.5.2.1. CMU-MOSI1 model

During a period of 10365.747 minutes, the NAS algorithm generated 626 models using the CMU-MOSI1 data set. The model that performed the best among the generated models using the CMU-MOSI1 data set consisted of an input layer with 42 nodes, three hidden layers and an output layer with three nodes, as depicted in Figure 5.7. All the hidden layers implemented the *ReLU* activation function and were followed by a drop-out layer as summarised in Table 5.16. The output layer used the *softmax* activation function. It was trained for 4.287 minutes; stopping after 51 epochs and used a batch size of 512.

This model was evaluated by determining how well it could classify the data points in the CMU-MOSI1 test data set. A validation accuracy and loss value of 49.647% and 1.371, respectively, were obtained during the training process. Furthermore, a test accuracy of 52.401% and a loss value of 1.407 were calculated. The results obtained from the testing phase of the model are presented in the confusion matrix in Table 5.17, and the performance measures calculated from it are given in Table 5.18.

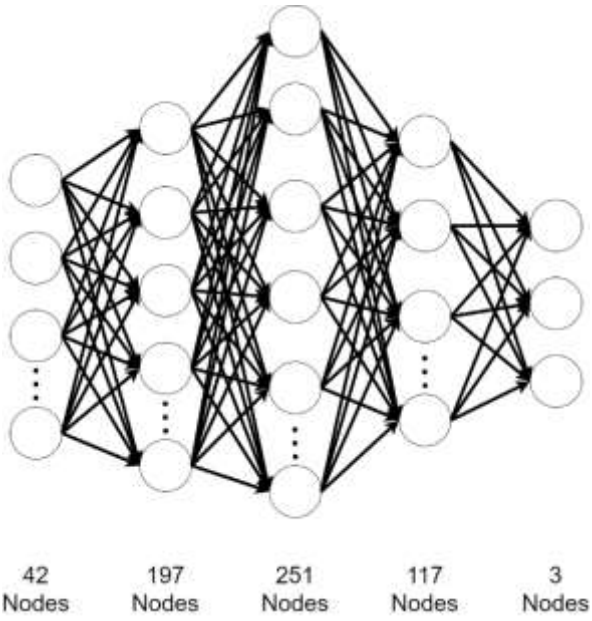


Figure 5.7: MLP architecture for the CMU-MOSI1 model

Table 5.16: Summary of the drop-out rates for each layer of the CMU-MOSI1 model

Hidden layer number	Drop-out rate
1	0.340
2	0.460
3	0.408

Table 5.17: Confusion matrix for the CMU-MOSI1 model

		Predicted sentiment		
		Positive	**Neutral**	**Negative**
Actual sentiment	**Positive**	8707	102	5750
	Neutral	443	0	267
	Negative	6791	31	6027

Table 5.18: Performance measures for the CMU-MOSI1 model

	Class-specific measures			Macro-averaging (M)	Micro-averaging (μ)
	Positive	**Neutral**	**Negative**		
Accuracy (%)	53.460	97.002	54.339	68.267	N/A
Error rate (%)	46.540	2.998	45.661	31.733	N/A
Precision (%)	54.620	0.000	50.042	34.887	52.401
Recall (%)	59.805	0.000	46.906	35.570	52.401
F-score (%)	57.095	0.000	48.423	35.226	52.401

5.5.2.2. CMU-MOSI2 model

The NAS algorithm generated a total of 626 models in a time of 6360.256 minutes. The CMU-MOSI2 model that performed the best contained five layers which each implemented the *ReLU* activation function. It also had an input layer with 42 input nodes and an output layer utilising the *softmax* activation function with three output nodes, as shown in Figure 5.8. A summary of the hidden nodes in each layer as well as the relevant drop-out rates is contained in Table 5.19. The model was trained for using a batch size of 4096 and stopped after 51 epochs or 4.405 minutes.

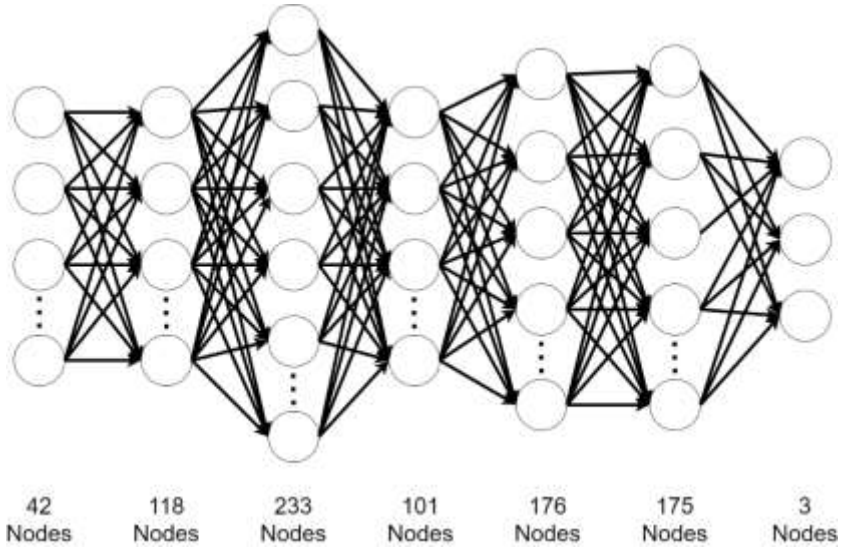


Figure 5.8: MLP architecture for the CMU-MOSI2 model

This model obtained a validation accuracy of 31.268% and a loss value of 1.227, as well as a test accuracy of 39.007 and loss value of 1.155, as calculated. The results obtained from evaluating the CMU-MOSI2 model, using the test data set, are presented in the form of a

confusion matrix in Table 5.20, and the calculated performance measures are summarised in Table 5.21.

Table 5.19: Summary of the drop-out rates for each layer of the CMU-MOSI2 model

Hidden layer number	Drop-out rate
1	0.397
2	0.278
3	0.269
4	0.332
5	0.327

Table 5.20: Confusion matrix for the CMU-MOSI2 model

		Predicted sentiment		
		Positive	**Neutral**	**Negative**
Actual sentiment	**Positive**	1598	7279	270
	Neutral	1073	8969	347
	Negative	1016	7165	401

Table 5.21: Performance measures for the CMU-MOSI2 model

	Class-specific measures			Macro-averaging (M)	Micro-averaging (μ)
	Positive	**Neutral**	**Negative**		
Accuracy (%)	65.723	43.581	68.710	59.338	N/A
Error rate (%)	34.277	56.419	31.290	40.662	N/A
Precision (%)	43.341	38.308	39.391	40.347	39.007
Recall (%)	17.470	86.332	4.673	36.158	39.007
F-score (%)	24.903	53.068	8.354	38.138	39.007

5.5.3. Discussion

These final two experiments that made use of the CMU-MOSI data set were performed to address the issues experienced with the collected data sets. As the CMU-MOSI data set contains labels corresponding to more than three classes, it had to relabelled to be used for this study. The labels of the CMU-MOSI1 data set were assigned by annotating all values greater than zero as positive, and all values less than zero as negative. All data points having a label of zero were then assigned as neutral. However, this led to an imbalance in the distribution of data points among the classes. Hence, the CMU-MOSI2 data set was created by relabelling all annotations with values greater than +1 as positive, values less than -1 as negative, and the remaining values were labelled as neutral.

5.5.3.1. CMU-MOSI1 model

Out of the five models presented in this study, the CMU-MOSI1 model is the shallowest, containing merely three hidden layers. When looking at the accuracy of this model, it is seen that it scored 68.267% with the neutral class, having a close to perfect accuracy of 97.002%. However, upon further inspection, it is seen that not a single sample was correctly classified as neutral, i.e. it had zero true positives. The high accuracy is, therefore, caused by a high number of true negatives. In other words, while the model showed difficulty in classifying data points as neutral correctly, it could easily identify data points as not belonging to this class. By merely relying on the accuracy without more in-depth inspection of the individual classes, this discrepancy would have been missed. Nevertheless, reviewing the values for the macro-average precision, recall and F-score, it was clear that this model would not be the ideal solution for performing sentiment analysis. The reason for this extremely low value for the neutral class can be contributed to the fact that only 3.662% of the data points in the data set were annotated as neutral. Better values were obtained for the other two classes, showing that the model performed much better in terms of classifying data points into these two classes.

Therefore, the original CMU-MOSI data set was relabelled to improve the class imbalance that occurred in this data set.

5.5.3.2. CMU-MOSI2 model

Though at first glance it may seem that the performance of this model is lower than that of the previous model, it did perform better than the CMU-MOSI1 model. This result is evident from comparing the values of the macro-average F-score for each of the models, as well as the performance measures for each of the sentiment classes. This model was able to classify data points belonging to the positive and negative sentiment classes with improved accuracy and precision. Even with a drastic fall in the accuracy of the neutral class, the model had a much better precision rate, indicating that a higher proportion of positive identifications was correct. At

86.332%, the recall of this class had the most significant improvement from the previous model. However, the same cannot be said for the recall of the other two classes that deteriorated to under 20%. This weak result indicated that the model could only identify a small proportion of the data points that belong to these classes.

5.5.3.3. Conclusions of experiments using the CMU-MOSI data set

After analysing the results obtained from the Advert22 model, it was decided to make use of an already existing data set to address the flaws in the previous data sets. In addition to splitting the data set at video level, it improved upon the previous data sets in the following ways:

1. Increase the number of subjects appearing in the videos;
2. It contained more videos;
3. The diversity of the people in the videos is greatly improved;
4. Human bias is minimised by making use of the already annotated data set; and
5. It is popularly used in the field of multimodal sentiment analysis.

Though the data set contained all these improvements, the models generated using this data set did perform worse than the first three models. The first reason for the poor performance can be attributed to the class imbalance that was present after relabelling the CMU-MOSI data set to create the CMU-MOSI1 data set. Because it only contained a minimal number of data points labelled as neutral, it led to the model being unable to classify data points labelled as neutral correctly. This imbalance was corrected in the CMU-MOSI2 model, which improved the overall results of the individual classes.

Another reason for the lower values of the performance measures for these two models is related to what the subjects in the data set are doing. In the CMU-MOSI data set, the subjects are speaking to an online audience. This difference may introduce additional noise into the data set because the movement of the mouth leads to more facial movements which influence the results obtained from Affectiva©.

The overall results obtained from the study for all the models are further discussed in the following section.

5.6. Results and conclusions

In Table 5.22, a summary of the results obtained from the five experiments are presented and will be briefly discussed in the subsections to follow.

Table 5.22: Summary of performance measures for each experiment

	Metric42	Emotion6	Advert22	CMU-MOSI1	CMU-MOSI2
Accuracy (%)	99.501	77.844	82.652	68.267	66.541
Error rate (%)	0.499	22.156	17.348	31.733	33.459
Micro-precision (%)	99.251	66.766	73.978	52.401	49.812
Macro-precision (%)	99.240	71.610	51.304	34.887	46.016
Micro-recall (%)	99.251	66.766	73.978	52.401	49.812
Macro-recall (%)	99.231	62.446	43.737	35.226	38.978
Micro-F-measure (%)	99.251	66.766	73.978	52.401	49.812
Macro-F-measure (%)	99.231	66.715	43.737	35.226	38.978

5.6.1. Overall evaluation of the models

The accuracies of the models found in the literature on multimodal sentiment analysis presented in Chapter 2 are summarised in Table 5.23, sorted according to the accuracies of the models. From the literature, it was clear that sentiment analysis based on the visual modality is challenging to perform as compared to the textual and visual modalities. This claim is further supported by the macro-averaged performance measures of the models generated in this study, as listed in Table 5.22. It can also be noted that the models trained using the collected data sets performed better than the models trained on the CMU-MOSI data set. This observation may be attributed to the tasks being performed by the subjects in the videos. Specifically, in the collected data sets, the participants were simply viewing content displayed on a computer monitor, whereas the subjects in the CMU-MOSI data set were actively, i.e. verbally, reviewing topics. The additional facial movements caused by the subjects speaking in the videos could lead to noise in the data. Thus, it can be deduced that sentiment can be detected much easier when subjects do not speak.

Comparing the five models generated as part of this study to those listed in Table 5.23 it becomes clear that the proposed technique of using an MLP trained on the data extracted using Affectiva© measures up to the models found in the literature. Moreover, the accuracies obtained with the CMU-MOSI1 and CMU-MOSI2 models, i.e. 68.267% and 59.338%, performed better than both the models presented by Poria *et al.* (2017b) and Zadeh *et al*. (2016a).

Table 5.23: Summary of accuracies of models found in the literature

Article	**Model**	**Data set**	**Accuracy (%)**
Junior and dos Santos (2018)	Support Vector Machine with Vector of Locally Aggregated Descriptors	Multimodal Opinion Utterances Data set	93.000
Junior and dos Santos (2018)	Support Vector Machine with Fisher Vector	CMU-MOSI	92.000
Junior and dos Santos (2018)	Support Vector Machine with Vector of Locally Aggregated Descriptors	CMU-MOSI	77.000
Junior and dos Santos (2018)	Support Vector Machine with Fisher Vector	YouTube data set	77.000
Junior and dos Santos (2018)	Support Vector Machine with Fisher Vector	Multimodal Opinion Utterances Data set	76.000
Poria *et al*. (2017c)	Support Vector Machine	YouTube data set	75.220
Pérez-Rosas *et al*. (2013)	Support Vector Machine	Spanish product opinions from YouTube	67.310
Wöllmer *et al*. (2013)	Support Vector Machine	Institute for Creative Technology's Multi-Modal Movie Opinion	61.200
Junior and dos Santos (2018)	Support Vector Machine with Vector of Locally Aggregated Descriptors	YouTube data set	57.000
Poria *et al*. (2017b)	CMU-MOSI (long short-term memory neural networks)	CMU-MOSI	55.800
Poria *et al*. (2017b)	Multimodal Opinion Utterances Data set (Long short-term memory neural networks)	Multimodal Opinion Utterances Data set	48.580
Morency *et al*. (2011)	Hidden Markov Model	YouTube data set	44.900
Ellis *et al*. (2014)	Support Vector Machine with Block-Based-Bag-of-Words	American news programs speakers	44.410
Zadeh *et al*. (2016a)	Nu-support vector regression	CMU-MOSI	36.000
Ellis *et al*. (2014)	Support Vector Machine with Local Binary Patterns	American news programs speakers	31.470

A few similarities came to light when comparing the five models to each other. Firstly, comparing the F-scores of each of the sentiment classes, it was clear that the models were, in general, more capable of classifying data points belonging to the positive sentiment class than into the other two classes. Because of the poor performance obtained from the neutral sentiment class, it can further be inferred that this sentiment is the most difficult to detect on human faces.

A second similarity between the models was in terms of their complexities. Apart from the CMU-MOSI1 model, a deeper architecture, i.e. more layers, seemed favourable to obtain good results. The CMU-MOSI1 model contained a mere three hidden layers, whereas the number of hidden layers for the other models ranged between five and ten. It subsequently seems that neural networks containing a larger number of hidden layers may be preferable for the task of sentiment analysis, based on the visual modality.

Even though the method of using an MLP to perform sentiment analysis, based on the metrics extracted from videos using Affectiva shows potential, other techniques may be better suited. The low values for precision, recall and the F-score can be linked back to the unstable and situational nature of emotions (Tian *et al.*, 2018), as well as the facial movements that are used to express these emotions. When a person shows emotion, it may appear as a sudden change in facial expressions, but in fact consists of several random facial movements over a short period (Barrett *et al.,* 2019). In other words, each frame that was extracted only contained small facial actions that were leading to the expression of a specific emotion or reaction. In the CMU-MOSI data set, the subjects are speaking, which adds additional facial actions, especially around the mouth area. For this reason, it is suggested that other types of neural networks that work with time-series data, such as a long-short term memory neural network (LSTM), could potentially improve the results from this study. This is because these neural networks contain memory blocks or vectors that store a temporal state of the neural network (Karim *et al.*, 2017; Sundermeyer *et al.*, 2012). A memory block is then referenced each time the corresponding node is activated, allowing them to take previous node input into account before giving an output. Though future research into using these techniques is encouraged, it is not within the scope of this study, due to time limitations that were in place.

5.6.2. Performance of the neural architecture search algorithm

In this study, all the models were generated using an evolutionary NAS algorithm, specifically the regularised evolution search strategy, as discussed in Chapter 4. This algorithm was implemented using Python, and the code for it is included in Annexure C. It allows the user to specify the maximum and minimum values for the number of hidden layers, number of nodes per hidden layer and drop-out rate. The sample size, initial population size and the total number of models to be generated can be specified.

This algorithm can generate a considerable number of models in a relatively short time, depending on the size of the data set, complexity of the models being generated and how fast the loss value of models stagnates. In this study, it took roughly 17 days to generate 2000 models for the Metric42 data set. Another factor that plays a crucial role in how fast the models

are generated is the amount of RAM and whether a GPU is available. Similarly, storage space can limit the number of models that are generated, as is evident in this study. Due to only having 15GB of storage available on Google Drive, merely 626 models were generated for the CMU-MOSI1 and CMU-MOSI2 data sets.

The algorithm can also generate a wide array of possible architectures as a starting point and quickly narrow the search space down when the mutation phase is executed. This is because the algorithm takes the best model from a small sample and only makes slight adjustments to some of the hyperparameters, thereby ensuring that the algorithm moves to a smaller, more optimal, area of the search space, resulting in models with high accuracies.

5.7. Summary

Five experiments were performed to answer the research question of “How can affective computing be used with deep learning techniques to perform sentiment analysis on videos to mitigate feedback problems, such as response bias?”. The first experiment formed part of a pilot study with a small number of participants, which obtained an unusually high accuracy during the testing phase of the model. However, this could be ascribed to too little variation within the data set due to the limited number of participants. The training, validation and test data sets were constructed by dividing the data at frame level, which may have led to overfitting during the training phase. The second experiment, also part of the pilot study, made use of the same video data set. However, it was processed to only make use of the six emotions identified by Ekman (1999) as input to the classification model. A third experiment was executed with a more significant number of participants to increase the levels of variations within the data set. The data set was also divided on video level to create the three subsets used to train and evaluate the model. Nevertheless, with this experiment, a bias may have been introduced by labelling the data points according to the responses given by the participants. Therefore, a third data set was used to address the limitations of both the previous studies. For the last two experiments, a third-party data set was used which already contained labels and contained a more significant number of subjects in the videos. The above-mentioned experiments, along with an analysis of the results obtained from them were presented in this chapter.

In the next chapter, an overview of the study will be provided, along with the conclusions and contributions made.

Chapter 6 Conclusions

There is no real ending. It's just the place where you stop the story.

~Frank Herbert

6.1. Introduction

The aim of this study was to determine whether sentiment analysis can effectively be performed using deep learning and affective computing for facial expression recognition. Therefore, several deep multilayer perceptron (MLP) neural networks were modelled and trained using affective data extracted from three distinct data sets. Results obtained from training five deep MLPs suggests that the technique can indeed produce usable results.

The objective of this chapter is to offer a summary and conclusions of the study and future directions. In Section 6.2, a discussion about how the research goals stipulated in Chapter 1 were achieved is given. Then, the contributions made by this study are highlighted in Section 6.3. Additionally, the problems that arose during the execution of the study are presented in Section 6.4, followed by a discussion in Section 6.5 on the possible future work that may be explored. This chapter is concluded with a summary in Section 6.6.

6.2. Evaluation of research goals

The research question guiding this study, as stated in Chapter 1, Section 1.2, is "How can affective computing be used with deep learning techniques to perform sentiment analysis on videos to mitigate feedback problems, such as response bias?". To find an answer to this question, four secondary objectives were set. A brief overview of how each secondary objective was met is provided below.

1. **Construct or find a data set consisting of video recordings of people's faces and annotate with the expressed sentiment**

Five experiments, using three distinct data sets, were performed during the study. Each subsequent data set was selected to address the issues that occurred with its predecessor. The first data set consisted of 27 recorded videos of nine participants reading three text passages, as described in Chapter 5, Section 5.3. Annotations were assigned to the data points, based on the intended evoked sentiment of each of the text passages, i.e. positive, neutral or negative. After pre-processing the videos, the data set consisted of 132261 data points which were divided into training, validation and testing data sets in a 70:20:10 ratio.

The second data set was created, using 22 participants watching three video advertisements, and presented in Chapter 5, Section 5.4. It resulted in a data set consisting of 66 videos with a total of 159905 data points. Video level division was used to create the three subsequent data sets needed for training and evaluating the deep MLP neural network. This was done to prevent overfitting during the training phase.

A final data set, i.e. the Carnegie Mellon University Multimodal Opinion Sentiment Intensity (CMU-MOSI) data set, was acquired for the following reasons:

1. The number of unique people appearing in the videos is increased to 89 instead of only 22 participants in the previous data set;
2. A large number of videos with different labels, i.e. 2199 segmented videos, is included in the data set;
3. Diversity in the people that appear within the videos is more significant than in the previously collected data sets;
4. Labels were assigned to the videos by five independent workers, thereby minimising the chance of introducing a bias to the model by either the author or the participants; and
5. Literature indicates that it has been widely used for creating models that perform multimodal sentiment analysis.

This data set contains 93 videos of 89 subjects voicing their opinions in different settings on a variety of topics. All videos are segmented at the utterance level, resulting in 2199 videos. Each utterance was annotated by five independent workers on a scale from strongly negative (-3) to strongly positive (+3). The annotations were adapted to only include three sentiment classes, i.e. positive, neutral and negative.

2. Extract useful features relating to emotions from the video data set, using affective computing techniques

To achieve this goal, Affectiva©'s emotion SDK was used to extract 42 metrics based on the facial expressions displayed in the videos. The metrics are shown in Chapter 5, Table 5.1 and include seven emotions, 21 facial expressions, 12 emojis, engagement and valence.

3. Develop a deep MLP for classifying affective data into one of three sentiment categories, i.e. positive, neutral, or negative

An evolutionary neural architecture search algorithm, as discussed in Chapter 4, Section 4.5, was implemented to find an appropriate deep MLP model for each of the three previously described data sets. This algorithm can be briefly summarised as follows:

1. Determine the search space boundaries which provide a definition for all the possible architectures that may be explored. The boundaries of the search space for this study are summarised in Table 6.1;

Table 6.1: Summary of search space boundaries

Hyperparameter	**Range of allowable values**
Number of hidden layers	3 to 10 layers
Number of nodes per hidden layer	10 to 256 nodes
Hidden layer activation function	*Rectified linear unit* (*ReLU*)
Output layer activation function	*softmax*
Drop-out rate	0.2 to 0.5
Batch size	Powers of two between 32 to 4096

2. Implement a search strategy to select an architecture from the search space. The regularised evolution search strategy presented in Section 4.5 was used;
3. Estimate the performance of the selected architecture and return the performance measure to the search strategy. Each generated model was evaluated, using the accuracy obtained from the validation data set;
4. Repeat step 2 and step 3 until C cycles have been completed; and
5. Select the architecture with the best performance measure.

The regularised evolution search strategy was executed for 200 cycles, after which the model with the best accuracy was selected. In Chapter 5, Section 5.3.2, Section 5.4.2 and Section 5.5.2, the resulting architectures for each respective experiment are discussed.

4. Determine how well the deep MLP performs by evaluating it in terms of accuracy, error rate, precision, recall and F-score

Six performance measures were calculated for each of the models that was represented in Section 5.3 to Section 5.5. These measures include the following:

- Accuracy: Calculates the ratio of correctly classified data points to the total number of data points to determine how well the model can classify both positive and negative data points correctly;
- Error rate: Indicates how often errors were made while the model classified the given test data;
- Precision: Represents the proportion of correctly classified true positives in relation to the total number of predicted positives;

- Recall: A measurement that compares the number of data points labelled as positive compared to those that are actually positive. It is used to describe the model's completeness; and
- F-score: A weighted average of precision and recall that indicates how accurately the model performed classification on the test data set. It is also known as the harmonic mean of precision and recall.

In summary, all the goals set in Chapter 1 were met and the following conclusions can be drawn:

- Combining affective computing with a deep MLP to perform sentiment analysis based on visual input, delivers accuracies comparable to other models found in the literature;
- Better results can be obtained using a data set containing data related to both emotions and facial expression, instead of using merely identified emotions; and
- Due to the inherent nature of the data, i.e. consisting of emotions and facial expressions, other types of deep neural networks, such as a long-short term memory (LSTM) neural networks, may be more suitable for this task.

In the next section, the contributions that this study has made to the existing literature is offered.

6.3. Contributions

This study contributed to the current body of knowledge in the following ways:

- Relevant information regarding the fields of multimodal sentiment analysis, and affective computing was summarised through a literature study into a central source for future reference;
- An approach towards sentiment analysis that only exploits the visual modality through means of a deep multilayer perceptron neural network for classification was presented. This approach made use of videos containing the faces of human subjects and obtained results that were similar to other models presented in literature;
- An application to record the facial expressions of a participant and extract the affective data was developed;
- It was also shown how affective computing techniques to extract features could be used to perform sentiment analysis;
- The results from the first two experiments suggest that it is less efficient to only make use of the six emotions identified by Ekman (1999); however, further investigation may be needed to confirm this;

- This approach was proposed to be applied to mitigate general feedback problems by detecting a respondent's true opinion by analysing his/her facial expressions; and
- It was revealed that dividing a data set consisting of multiple videos at frame level might lead to overfitting by the model and should instead be done at video level.

The proposed approach shows potential for being applied to the problem of sentiment analysis; however, some adjustments, as proposed in Section 6.5, need to be made to further improve the results. Thus, the work done in this study may be practically used by organisations to gain insights into their customers' true opinions and avoid general feedback problems, such as response and social desirability bias, by analysing their facial expressions in online surveys. The limitations of the study are presented within the next section.

6.4. Limitations

Since direct access to a computer with a GPU was not possible, it was decided to run the neural architecture search algorithm on Google's Colaboratory cloud service. Though it provides free access to a GPU and other tools to run data analytics and other machine learning activities, it limits users' sessions to 12 hours only. This limitation led to having to start a new session once the previous session ended, which could not always be done immediately and caused several hours to be wasted. The code for the algorithm had to be adapted to store the *population* queue and *history* list used by the algorithm on Google Drive, in order for it to continue seamlessly when the new session started. This workflow also imposed another restriction in the form of a 15GB storage capacity for the experiment.

Computer Science and Information Technology students were identified as the targeted participants because they were the most accessible group. Nevertheless, making use of only these students caused a lack of diversity and variation within the data set. Furthermore, only a limited number of students volunteered to participate, and an even smaller number of this group showed up on the day of the recordings. This shortage of participation was another reason why a more extensive data set was identified to perform the final two experiments.

6.5. Future work

This study made use of a deep MLP neural network as a means of classifying the collected data points as a positive, neutral, or negative sentiment. Other types of deep learning model, such as recurrent, convolutional or LSTM neural networks can be examined to possibly improve the efficiency of the sentiment analysis task. Ensembles of neural networks may also be explored for this purpose. The focus of the MLP was to classify each frame into the three sentiment categories. However, it may be useful to consider the overall sentiment of the participant or to

identify which aspects in the video evoked a specific sentiment from the participant. Thus, in future research, deep learning models which perform at the video level, aspect level, and utterance level, where consecutive frames are grouped, can be developed.

The collected videos were processed using the Affectiva© emotion SDK to extract affective data. Other systems and techniques for extracting this data based on the participants' facial expressions, such as a 3D Constrained Local Model (Baltrušaitis *et al.*, 2012), can be inspected which may result in new features that can be provided to the deep learning model as input.

Video advertisements were shown to participants to evoke specific sentiments. Future research may consider making use of other content, such as an authority explaining a new policy or procedure, or an instructional video. Furthermore, other media types, for example, written passages, or podcasts, can also be presented to the participants. The different reactions based on the type of media can provide significant insights into what type of media would be best suited to convey different types of information to a person.

6.6. Summary

In this study, the research question of "How can affective computing be used with deep learning techniques to perform sentiment analysis on videos to mitigate feedback problems, such as response bias?" was investigated. It was shown that a deep MLP neural network trained using 42 affective metrics extracted from videos containing human faces to perform sentiment analysis could successfully classify the sentiment experienced by participants in videos. The results obtained from the five models created as part of this study were comparable to those of models found in literature, further highlighting the potential this technique holds. However, these results can further be improved by using other types of neural networks, such as long-short term memory neural networks.

Bibliography

Abdi, A., Shamsuddin, S. M., Hasan, S. and Piran, J. 2019. Deep learning-based sentiment classification of evaluative text based on multi-feature fusion. *Information Processing & Management,* 56(4):1245-1259.

Affectiva. 2017. Emotion ai 101: All about emotion detection and affectiva's emotion metrics. https://blog.affectiva.com/emotion-ai-101-all-about-emotion-detection-and-affectivas-emotion-metrics (Date of access: 7 September 2019).

Affectiva. s.a. How it works. https://www.affectiva.com/how/how-it-works/ Date of access: 2019/09/20.

Amazon. 2019. Amazon Alexa - "Not Everything Makes the Cut". Available at: https://www.youtube.com/watch?v=e7iSdU7cuCA (Date of access: 7 September 2019).

Araque, O., Corcuera-Platas, I., Sanchez-Rada, J. F. and Iglesias, C. A. 2017. Enhancing deep learning sentiment analysis with ensemble techniques in social applications. *Expert Systems with Applications,* 77:236-246.

Azam, F. 2000. Biologically inspired modular neural networks. Virginia Tech. (Thesis - PhD).

Baker, B., Gupta, O., Raskar, R. and Naik, N. 2017. Accelerating neural architecture search using performance prediction. (*In* Conference on Neural Information Processing Systems 2017 Workshop on Meta-Learning, Hyatt Regency Long Beach, California, United States).

Baltrušaitis, T., Zadeh, A., Lim, Y. C. and Morency, L.-P. 2018. Openface 2.0: Facial behavior analysis toolkit. (*In* 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018), Xi'an, China: IEEE, p. 59-66).

Baltrušaitis, T., Robinson, P. and Morency, L.-P. 2016. Openface: An open source facial behavior analysis toolkit. (*In* 2016 IEEE Winter Conference on Applications of Computer Vision (WACV), Lake Placid, NY, USA: IEEE, p. 1-10).

Baltrušaitis, T., Robinson, P. and Morency, L.-P. 2013. Constrained local neural fields for robust facial landmark detection in the wild. (*In* Proceedings of the IEEE international conference on computer vision workshops, Sydney, Australia, p. 354-361).

Baltrušaitis, T., Robinson, P. and Morency, L.-P. 2012. 3D constrained local model for rigid and non-rigid facial tracking. (*In* 2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, Rhode Island: IEEE, p. 2610-2617).

Barrett, L. F., Adolphs, R., Marsella, S., Martinez, A. M. and Pollak, S. D. 2019. Emotional expressions reconsidered: Challenges to inferring emotion from human facial movements. *Psychological science in the public interest,* 20(1):1-68.

Basheer, I. A. and Hajmeer, M. 2000. Artificial neural networks: Fundamentals, computing, design, and application. *Journal of microbiological methods,* 43(1):3-31.

Beigi, G., Hu, X., Maciejewski, R. and Liu, H. 2016. An overview of sentiment analysis in social media and its applications in disaster relief. (*In* Pedrycz, W. and Chen, S. *eds*. Sentiment analysis and ontology engineering. Switzerland: Springer, p. 313-340).

Bender, G., Kindermans, P.-J., Zoph, B., Vasudevan, V. and Le, Q. 2018. Understanding and simplifying one-shot architecture search. (*In* International Conference on Machine Learning, Stockholm Sweden, p. 549-558).

Byeon, Y.-H. and Kwak, K.-C. 2014. Facial expression recognition using 3d convolutional neural network. *International journal of advanced computer science and applications,* 5(12).

Cai, H., Chen, T., Zhang, W., Yu, Y. and Wang, J. 2018a. Efficient architecture search by network transformation. (*In* Thirty-Second AAAI Conference on Artificial Intelligence).

Cai, H., Yang, J., Zhang, W., Han, S. and Yu, Y. 2018b. Path-level network transformation for efficient architecture search. (*In* Proceedings of the 35th International Conference on Machine Learning 2018, Stockholm Sweden, p. 677-686).

Cai, H., Zhu, L. and Han, S. 2018c. Proxylessnas: Direct neural architecture search on target task and hardware. (*In* 7th International Conference on Learning Representations, Ernest N. Morial Convention Center, New Orleans).

Calvo, R. A., D'Mello, S., Gratch, J. M. and Kappas, A. 2015. The Oxford handbook of affective computing. USA: Oxford University Press.

Cambria, E., Poria, S., Hussain, A. and Liu, B. 2019. Computational intelligence for affective computing and sentiment analysis. *IEEE Computational Intelligence Magazine,* 14(2):16-17.

Cambria, E., Das, D., Bandyopadhyay, S. and Feraco, A. 2017. Affective computing and sentiment analysis. (*In* Cambria, E., Das, D., Bandyopadhyay, S., Feraco, A. *eds*. A practical guide to sentiment analysis. Singapore: Springer, p. 1-10).

Catal, C. and Nangir, M. 2017. A sentiment classification model based on multiple classifiers. *Applied Soft Computing,* 50:135-141.

Chapelle, O., Schölkopf, B. and Zien, A. 2006. Semi-supervised learning. Cambridge, United States: MIT Press.

Chebli, A., Djebbar, A. and Marouani, H. F. 2018. Semi-supervised learning for medical application: A survey. (*In* 2018 International Conference on Applied Smart Systems (ICASS), Medea, Algeria: Institute of Electrical and Electronics Engineers (IEEE), p. 1-9).

Chen, D., Ren, S., Wei, Y., Cao, X. and Sun, J. 2014. Joint cascade face detection and alignment. (*In* European Conference on Computer Vision, Zurich, Switzerland: Springer, p. 109-122).

Chen, J., Chen, Z., Chi, Z. and Fu, H. 2018. Facial expression recognition in video with multiple feature fusion. *IEEE Transactions on Affective Computing,* 9(1):38-50.

Coolen, A. 1998. A beginner's guide to the mathematics of neural networks. (*In* Landau, L.J. & Taylor, J.G. *eds*. Concepts for neural networks. London: Springer: 13-70).

Dahl, G. E., Sainath, T. N. and Hinton, G. E. 2013. Improving deep neural networks for LVCSR using rectified linear units and drop-out. (*In* Proceedings of 2013 IEEE international conference on acoustics, speech and signal processing,Vancouver, Canada, p. 8609-8613).

D'Andrea, A., Ferri, F., Grifoni, P. and Guzzo, T. 2015. Approaches, tools and applications for sentiment analysis implementation. *International Journal of Computer Applications,* 125(3): 26-33.

D'Mello, S., Kappas, A. and Gratch, J. 2018. The affective computing approach to affect measurement. *Emotion Review,* 10(2):174-183.

Darwin, C. 1872. The expression of the emotions in man and animals. London, UK: John Marry.

Deng, L. and Yu, D. 2014. Deep learning: Methods and applications. *Foundations and Trends® in Signal Processing,* 7(3-4):197-387.

Devika, M., Sunitha, C. and Ganesh, A. 2016. Sentiment analysis: A comparative study on different approaches. *Procedia Computer Science,* 87:44-49.

Ding, B., Qian, H. and Zhou, J. 2018. Activation functions and their characteristics in deep neural networks. (*In* 2018 Chinese Control And Decision Conference (CCDC): IEEE, p. 1836-1841).

Domhan, T., Springenberg, J. T. and Hutter, F. 2015. Speeding up automatic hyperparameter optimization of deep neural networks by extrapolation of learning curves. (*In* Twenty-Fourth International Joint Conference on Artificial Intelligence, Palo Alto, California USA).

Du, K.-L. and Swamy, M. N. 2013. Neural networks and statistical learning. London: Springer Science & Business Media.

Ekkekakis, P. and Zenko, Z. 2016. Measurement of affective responses to exercise: From "affectless arousal" to "the most well-characterized" relationship between the body and affect. (*In* Meiselman H.L. *eds* Emotion measurement. Amsterdam: Elsevier: p. 299-321).

Ekman, P. and Friesen, W. V. 2002. Facial action coding system: Manual. Available at: https://www.paulekman.com/product/facs-manual/. (Date of access: 12 June 2019).

Ekman, P. 1999. Basic emotions. *Handbook of Cognition and Emotion,* 98(45-60):45-60.

Ekman, P. and Friesen, W. V. 1976. Measuring facial movement. *Environmental Psychology and Nonverbal Behavior,* 1(1):56-75.

Ekman, P. and Friesen, W. V. 1971. Constants across cultures in the face and emotion. *Journal of Personality and Social Psychology,* 17(2):124.

Ellis, J. G., Jou, B. and Chang, S.-F. 2014. Why we watch the news: A data set for exploring sentiment in broadcast video news. (*In* Proceedings of the 16th international conference on multimodal interaction, Istanbul, Turkey: ACM, p. 104-111).

Elsken, T., Metzen, J. H. and Hutter, F. 2019a. Efficient multi-objective neural architecture search via lamarckian evolution. (*In* Proceedings of 7th International Conference on Learning Representations, Ernest N. Morial Convention Center, New Orleans).

Elsken, T., Metzen, J. H. and Hutter, F. 2019b. Neural architecture search: A survey. *Journal of Machine Learning Research,* 20(55):1-21.

Elsken, T., Metzen, J. H. and Hutter, F. 2018. Simple and efficient architecture search for convolutional neural networks. (*In* Proceedings of 6th International Conference on Learning Representations, Vancouver Convention Center, Vancouver Canada).

Eluyode, O. and Akomolafe, D. T. 2013. Comparative study of biological and artificial neural networks. *European Journal of Applied Engineering and Scientific Research,* 2(1):36-46.

Facebook. 2014. Chairs Are Like Facebook - Original Ad. Available at: https://www.youtube.com/watch?v=SSzoDPptYNA (Date of access: 7 July 2019).

Falkner, S., Klein, A. and Hutter, F. 2018. Bohb: Robust and efficient hyperparameter optimization at scale. (*In* Proceedings of 35th International Conference on Machine Learning, Stockholmsmässan, Stockholm Sweden, p. 1436-1445).

Farhadloo, M. and Rolland, E. 2016. Fundamentals of sentiment analysis and its applications. (*In* Pedrycz, W. and Chen, S. *eds*. Sentiment analysis and ontology engineering. Switzerland: Springer: 1-24).

Feng, J., Cai, S. and Ma, X. 2018. Enhanced sentiment labeling and implicit aspect identification by integration of deep convolution neural network and sequential algorithm. *Cluster Computing*:1-19.

Fisher, R. A. 1936. The use of multiple measurements in taxonomic problems. *Annals of Eugenics,* 7(2):179-188.

Flach, P. 2019. Performance evaluation in machine learning: The good, the bad, the ugly and the way forward. (*In* Proceedings of 33rd AAAI Conference on Artificial Intelligence, Hilton Hawaiian Village, Honolulu, Hawaii, USA).

Friedman, J., Hastie, T. and Tibshirani, R. 2000. Additive logistic regression: A statistical view of boosting (with discussion and a rejoinder by the authors). *The Annals of Statistics,* 28(2):337-407.

Friesen, W. V. and Ekman, P. 1983. EMFACS-7: Emotional facial action coding system. *Unpublished manuscript, University of California at San Francisco,* 2(36).

Ghiassi, M. and Lee, S. 2018. A domain transferable lexicon set for twitter sentiment analysis using a supervised machine learning approach. *Expert Systems with Applications,* 106:197-216.

Girard, J. M., Cohn, J. F., Jeni, L. A., Lucey, S. and De la Torre, F. 2015. How much training data for facial action unit detection? (*In* Proceedings of 11th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG), Ljubljana, Slovenia: IEEE, p. 1-8).

Gittelman, S., Lange, V., Cook, W. A., Frede, S. M., Lavrakas, P. J., Pierce, C. and Thomas, R. K. 2015. Accounting for social-desirability bias in survey sampling: A model for predicting and calibrating the direction and magnitude of social-desirability bias. *Journal of Advertising Research,* 55(3):242-254.

Gokgoz, E. and Subasi, A. 2015. Comparison of decision tree algorithms for EMG signal classification using DWT. *Biomedical Signal Processing and Control,* 18:138-144.

Goodfellow, I., Bengio, Y. and Courville, A. 2016. Deep learning. Cambridge Massachusetts: MIT press.

Graves, A. and Schmidhuber, J. 2005. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks,* 18(5-6):602-610.

Gray, D. E. 2014. Doing research in the real world. Los Angeles: Sage Publications.

Guyon, I., Bennett, K., Cawley, G., Escalante, H. J., Escalera, S., Ho, T. K., Macia, N., Ray, B., Saeed, M. and Statnikov, A. 2015. Design of the 2015 ChaLearn AutoML challenge. (*In* Proceedings of 2015 International Joint Conference on Neural Networks (IJCNN): IEEE, p. 1-8).

Hagan, M. T., Demuth, H. B., Beale, M. H. and De Jesús, O. 2014. Neural network design. 2nd ed. Texas, USA: Martin Hagan.

Han, J., Zhang, Z. and Schuller, B. 2019. Adversarial training in affective computing and sentiment analysis: Recent advances and perspectives. *IEEE Computational Intelligence Magazine,* 14(2):68-81.

Hanin, B. 2018. Which neural net architectures give rise to exploding and vanishing gradients? (*In* Proceedings of Advances in Neural Information Processing Systems 2018, Montréal, Canada, p. 582-591).

Haykin, S. 2009. Neural networks and learning machines. Upper Saddle River: Pearson.

Hesch, J. A. and Roumeliotis, S. I. 2011. A direct least-squares (DLS) method for PnP. (*In* Proceedings of 2011 International Conference on Computer Vision: IEEE, p. 383-390).

Jagielski, M., Oprea, A., Biggio, B., Liu, C., Nita-Rotaru, C. and Li, B. 2018. Manipulating machine learning: Poisoning attacks and countermeasures for regression learning. (*In* Proceedings of 2018 IEEE Symposium on Security and Privacy (SP): IEEE, p. 19-35).

James, W. 1884. What is an emotion? *Mind,* 9(34):188-205.

Jiang, B., Valstar, M., Martinez, B. and Pantic, M. 2014. A dynamic appearance descriptor approach to facial actions temporal modeling. *IEEE Transactions on Cybernetics,* 44(2):161-174.

Jianqiang, Z., Xiaolin, G. and Xuejun, Z. 2018. Deep convolution neural networks for twitter sentiment analysis. *IEEE Access,* 6:23253-23260.

Jiao, Y. and Du, P. 2016. Performance measures in evaluating machine learning based bioinformatics predictors for classifications. *Quantitative Biology,* 4(4):320-330.

Jing, L. and Tian, Y. 2019. Self-supervised visual feature learning with deep neural networks: A survey. *arXiv preprint arXiv:1902.06162*.

Jordan, M. I. and Mitchell, T. M. 2015. Machine learning: Trends, perspectives, and prospects. *Science,* 349(6245):255-260.

Joy, N. C. and Prasad, J. 2016. Feature extraction techniques for facial expression recognition systems. *GRD Journals-Global Research and Development Journal for Engineering,* 1(2):22-25.

Junior, A. G. and Dos Santos, E. M. 2018. A method for opinion classification in video combining facial expressions and gestures. (*In* 2018 31st SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI), Foz do Iguaçu, Paraná, Brazil: IEEE, p. 33-40).

Kahou, S. E., Bouthillier, X., Lamblin, P., Gulcehre, C., Michalski, V., Konda, K., Jean, S., Froumenty, P., Dauphin, Y., Boulanger-Lewandowski, N., Chandias Ferrari, R., Mirza, M., Warde-Farley, D., Courville, A., Vincent, P., Memisevic, R., Pal, C. and Bengio, Y. 2016. EmoNets: Multimodal deep learning approaches for emotion recognition in video. *Journal on Multimodal User Interfaces,* 10(2):99-111.

Kahou, S. E., Michalski, V., Konda, K., Memisevic, R. and Pal, C. 2015. Recurrent neural networks for emotion recognition in video. (*In* Proceedings of the 2015 ACM on International Conference on Multimodal Interaction, Seattle, United States: ACM, p. 467-474).

Kahou, S. E., Pal, C., Bouthillier, X., Froumenty, P., Gülçehre, Ç., Memisevic, R., Vincent, P., Courville, A., Bengio, Y. and Ferrari, R. C. 2013. Combining modality specific deep neural networks for emotion recognition in video. (*In* Proceedings of the 15th ACM on International conference on multimodal interaction: ACM, p. 543-550).

Karim, F., Majumdar, S., Darabi, H. and Chen, S. 2017. LSTM fully convolutional networks for time series classification. *IEEE Access*, 6:1662-1669.

Khalafi, F. F. H. and Mirvakili, S. 2011. A literature survey of neutronics and thermal-hydraulics codes for investigating reactor core parameters; artificial neural networks as the VVER-1000 core predictor. (*In* Tsvetkov, P. *eds*. Nuclear power-system simulations and operation: IntechOpen. p.103-122).

Kirchner, J., Heberle, A. and Löwe, W. 2015. Classification vs. Regression-machine learning approaches for service recommendation based on measured consumer experiences. (*In* Proceedings of 2015 IEEE World Congress on Services, New York City, NY, USA: IEEE, p. 278-285).

Klein, A., Falkner, S., Springenberg, J. T. and Hutter, F. 2016. Learning curve prediction with bayesian neural networks. (*In* Proceedings of 5th International Conference on Learning Representations, Palais des Congrès Neptune, Toulon, France).

Krüger, F. 2016. Activity, context, and plan recognition with computational causal behaviour models. Universität Rostock. (Dissertation - PhD).

Kumar, A. and Jaiswal, A. 2017. Empirical study of twitter and Tumblr for sentiment analysis using soft computing techniques. (*In* Proceedings of the world congress on engineering and computer science, San Francisco, USA, p. 1-5).

Landsberger, H. A. 1958. Hawthorne revisited: Management and the worker, its critics, and developments in human relations in industry. Ithaca, N.Y. : Cornell University.

Larson, R. B. 2019. Controlling social desirability bias. *International Journal of Market Research,* 61(5):1-14.

LeCun, Y., Bengio, Y. and Hinton, G. 2015. Deep learning. *Nature,* 521(7553): 436-444.

Lee, W. and Norman, M. D. 2016. Affective computing as complex systems science. *Procedia Computer Science,* 95:18-23.

Li, L., Jamieson, K., DeSalvo, G., Rostamizadeh, A. and Talwalkar, A. 2017. Hyperband: A novel bandit-based approach to hyperparameter optimization. *The Journal of Machine Learning Research,* 18(1):6765-6816.

Li, S., Walters, G., Packer, J. and Scott, N. 2018. Using skin conductance and facial electromyography to measure emotional responses to tourism advertising. *Current Issues in Tourism*, 21(15):1761-1783.

Littlewort, G., Whitehill, J., Wu, T., Fasel, I., Frank, M., Movellan, J. and Bartlett, M. 2011. The computer expression recognition toolbox (cert). (*In* Proceedings of Face and gesture 2011: IEEE, p. 298-305).

Liu, B. 2017. Many facets of sentiment analysis. (*In* Cambria, E., Das, D., Bandyopadhyay, S., Feraco, A. *eds*. A practical guide to sentiment analysis. Singapore: Springer: 11-39).

Liu, B. and Zhang, L. 2012. A survey of opinion mining and sentiment analysis. (*In* Aggarwal, C.C. and Zhai, C.X. *eds*. Mining text data. Boston: Springer: 415-463).

Liu, H., Simonyan, K. and Yang, Y. 2019. DARTS: Differentiable architecture search. (*In* Proceedings of 7th International Conference on Learning Representations, Ernest N. Morial Convention Center, New Orleans).

Liu, W., Wen, Y., Yu, Z. and Yang, M. 2016. Large-margin softmax loss for convolutional neural networks. (*In* Proceedings of 33rd International Conference on Machine Learning (ICML 2016), New York City, New York, USA, p. 7-16).

Mäntylä, M. V., Graziotin, D. and Kuutila, M. 2018. The evolution of sentiment analysis—a review of research topics, venues, and top cited papers. *Computer Science Review,* 27:16-32.

Maree, N. J., Drevin, L., Du Toit, J. V. and Kruger, H. A. 2019a. Affective computing and deep learning to perform sentiment analysis, (*In* Southern Africa Telecommunication Networks and Applications Conference (SATNAC) 2019, Fairmont Zimbali Resort, Ballito, KwaZulu-Natal, South Africa, p. 94-99).

Maree, N. J., Drevin, L., Du Toit, J. V. and Kruger, H. A. 2019b. Performing visual sentiment analysis using a deep learning approach. (*In* 48th Operations Research Society of South Africa (ORSSA) Annual Conference, The Vineyard Hotel, Cape Town, South Africa).

Maree, N. J., Drevin, L., Du Toit, J. V. and Kruger, H. A. 2018. Affective computing and deep learning to perform sentiment analysis in order to address response bias. (*In* 47th Operations Research Society of South Africa (ORSSA) Annual Conference, CSIR International Convention Centre, Pretoria, South Africa).

Martinez, A. M. 2019. Context may reveal how you feel. *Proceedings of the National Academy of Sciences,* 116(15):7169-7171.

Matsumoto, D. 1992. More evidence for the universality of a contempt expression. *Motivation and Emotion,* 16(4):363-368.

McDuff, D., Amr, M. and El Kaliouby, R. 2019. AM-FED+: An extended data set of naturalistic facial expressions collected in everyday settings. *IEEE Transactions on Affective Computing,* 10(1):7-17.

McDuff, D., El Kaliouby, R., Cohn, J. F. and Picard, R. W. 2014a. Predicting ad liking and purchase intent: Large-scale analysis of facial responses to ads. *IEEE Transactions on Affective Computing,* 6(3):223-235.

McDuff, D., El Kaliouby, R., Senechal, T., Demirdjian, D. and Picard, R. 2014b. Automatic measurement of ad preferences from facial responses gathered over the internet. *Image and Vision Computing,* 32(10):630-640.

McDuff, D., Kaliouby, R., Senechal, T., Amr, M., Cohn, J. and Picard, R. 2013. Affectiva-MIT facial expression data set (AM-FED): Naturalistic and spontaneous facial expressions collected. (*In* Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, p. 881-888).

McDuff, D., Mahmoud, A., Mavadati, M., Amr, M., Turcot, J. and el Kaliouby, R. 2016. Affdex SDK: A cross-platform real-time multi-face expression recognition toolkit. (*In* Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems [Association for Computing Machinery (ACM), 2016], p. 3723-3726).

Mehrabian, A. and Russell, J. A. 1974. The basic emotional impact of environments. *Perceptual and Motor Skills,* 38(1):283-301.

Mennig, P., Scherr, S. A. and Elberzhager, F. 2019. Supporting rapid product changes through emotional tracking. (*In* Proceedings of the 4th International Workshop on Emotion Awareness in Software Engineering, Montreal, QC, Canada: IEEE Press, p. 8-12).

Micu, A., Micu, A. E., Geru, M. and Lixandroiu, R. C. 2017. Analyzing user sentiment in social media: Implications for online marketing strategy. *Psychology & Marketing,* 34(12):1094-1100.

Mogaji, E., Czarnecka, B. and Danbury, A. 2018. Emotional appeals in UK business-to-business financial services advertisements. *International Journal of Bank Marketing*, 36.

Mohammad, S. M. 2017. Challenges in sentiment analysis. (*In* Cambria, E., Das, D., Bandyopadhyay, S., Feraco, A. *eds*. A practical guide to sentiment analysis. Singapore: Springer: 61-83).

Morency, L.-P., Mihalcea, R. and Doshi, P. 2011. Towards multimodal sentiment analysis: Harvesting opinions from the web. (*In* Proceedings of the 13th international conference on multimodal interfaces, Alicante, Spain, p. 169-176).

Mundhenk, T. N., Ho, D. and Chen, B. Y. 2018. Improvements to context based self-supervised learning. (*In* Proceedings of Computer Vision and Pattern Recognition 2018 (CVPR2018), Salt Lake City, Utah, United States , p. 9339-9348).

Nezami, O. M., Dras, M., Anderson, P. and Hamey, L. 2018. Face-cap: Image captioning using facial expression analysis. (*In* Proceedings of Joint European Conference on Machine Learning and Knowledge Discovery in Databases, Dublin, Ireland: Springer, p. 226-240).

Oates, B. J. 2005. Researching information systems and computing. London: Sage.

Oliver, M. B., Raney, A. A. and Bryant, J. 2020. Media effects advances in theory and research. 4th ed. New York, NY: Routledge.

Paré, G. 2004. Investigating information systems with positivist case research. *Communications of the association for information systems,* 13(1):233-264.

Pawar, A., Jawale, M. and Kyatanavar, D. 2016. Fundamentals of sentiment analysis: Concepts and methodology. (*In* Pedrycz, W. and Chen, S. *eds*. Sentiment analysis and ontology engineering. Switzerland: Springer: 25-48).

Pérez-Rosas, V., Mihalcea, R. and Morency, L.-P. 2013. Utterance-level multimodal sentiment analysis. (*In* Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Sofia, Bulgaria, p. 973-982).

Pham, H., Guan, M., Zoph, B., Le, Q. and Dean, J. 2018. Efficient neural architecture search via parameter sharing. (*In* Proceedings of International Conference on Machine Learning, Jinan, China, p. 4092-4101).

Pham, V., Bluche, T., Kermorvant, C. and Louradour, J. 2014. Drop-out improves recurrent neural networks for handwriting recognition. (*In* Proceedings of 2014 14th International Conference on Frontiers in Handwriting Recognition, Crete, Greece, p. 285-290).

Picard, R. W. 2003. Affective computing: Challenges. *International Journal of Human-Computer Studies,* 59(1-2):55-64.

Picard, R. W. 1999. Affective computing for HCI. (*In* Proceedings of 8th International Conference on Human-Computer Interaction, Pilani, India, p. 829-833).

Picard, R. W. 1995. Affective computing. Cambridge: MIT Press.

Playstation. 2007. David Lynch Playstation Commercial. Available at: https://www.youtube.com/watch?v=msMehuZo3x8 (Date of access: 7 July 2019).

Plutchik, R. 1980. A general psychoevolutionary theory of emotion. (*In* Plutchik, R. and Kellerman, H. *eds*. Theories of emotion. New York: Elsevier: 3-33).

Politou, E., Alepis, E. and Patsakis, C. 2017. A survey on mobile affective computing. *Computer Science Review,* 25:79-100.

Poria, S., Cambria, E., Bajpai, R. and Hussain, A. 2017a. A review of affective computing: From unimodal analysis to multimodal fusion. *Information Fusion,* 37:98-125.

Poria, S., Cambria, E., Hazarika, D., Majumder, N., Zadeh, A. and Morency, L.-P. 2017b. Context-dependent sentiment analysis in user-generated videos. (*In* Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Vancouver, Canada, p. 873-883).

Poria, S., Peng, H., Hussain, A., Howard, N. and Cambria, E. 2017c. Ensemble application of convolutional neural networks and multiple kernel learning for multimodal sentiment analysis. *Neurocomputing,* 261:217-230.

Poria, S., Cambria, E., Howard, N., Huang, G.-B. and Hussain, A. 2016a. Fusing audio, visual and textual clues for sentiment analysis from multimodal content. *Neurocomputing,* 174:50-59.

Poria, S., Chaturvedi, I., Cambria, E. and Hussain, A. 2016b. Convolutional MKL based multimodal emotion recognition and sentiment analysis. (*In* Proceedings of 2016 IEEE 16th international conference on data mining (ICDM), Barcelona, Spain: IEEE, p. 439-448).

Powers, D. M. 2011. Evaluation: From precision, recall and F-measure to ROC, informedness, markedness and correlation. *Journal of Machine Learning Technologies* 2(1):37-63.

Ramchoun, H., Idrissi, M. A. J., Ghanou, Y. and Ettaouil, M. 2016. Multilayer perceptron: Architecture optimization and training. *International Journal of Interactive Multimedia and Artificial Intelligence,* 4(1):26-30.

Real, E., Aggarwal, A., Huang, Y. and Le, Q. V. 2019. Regularized evolution for image classifier architecture search. (*In* Proceedings of the AAAI Conference on Artificial Intelligence, Hilton Hawaiian Village, Honolulu, Hawaii, USA, p. 4780-4789).

Real, E., Moore, S., Selle, A., Saxena, S., Suematsu, Y. L., Tan, J., Le, Q. V. and Kurakin, A. 2017. Large-scale evolution of image classifiers. (*In* Proceedings of the 34th International Conference on Machine Learning-Volume 70, Sydney, Australia: JMLR. org, p. 2902-2911).

Rizwana, K. H. and Kalpana, B. 2018. A survey on sentiment analysis and opinion mining. *International Journal of Pure and Applied Mathematics,* 118(18):2681-2688.

Rosenblatt, F. 1958. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review,* 65(6):386.

Runge, F., Stoll, D., Falkner, S. and Hutter, F. 2018. Learning to design RNA. (*In* 2nd Workshop on Meta-Learning at NeurIPS, Montréal, Canada).

Russell, J. A. 1980. A circumplex model of affect. *Journal of Personality and Social Psychology,* 39(6):1161.

Saragih, J. M., Lucey, S. and Cohn, J. F. 2009. Face alignment through subspace constrained mean-shifts. (*In* Proceedings of 2009 IEEE 12th International Conference on Computer Vision, Kyoto, Japan: IEEE, p. 1034-1041).

Sasaki, Y. 2007. The truth of the f-measure. *Teach Tutor Mater,* 1(5):1-5.

Saxena, S. and Verbeek, J. 2016. Convolutional neural fabrics. (*In* Proceedings of Advances in Neural Information Processing Systems, Barcelona, Spain, p. 4053-4061).

Schachter, S. and Singer, J. 1962. Cognitive, social, and physiological determinants of emotional state. *Psychological Review,* 69(5):379-399.

Schwark, J. D. 2015. Toward a taxonomy of affective computing. *International Journal of Human-Computer Interaction,* 31(11):761-768.

SemEval-2019. 2019. Semeval-2019 international workshop on semantic evaluation. http://alt.qcri.org/semeval2019/ (Date of access: 20 Aug. 2019).

Shang, W., Huang, H., Zhu, H., Lin, Y., Qu, Y. and Wang, Z. 2007. A novel feature selection algorithm for text categorization. *Expert Systems with Applications,* 33(1):1-5.

Sharma, R., Le Tan, N. and Sadat, F. 2018. Multimodal sentiment analysis using deep learning. (*In* Proceedings of 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA), Orlando, Florida, USA: IEEE, p. 1475-1478).

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A. and Potts, C. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. (*In* Proceedings of the 2013 conference on empirical methods in natural language processing, Seattle, Washington, USA, p. 1631-1642).

Sochman, J. and Matas, J. 2005. Waldboost-learning for time constrained sequential detection. (*In* Proceedings of 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), San Diego, CA, USA: IEEE, p. 150-156).

Sokolova, M. and Lapalme, G. 2009. A systematic analysis of performance measures for classification tasks. *Information Processing & Management,* 45(4):427-437.

Sreeja, P. and Mahalakshmi, G. 2017. Emotion models: A review. *International Journal of Control Theory and Applications,* 10:651-657.

Stöckli, S., Schulte-Mecklenbeck, M., Borer, S. and Samson, A. C. 2018. Facial expression analysis with affdex and facet: A validation study. *Behavior research methods,* 50(4):1446-1460.

Srivastava, N. 2013. Improving neural networks with drop-out. Toronto, Canada: University of Toronto. (Thesis - MSc).

Sun, S., Luo, C. and Chen, J. 2017. A review of natural language processing techniques for opinion mining systems. *Information Fusion,* 36:10-25.

Sun, M., Yang, J., Wang, K. and Shen, H. 2016. Discovering affective regions in deep convolutional neural networks for visual sentiment prediction. (*In* Proceedings of 2016 IEEE International Conference on Multimedia and Expo (ICME), Seattle, WA, USA: IEEE, p. 1-6).

Sundermeyer, M., Schlüter, R. and Ney, H. 2012. LSTM neural networks for language modeling. (*In* Proceedings of the Thirteenth annual conference of the international speech communication association, Portland, Oregon, USA)

Swersky, K., Snoek, J. and Adams, R. P. 2014. Freeze-thaw bayesian optimization. *arXiv preprint arXiv*:1406.3896

Symeonidis, S., Effrosynidis, D. and Arampatzis, A. 2018. A comparative evaluation of pre-processing techniques and their interactions for twitter sentiment analysis. *Expert Systems with Applications,* 110:298-310.

Tian, L., Lai, C. and Moore, J. D. 2018. Polarity and intensity: The two aspects of sentiment analysis. *(In* Proceedings of 56th Annual Meeting of the Association for Computational Linguistics 2018*,* Melbourne, Australia. p. 40-47).

Tripathy, A., Agrawal, A. and Rath, S. K. 2016. Classification of sentiment reviews using n-gram machine learning approach. *Expert Systems with Applications,* 57:117-126.

Uğuz, H. 2011. A two-stage feature selection method for text categorization by using information gain, principal component analysis and genetic algorithm. *Knowledge-based Systems,* 24(7):1024-1032.

Uysal, A. K. and Murphey, Y. L. 2017. Sentiment classification: Feature selection based approaches versus deep learning. (*In* Proceedings of 2017 IEEE International Conference on Computer and Information Technology (CIT), Helsinki, Finland: IEEE, p. 23-30).

Uysal, A. K. and Gunal, S. 2012. A novel probabilistic feature selection method for text classification. *Knowledge-based Systems,* 36:226-235.

Valstar, M. 2015. Automatic facial expression analysis. (*In* Mandal, M.K. & Awasthi, A. eds. Understanding facial expressions in communication: Cross-cultural and multidisciplinary perspectives. New Delhi: Springer India: 143-172).

Varghese, R. and Jayasree, M. 2013. A survey on sentiment analysis and opinion mining. *International Journal of Research in Engineering and Technology,* 2(11):312-317.

Viola, P. and Jones, M. 2001. Rapid object detection using a boosted cascade of simple features. (*In* 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2001), Kauai, HI, USA, p. 511-518).

Vyrva, N. 2016. Sentiment analysis in social media. Halden, Norway: Østfold University College. (Thesis - Master's).

Wang, H., Meghawat, A., Morency, L.-P. and Xing, E. P. 2017. Select-additive learning: Improving generalization in multimodal sentiment analysis. (*In* Proceedings of 2017 IEEE International Conference on Multimedia and Expo (ICME), Hong Kong, China: IEEE, p. 949-954).

Werbos, P. 1974. Beyond regression: New tools for prediction and analysis in the behavioral sciences. Cambridge, Massachusetts: Harvard University. (Dissertation - PhD).

Wöllmer, M., Weninger, F., Knaup, T., Schuller, B., Sun, C., Sagae, K. and Morency, L.-P. 2013. Youtube movie reviews: Sentiment analysis in an audio-visual context. *IEEE Intelligent Systems,* 28(3):46-53.

Xie, S., Zheng, H., Liu, C. and Lin, L. 2018. SNAS: Stochastic neural architecture search. (*In* 7th International Conference on Learning Representations, Ernest N. Morial Convention Center, New Orleans).

Xu, J., Chen, D., Qiu, X. and Huang, X. 2016. Cached long short-term memory neural networks for document-level sentiment classification. (*In* Proceedings of 2016 Conference on Empirical Methods in Natural Language Processing, Austin, Texas: Association for Computational Linguistics, p. 1660–1669).

Yannakakis, G. N. 2018. Enhancing healthcare via affective computing. *Malta Journal of Health Sciences*:38.

Yu, Z. and Zhang, C. 2015. Image based static facial expression recognition with multiple deep network learning. (*In* Proceedings of the 2015 ACM on International Conference on Multimodal Interaction, Motif Hotel, Seattle, USA: ACM, p. 435-442).

Zadeh, A. 2018. Cmu-multimodalsdk. Available at: https://github.com/A2Zadeh/CMU-MultimodalSDK. (Date of access: 7 July 2019).

Zadeh, A., Liang, P. P., Poria, S., Vij, P., Cambria, E. and Morency, L.-P. 2018a. Multi-attention recurrent network for human communication comprehension. (*In* Proceedings of Thirty-Second AAAI Conference on Artificial Intelligence, Hilton New Orleans Riverside, New Orleans, Louisiana, USA, p. 5642-5649).

Zadeh, A. B., Liang, P. P., Poria, S., Cambria, E. and Morency, L.-P. 2018b. Multimodal language analysis in the wild: CMU-MOSEI data set and interpretable dynamic fusion graph. (*In* Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Melbourne, Australia, p. 2236-2246).

Zadeh, A., Chen, M., Poria, S., Cambria, E. and Morency, L.-P. 2017a. Tensor fusion network for multimodal sentiment analysis. (*In* Proceedings of 2017 Conference on Empirical Methods in Natural Language Processing, Copenhagen, Denmark: Association for Computational Linguistics, p. 1103–1114).

Zadeh, A., Chong Lim, Y., Baltrusaitis, T. and Morency, L.-P. 2017b. Convolutional experts constrained local model for 3d facial landmark detection. (*In* Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, p. 2519-2528).

Zadeh, A., Zellers, R., Pincus, E. and Morency, L.-P. 2016a. Mosi: Multimodal corpus of sentiment intensity and subjectivity analysis in online opinion videos. *arXiv preprint arXiv:1606.06259*.

Zadeh, A., Zellers, R., Pincus, E. and Morency, L.-P. 2016b. Multimodal sentiment intensity analysis in videos: Facial gestures and verbal messages. *IEEE Intelligent Systems,* 31(6):82-88.

Zela, A., Klein, A., Falkner, S. and Hutter, F. 2018. Towards automated deep learning: Efficient joint neural architecture and hyperparameter search. (*In* Proceedings of International Conference on Machine Learning (ICML) 2018 Workshop on AutoML, Stockholm, Sweden, p. 1-10).

Zhang, L., Wang, S. and Liu, B. 2018. Deep learning for sentiment analysis: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery,* 8(4).

Zhang, K., Zhang, Z., Li, Z. and Qiao, Y. 2016. Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE Signal Processing Letters,* 23(10):1499-1503.

Zhang, C. and Zhang, Z. 2014. Improving multiview face detection with multi-task deep convolutional neural networks. (*In* Proceedings of IEEE Winter Conference on Applications of Computer Vision, Steamboat Springs, Colorado, USA: IEEE, p. 1036-1041).

Zhu, X. and Ramanan, D. 2012. Face detection, pose estimation, and landmark localization in the wild. (*In* Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR): Citeseer, p. 2879-2886).

Zoph, B. and Le, Q. 2016. Neural architecture search with reinforcement learning. (*In* International Conference on Learning Representations, Caribe Hilton, San Juan, Puerto Rico).

Annexure A: Recording web application interface

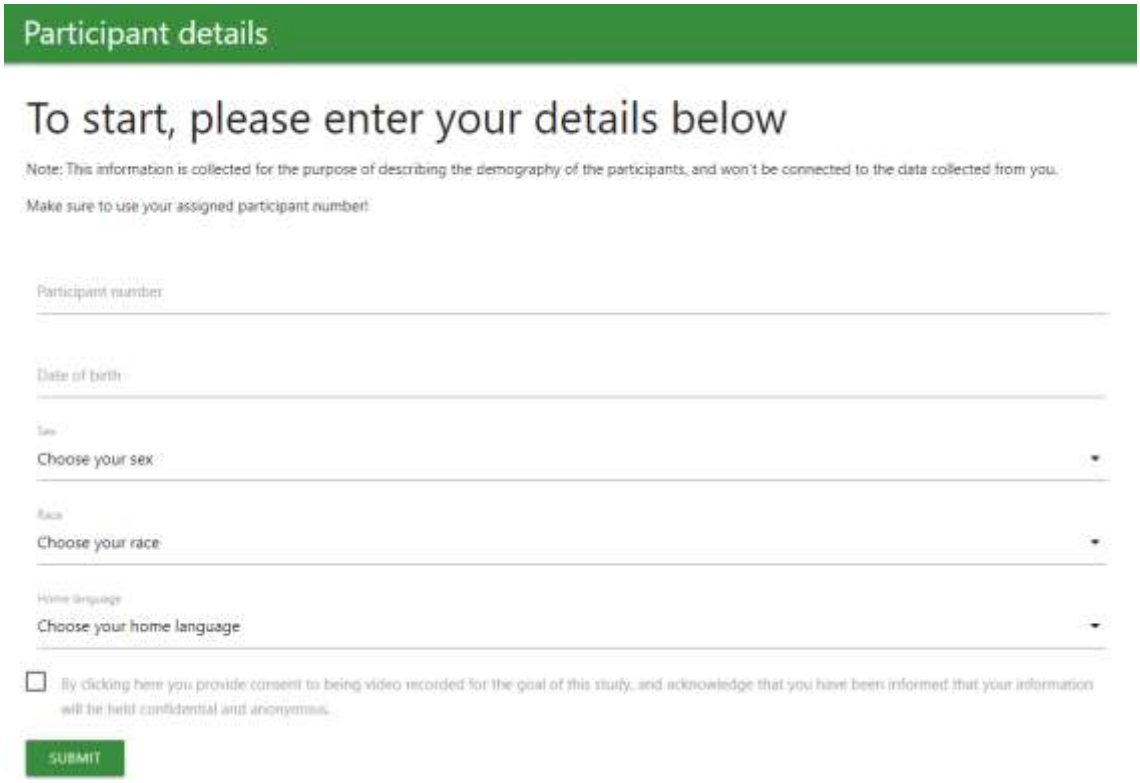


Figure A.1: Web page to collect demographic information of participants

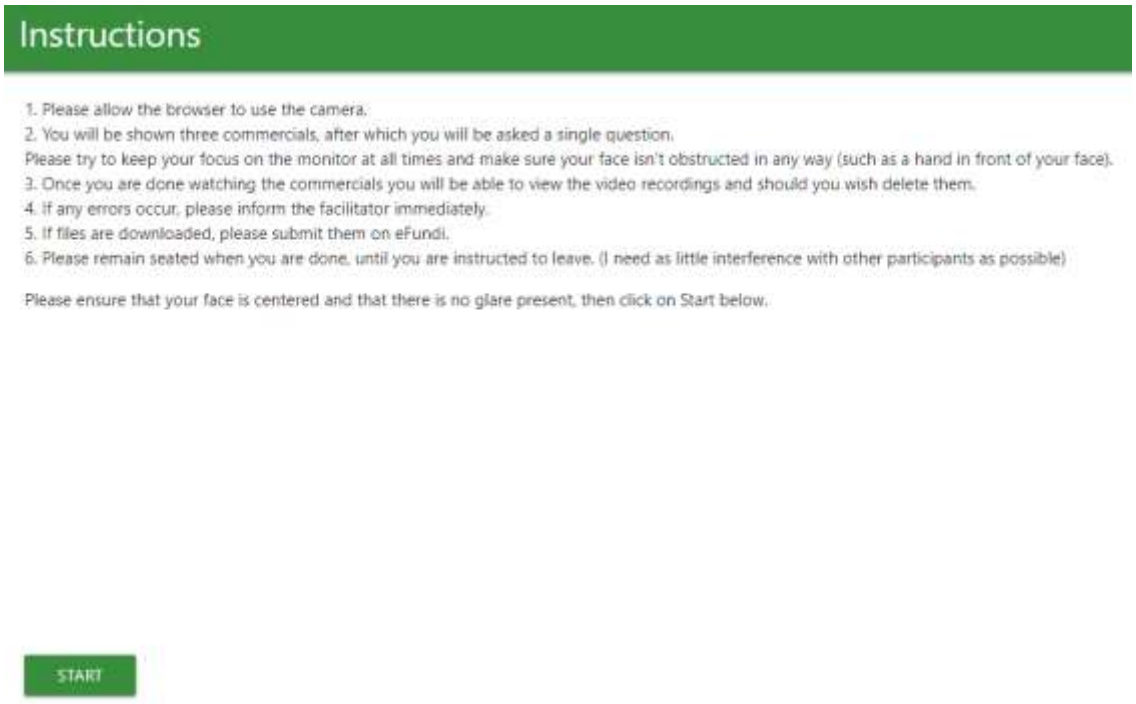


Figure A.2: Web page providing instructions to the participants

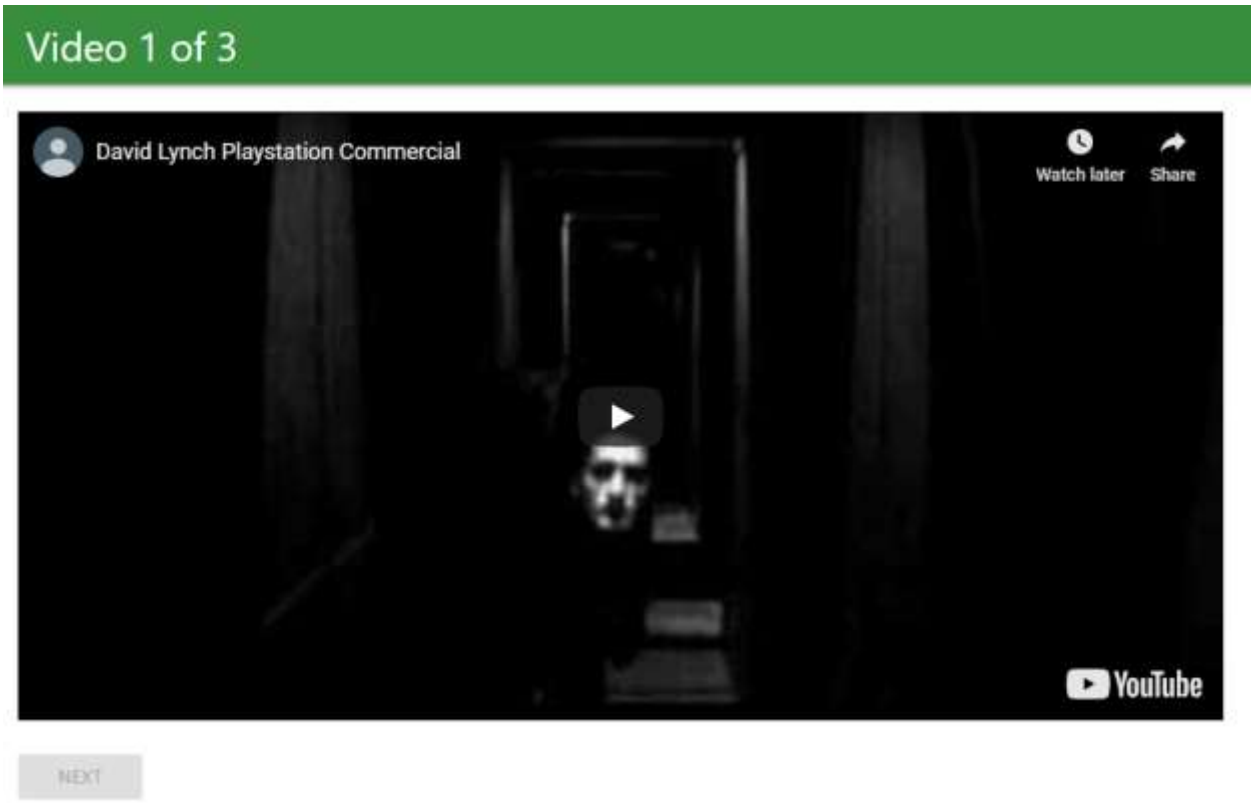


Figure A.3: Web page displaying the first video

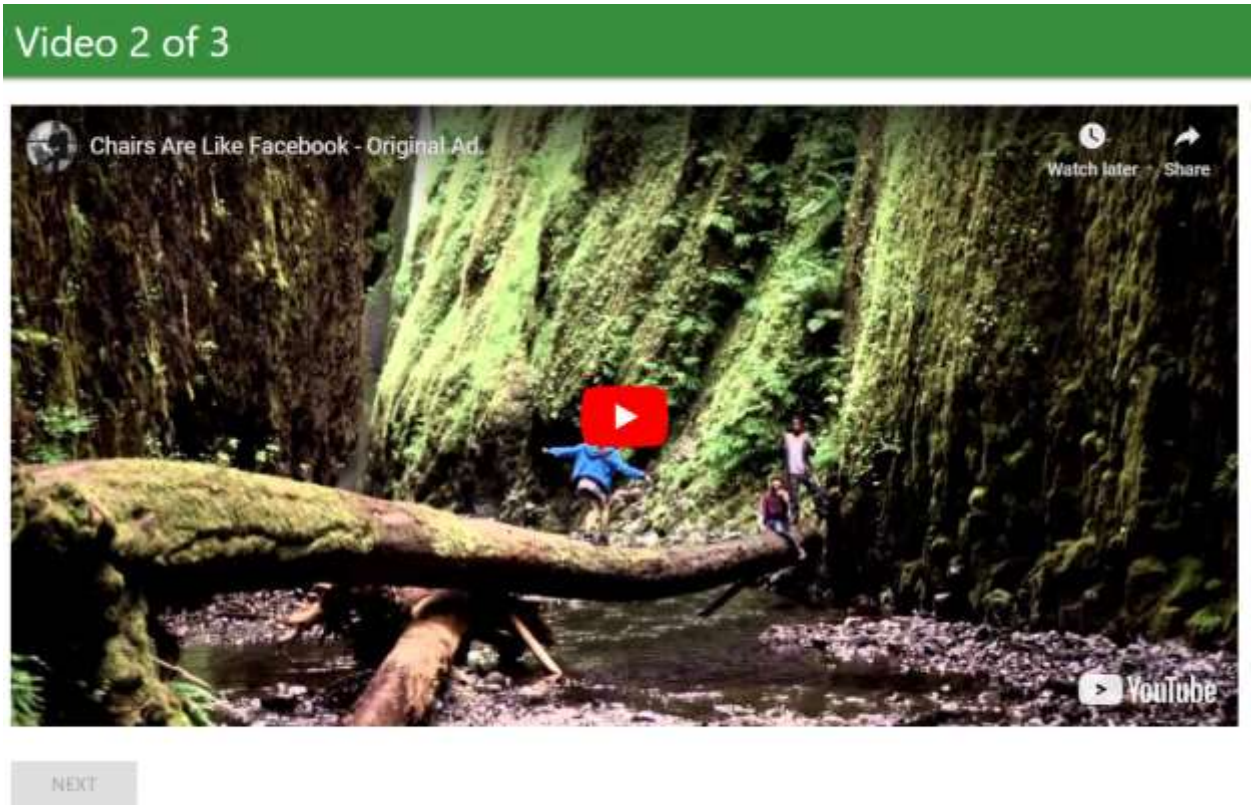


Figure A.4: Web page displaying the second video

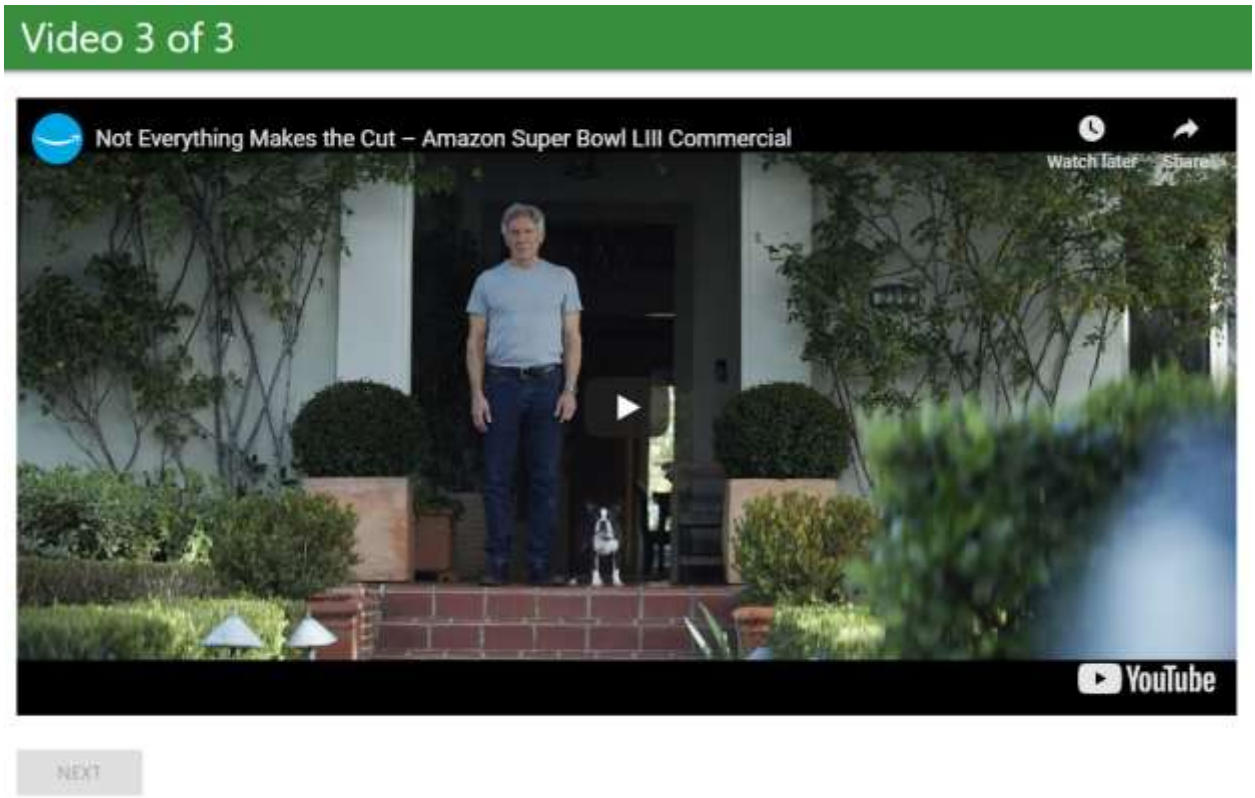


Figure A.5: Web page displaying the third video

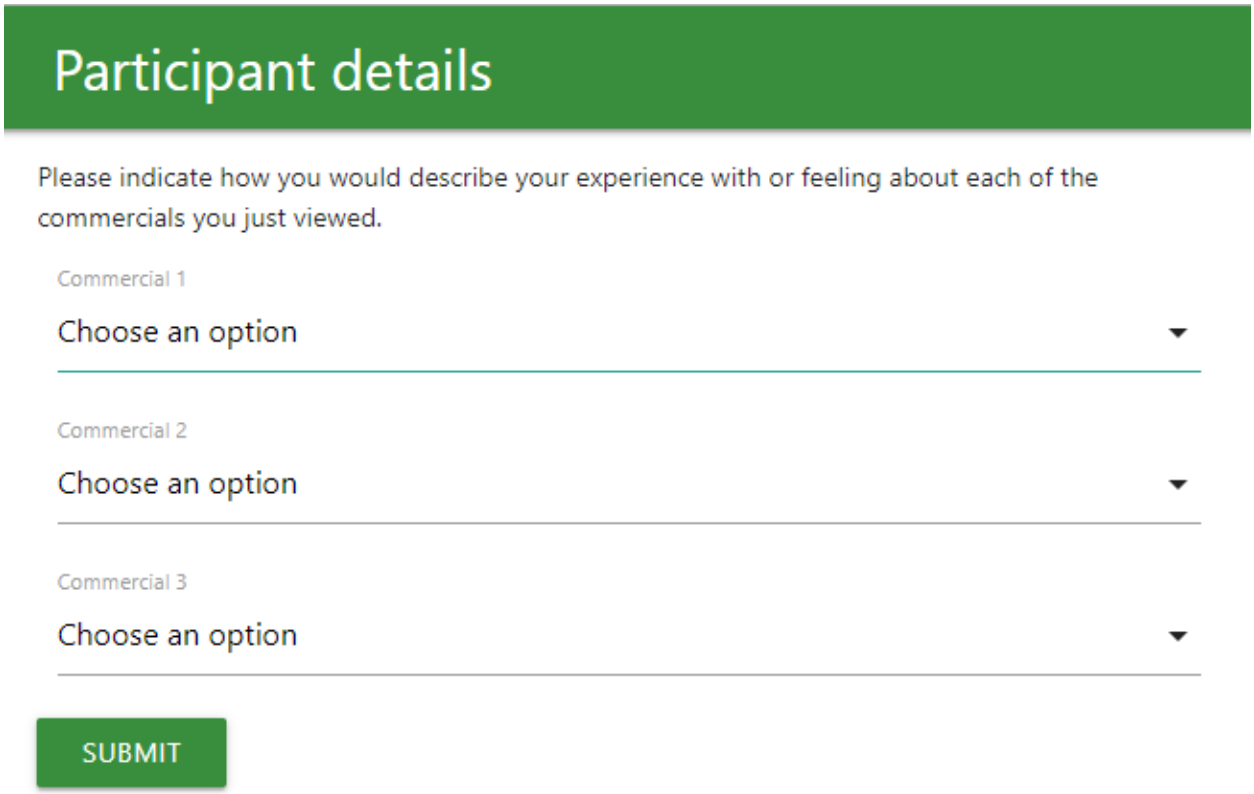


Figure A.6: Web page for rating sentiment

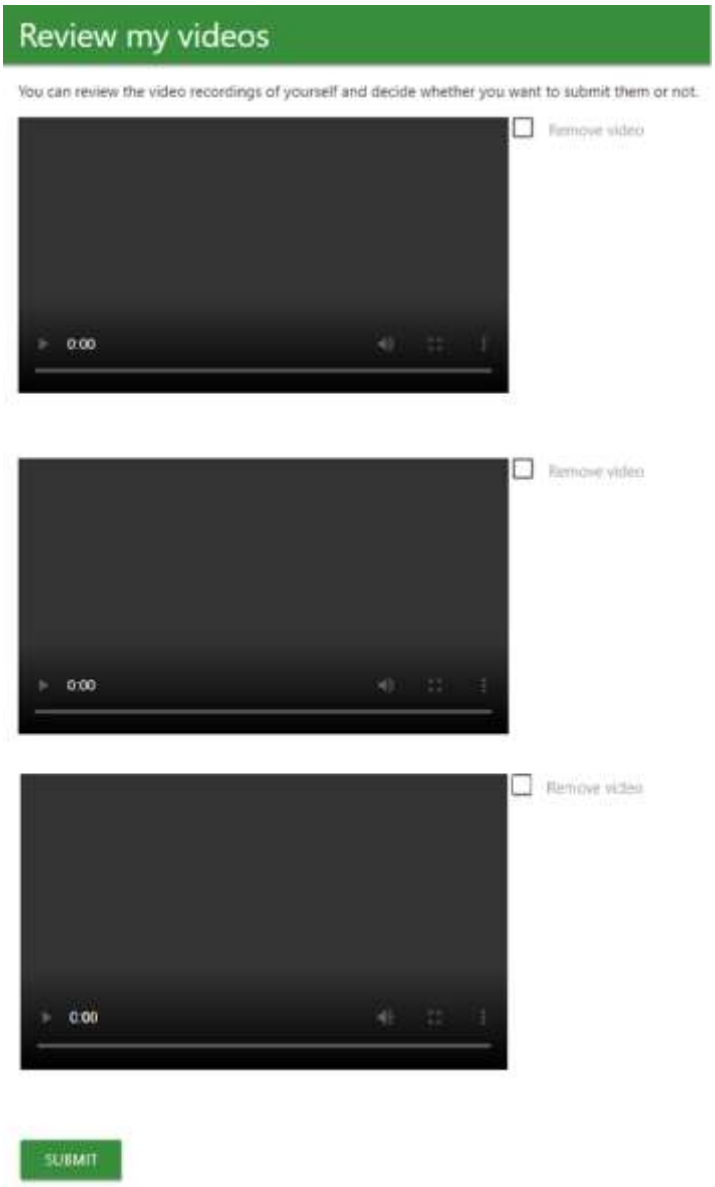


Figure A.7: Web page where participants can decide whether their videos may be included

Figure A.8: Last web page

Annexure B: Extract from the Advert22 data set

Table B.1: 15 rows of facial expression data

Table B.2: 15 rows of data including emojis, emotions, engagement and valence

Facial expressions																			
Attention	BrowFurrow	BrowRaise	CheekRaise	ChinRaise	Dimpler	EyeClosure	EyeWiden	InnerBrow Raise	JawDrop	LidTighten	LipCornerD epressor	LipPress	LipPucker	LipStretch	LipSuck	MouthOpen	NoseWrinkle	Smile	Smirk	UpperLipRai se
0.1998648	22.7271400	96.8522200	0.1004821	0.0000017	0.0269635	5.4026170	0.0507519	0.0000000	0.0000001	0.0897377	0.3034224	0.1870743	0.0098008	0.0542260	0.0158690	0.0505055	2.5705620	1.3952900	0.0024381	29.5170200
0.1981633	21.3742300	96.9685300	0.0963947	0.0000017	0.0248871	5.3915550	0.0559052	0.0000000	0.0000001	0.0886958	0.2845050	0.1869266	0.0097376	0.0500754	0.0156432	0.0487002	2.2916230	1.0587710	0.0022493	28.1507800
0.1967419	15.3979500	96.5534600	0.1034331	0.0000016	0.0182369	5.6565870	0.0629472	0.0000000	0.0000001	0.0786036	0.2744316	0.2133561	0.0095419	0.0503106	0.0130710	0.0489916	2.2960010	0.8032717	0.0019873	22.5915400
0.1958285	13.8481300	96.6270200	0.1046136	0.0000017	0.0168739	6.1383070	0.0672953	0.0000000	0.0000001	0.0751773	0.2594303	0.2222791	0.0100267	0.0486914	0.0119436	0.0480612	2.0381670	0.6343393	0.0022498	21.1955400
0.1950506	12.7301400	96.8965900	0.0986558	0.0000018	0.0157599	6.1498530	0.0659977	0.0000000	0.0000001	0.0681973	0.2514145	0.2325198	0.0104075	0.0422916	0.0111928	0.0455401	1.9424540	0.4725292	0.0017251	20.1095500
0.1945366	10.9351900	97.2709000	0.1008763	0.0000019	0.0142687	6.5760840	0.0711031	0.0000001	0.0000002	0.0626641	0.2614093	0.2400567	0.0113776	0.0372848	0.0102583	0.0427757	1.7822690	0.3685331	0.0018068	18.4937500
0.1942928	9.7828050	96.7770000	0.0877550	0.0000019	0.0129766	6.4606960	0.0658266	0.0000001	0.0000003	0.0606776	0.2552114	0.2413176	0.0119137	0.0302790	0.0088112	0.0409925	1.3446260	0.2933152	0.0011192	17.2339000
0.1939958	8.6652500	96.9768400	0.0823737	0.0000020	0.0117392	5.8098400	0.0607712	0.0000001	0.0000003	0.0651271	0.2478816	0.2639306	0.0120466	0.0349765	0.0078849	0.0399207	1.0048530	0.2214127	0.0013162	15.8835600
0.1936987	6.3299340	96.5742600	0.0944454	0.0000019	0.0096391	5.1027380	0.0515078	0.0000000	0.0000003	0.0713744	0.2470964	0.2500274	0.0130472	0.0305851	0.0084325	0.0371622	0.7637879	0.1845835	0.0012897	13.4002700
0.1934563	4.4436570	96.8523200	0.1117196	0.0000017	0.0084262	5.1017340	0.0473581	0.0000000	0.0000004	0.0789727	0.2429038	0.2562779	0.0135587	0.0326054	0.0084309	0.0365700	0.6066926	0.1731232	0.0024732	11.5817100
0.1932655	4.2825340	96.6997700	0.1106158	0.0000019	0.0084324	4.9530380	0.0476855	0.0000000	0.0000005	0.0760241	0.2280732	0.2652479	0.0144492	0.0367674	0.0083615	0.0359374	0.5476918	0.1451360	0.0042562	11.3688600
0.1929599	2.4142620	96.9887500	0.1246128	0.0000017	0.0069620	4.5838540	0.0416586	0.0000000	0.0000007	0.0736199	0.2311549	0.2744581	0.0160294	0.0460802	0.0074737	0.0346836	0.5105968	0.1147956	0.0035362	9.4909250
0.1927900	1.1431500	96.6602900	0.1268962	0.0000020	0.0062069	4.7485190	0.0440427	0.0000000	0.0000008	0.0699105	0.2183734	0.2958586	0.0177565	0.0515343	0.0076774	0.0363618	0.4568657	0.0981709	0.0049598	8.3277180
0.1927240	1.0884360	96.6650500	0.1279396	0.0000024	0.0062213	4.4270840	0.0417893	0.0000000	0.0000011	0.0642852	0.2135520	0.2939998	0.0194404	0.0672763	0.0076000	0.0358981	0.3686301	0.0952941	0.0069235	8.1586830
0.1925884	0.2226365	96.3353700	0.1362825	0.0000024	0.0055617	4.1491680	0.0380288	0.0000000	0.0000013	0.0632082	0.2200865	0.2796386	0.0221371	0.0582765	0.0078834	0.0336221	0.2897808	0.0827309	0.0084191	7.2367220

Emojis												Emotions							Other	
Disappointed	Flushed	Kissing	Laughing	Rage	Relaxed	Scream	Smiley	Smirk	StuckOutTo ngue	StuckOutTongue WinkingEye	Wink	Anger	Contempt	Disgust	Fear	Joy	Sadness	Surprise	Engagement	Valence
0.0018319	0.0018291	0.0018338	0.0017976	0.0003775	0.6251219	0.0283402	0.0025102	0.0000019	2.2977960	2.2977960	0.0018298	0.0003775	0.0211566	0.0560047	5.0876430	0.0006319	0.2328404	0.0031120	0.0106428	0.0000487
0.0018319	0.0018291	0.0018339	0.0017990	0.0004075	0.6320902	0.0283346	0.0025041	0.0000019	2.2977940	2.2977940	0.0018298	0.0004075	0.0234365	0.0617049	4.2236490	0.0006924	0.1851958	0.0034622	0.0103944	0.0000411
0.0018319	0.0018291	0.0018339	0.0018002	0.0005962	0.6294326	0.0283284	0.0024953	0.0000019	2.2977930	2.2977930	0.0018298	0.0005962	0.0355644	0.0883314	1.9541230	0.0010035	0.0742712	0.0050571	0.0113045	0.0000379
0.0018319	0.0018291	0.0018340	0.0018014	0.0006566	0.6284906	0.0283209	0.0024869	0.0000019	2.2977920	2.2977920	0.0018298	0.0006566	0.0394874	0.0977072	1.6064910	0.0011019	0.0589266	0.0056051	0.0108618	0.0000453
0.0018319	0.0018291	0.0018340	0.0018026	0.0007026	0.6267477	0.0283135	0.0024767	0.0000019	2.2977900	2.2977900	0.0018298	0.0007026	0.0428286	0.1050780	1.3788130	0.0011845	0.0492116	0.0060634	0.0101168	0.0000356
0.0018319	0.0018291	0.0018340	0.0018037	0.0007877	0.6246384	0.0283076	0.0024669	0.0000019	2.2977880	2.2977880	0.0018298	0.0007877	0.0483430	0.1174880	1.0977250	0.0013186	0.0377055	0.0067939	0.0095874	0.0000372
0.0018320	0.0018291	0.0018339	0.0018050	0.0008340	0.6223904	0.0283015	0.0024577	0.0000019	2.2977860	2.2977860	0.0018298	0.0008340	0.0531142	0.1301560	0.9185771	0.0014337	0.0306677	0.0074545	0.0089063	0.0000219
0.0018320	0.0018291	0.0018339	0.0018063	0.0008775	0.6180791	0.0282949	0.0024455	0.0000019	2.2977840	2.2977840	0.0018298	0.0008775	0.0587644	0.1443695	0.7586441	0.0015689	0.0245657	0.0082212	0.0086702	0.0000273
0.0018321	0.0018291	0.0018338	0.0018074	0.0010000	0.6088692	0.0282891	0.0024324	0.0000019	2.2977820	2.2977820	0.0018297	0.0010000	0.0707975	0.1717077	0.5332795	0.0018522	0.0163400	0.0097661	0.0080494	0.0000199
0.0018321	0.0018291	0.0018338	0.0018082	0.0011214	0.5913567	0.0282833	0.0024171	0.0000019	2.2977810	2.2977810	0.0018297	0.0011214	0.0811622	0.1948474	0.4117728	0.0020926	0.0121186	0.0110776	0.0081758	0.0000391
0.0018322	0.0018291	0.0018337	0.0018090	0.0011284	0.5739660	0.0282771	0.0024024	0.0000019	2.2977790	2.2977790	0.0018297	0.0011284	0.0824602	0.1980777	0.3994896	0.0021233	0.0116987	0.0112538	0.0075834	0.0000687
0.0018323	0.0018291	0.0018336	0.0018096	0.0012625	0.5513589	0.0282719	0.0023835	0.0000019	2.2977780	2.2977780	0.0018297	0.0012625	0.0949374	0.2241978	0.3057759	0.0024072	0.0085912	0.0127852	0.0068231	0.0000565
0.0018324	0.0018291	0.0018336	0.0018103	0.0013686	0.5373537	0.0282664	0.0023641	0.0000019	2.2977760	2.2977760	0.0018297	0.0013686	0.1035770	0.2424921	0.2590829	0.0026024	0.0070990	0.0138343	0.0058922	0.0000787
0.0018325	0.0018291	0.0018335	0.0018111	0.0013637	0.5252137	0.0282612	0.0023431	0.0000019	2.2977740	2.2977740	0.0018297	0.0013637	0.1048899	0.2462774	0.2529171	0.0026321	0.0069046	0.0140057	0.0050605	0.0001060
0.0018327	0.0018291	0.0018334	0.0018116	0.0014321	0.5096112	0.0282569	0.0023218	0.0000019	2.2977720	2.2977720	0.0018297	0.0014321	0.1123943	0.2625422	0.2217755	0.0027988	0.0059331	0.0149308	0.0038706	0.0001144

Annexure C: Python code for neural architecture search algorithm

```
import tensorflow as tf
tf.random.set_seed(256)

import time
import keras
from keras.models import Sequential
from keras.layers import Dense, Dropout
from keras.optimizers import RMSprop
from keras.models import load_model
from keras.callbacks import EarlyStopping, ModelCheckpoint, CSVLogger
import random
import numpy as np
import pickle
import copy
from os import path
from os import listdir
from os.path import isfile, join
from filelock import Timeout, FileLock

trainX = []
trainY = []
valX = []
valY = []
testX = []
testY = []
sampleSize = 50
maxPopulationSize = 200
numberOfCycles = 2000
stopIfNoImprovementAfterEpochs = 50
numberOfEpochs = 1000000
BASEPATH = '{path to data set files and where models will be stored}/'
inputNodes = 42
maxNodes = 256
minNodes = 3
maxDropout = 0.5
minDropout = 0.2
maxLayers = 10
minLayers = 3
outputNodes = 3
from keras.models import load_model

def run():
  getDataset()
  generateInitialPopulation()
```

```
  print("Done generating population")
  generateNewPopulation()

def getPopulation():
  global maxPopulationSize

history = sorted([f for f in listdir(BASEPATH) if isfile(BASEPATH + f) and
 "PopulationDetails" in f])
  print(len(history))
  if (len(history) > maxPopulationSize):
    population = history[-maxPopulationSize:]
    return population
  else:
    return history

def loadPickleFile(fileName):
  global BASEPATH

  attempts = 0
  while attempts < 3:
   try:
    if (path.exists(BASEPATH + fileName)):
     with open(BASEPATH + fileName, "rb") as file:
      result = pickle.load(file)
      return result
    else:
      return []
   except:
    attempts += 1
    time.sleep(10)
  return []

def storePickleFile(fileName, data):
  global BASEPATH
  try:
    with open(BASEPATH + fileName, "wb") as file:
      pickle.dump(data, file)
    return True
  except:
    return False

def getDataset():
  global trainX, trainY, valX, valY, testX, testY
  trainX = np.load(BASEPATH + "trainX.npy", allow_pickle=True)
  trainY = np.load(BASEPATH + "trainY.npy", allow_pickle=True)
  valX = np.load(BASEPATH + "valX.npy", allow_pickle=True)
  valY = np.load(BASEPATH + "valY.npy", allow_pickle=True)
  testX = np.load(BASEPATH + "testX.npy", allow_pickle=True)
  testY = np.load(BASEPATH + "testY.npy", allow_pickle=True)
```

```
def generateInitialPopulation():
  global maxPopulationSize
  populationSize = len(getPopulation())
  while populationSize < maxPopulationSize:
    modelArhitecture, batchSize = randomArchitecture()
    newModel = train(modelArhitecture, batchSize)
    populationSize = len(getPopulation())

def generateNewPopulation():
  global numberOfCycles

history = [f for f in listdir(BASEPATH) if isfile(BASEPATH + f) and "FullD
etails" in f]
  while (len(history) < numberOfCycles):
   try:
    sample = getSampleFromPopulation()
    parent = max(sample, key=lambda x:x['accuracy'])
    oldModel = loadPickleFile(parent["modelName"] + "FullDetails.pickle")
    mutationRate = random.randint(1,2)

modelArhitecture, batchSize = mutate(oldModel["architecture"], oldModel["b
atchSize"], mutationRate)
    newModel = train(modelArhitecture, batchSize)
   except:
    pass

def getSampleFromPopulation():
  global sampleSize
  sample = []
  population = getPopulation()
  while len(sample) < sampleSize:
    candidate = population[random.randint(0, len(population) - 1)]
    while (candidate in sample):
      candidate = population[random.randint(0, len(population) - 1)]
    sample.append(loadPickleFile(candidate))
  return sample

def train(modelArchitecture, batchSize):

global BASEPATH, stopIfNoImprovementAfterEpochs, trainX, trainY, valX, val
Y, testX, testY

  modelName = "model" + str(time.time())
  keras_callbacks = [

EarlyStopping(monitor='val_loss', mode='min', verbose=0, patience=stopIfNo
ImprovementAfterEpochs, min_delta=0),
  ]
```

```
  start = time.time()

trainHistory = modelArchitecture.fit(trainX, trainY, batch_size=batchSize,
 epochs= numberOfEpochs,
                     validation_data=(valX, valY), verbose=0, callbacks=ke
ras_callbacks)
  stop = time.time()
  modelDetails = {
    "modelName": modelName,
    "accuracy": trainHistory.history["val_accuracy"][-1],
    "loss": trainHistory.history["val_loss"][-1],
    "testAccuracy": modelArchitecture.evaluate(testX, testY, verbose=0),
    "batchSize": batchSize,
    "trainTime": (stop - start),
    "architecture": modelArchitecture,
    "trainHistory": trainHistory
  }
  storePickleFile(modelName + "FullDetails.pickle", modelDetails)
  modelDetailsForPopulation = {
    "modelName": modelName,
    "accuracy": trainHistory.history["val_accuracy"][-1]
  }

storePickleFile(modelName + "PopulationDetails.pickle", modelDetailsForPop
ulation)

  return modelDetails

def randomArchitecture():

global minDropout, maxDropout, maxNodes, minNodes, minLayers, maxLayers, i
nputNodes, outputNodes
  numNodes = random.randint(minNodes, maxNodes)
  numLayers = random.randint(minLayers, maxLayers)
  modelArchitecture = Sequential()

modelArchitecture.add(Dense(numNodes, activation='relu', input_shape=(inpu
tNodes,)))
  dropOut = random.uniform(minDropout, maxDropout)
  modelArchitecture.add(Dropout(dropOut))
  for i in range(1, numLayers):
    numNodes = random.randint(minNodes, maxNodes)
    modelArchitecture.add(Dense(numNodes, activation='relu'))
    dropOut = random.uniform(minDropout, maxDropout)
    modelArchitecture.add(Dropout(dropOut))
  modelArchitecture.add(Dense(outputNodes, activation='softmax'))

modelArchitecture.compile(loss='categorical_crossentropy', optimizer=RMSpr
op(),metrics=['accuracy'])
```

```
  batchSize = getBatchSize()
  return modelArchitecture, batchSize

def getBatchSize():
  return 2 ** random.randint(5, 12)

def mutate(parent, batchSize, mutationRate):
  mutateHyperparameters = []
  for i in range(0, mutationRate):

hyperparameterToMutate = random.randint(1, 4) #decide which hyperparameter
s to mutate
    while (hyperparameterToMutate in mutateHyperparameters):
      hyperparameterToMutate = random.randint(1, 4)
    mutateHyperparameters.append(hyperparameterToMutate)
    if (hyperparameterToMutate == 1):
      model = mutateDropoutRates(parent)
    elif (hyperparameterToMutate == 2):
      model = mutateNumLayers(parent)
    elif (hyperparameterToMutate == 3):
      model = mutateNumNodesInLayerN(parent)
    else:
      model = parent
      batchSize = getBatchSize()

model.compile(loss='categorical_crossentropy', optimizer=RMSprop(),metrics
=['accuracy'])
  return model, batchSize

def mutateDropoutRates(model):
  global minDropout, maxDropout, outputNodes
  layers = [l for l in model.layers]
  changeLayer = random.randint(1, len(layers) -
 1) #don't alter input or output layers
  while ("dropout" not in layers[changeLayer].name):
    changeLayer = random.randint(1, len(layers) - 1)
  drpOut = random.uniform(minDropout, maxDropout)
  layers[changeLayer].rate = drpOut
  newModel = createNewModelBasedOnOldModelLayers(layers)
  return newModel

def mutateNumNodesInLayerN(model):
  global minNodes, maxNodes
  layers = [l for l in model.layers]
  layerToRandomize = random.randint(1, len(layers) - 1)
  while ("dense" not in layers[layerToRandomize].name):
    layerToRandomize = random.randint(1, len(layers) - 1)
  layers[layerToRandomize].units = random.randint(minNodes, maxNodes)
  newModel = createNewModelBasedOnOldModelLayers(layers)
```

```
  return newModel

def createNewModelBasedOnOldModelLayers(layers):
  global inputNodes
  newModel = Sequential()

newModel.add(Dense(layers[0].units, activation='relu', input_shape=(inputN
odes,)))
  try:
    newModel.layers[0].set_weights(layers[0].get_weights())
  except:
    pass
  for l in range(1, len(layers) -
 1): #only add layers with relu activation function
    if ("dense" in layers[l].name):
      newModel.add(Dense(layers[l].units, activation='relu'))
      try:
        newModel.layers[l].set_weights(layers[l].get_weights())
      except:
        pass
    elif ("dropout" in layers[l].name):
      newModel.add(Dropout(layers[l].rate))
  newModel.add(Dense(outputNodes, activation='softmax'))
  try:
    newModel.layers[len(newModel.layers) -
 1].set_weights(layers[len(layers) - 1].get_weights())
  except:
    pass
  return newModel

def mutateNumLayers(model):
  global minLayers, maxLayers
  layers = [l for l in model.layers]
  newNumLayers = random.randint(minLayers, maxLayers)
  numHiddenLayers = (len(layers) -
 1) // 2 #subtract 1 to ignore the output layer (divide by 2 to ignore dro
pout layers as they are seen as part of the hiden layer)
  if (newNumLayers > numHiddenLayers): #add layers
    newLayers = addNewLayers(layers, newNumLayers - numHiddenLayers)
    return createNewModelBasedOnOldModelLayers(newLayers)
  else: #remove layers
    newLayers = removeLayers(layers, numHiddenLayers - newNumLayers)
    return createNewModelBasedOnOldModelLayers(newLayers)

def addNewLayers(layers, numLayersToAdd):
  global minNodes, maxNodes, minDropout, maxDropout
  for k in range(0, numLayersToAdd):
    insertAt = random.randint(0, len(layers) -
 1) #cannot insert after output layer
```

```
    numNodes = random.randint(minNodes, maxNodes)
    drpOut = random.uniform(minDropout, maxDropout)
    if ("dense" in layers[insertAt].name):
      layers.insert(insertAt, Dense(numNodes, activation='relu'))
      layers.insert(insertAt + 1, Dropout(drpOut))
    else:
      layers.insert(insertAt - 1, Dense(numNodes, activation='relu'))
      layers.insert(insertAt, Dropout(drpOut))
  return layers

def removeLayers(layers, numLayersToRemove):
  for k in range(0, numLayersToRemove):
    removeAt = random.randint(2, len(layers) -
 2) #cannot remove input layer, its dropoutlayer nor the output layer
    if ("dense" in layers[removeAt].name):
      layers.pop(removeAt) #remove dense layer
      layers.pop(removeAt) #remove dropout layer
    else:
      layers.pop(removeAt - 1) #remove dense layer
      layers.pop(removeAt - 1) #remove dropout layer
  return layers

run()
```

Annexure D: Full conference paper presented at SATNAC 2019

Affective Computing and Deep Learning to Perform Sentiment Analysis

Nicolaas Maree[1], Lynette Drevin[2], Tiny du Toit[3], Hennie Kruger[4]

School of Computer Science and Information Systems

North-West University, Potchefstroom Campus, South Africa

[1]Nicolaas.Maree@nwu.ac.za

[2]Lynette.Drevin@nwu.ac.za

[3]Tiny.DuToit@nwu.ac.za

[4]Hennie.Kruger@nwu.ac.za

Abstract—**The opinions of others are one of the main influencers of human behaviour and activities. Therefore, individuals and organizations often consult with others to understand their opinions or attitudes towards a certain topic, before making decisions. Also, for telecommunication enterprises to survive, they need to be attentive to their customers' opinions. Sentiment analysis is a technique that is often used by organizations to categorize and understand the underlying attitude of a person towards an entity, product, topic, etc. Though it has been traditionally performed using text-based sources, it has been suggested that other modalities should be explored. One such alternative to text-based sources is video recordings of people using or reviewing content. Videos can contain multiple modals including text, voice, and facial expressions, which can be used to detect a person's attitude towards a topic. An approach to performing sentiment analysis using affective computing for extracting an opinion holder's affective data based on their facial expressions, and then feeding this data to a deep learning multilayer perceptron neural network, is proposed in this paper. The results of this study indicate that the proposed approach is highly feasible to gain accurate insights into a person's sentiment towards a specific topic.**

Keywords — affective computing, deep learning, sentiment analysis

I. INTRODUCTION

Opinions play a central role in influencing how humans act and behave [1]. Worldviews, or the way one perceives the world around them, are often formed by the way those people surrounding a person perceive the world. Organizations, similar to individuals asking their peers for their opinions, often rely on the opinions of their customers when they need to make decisions. This can be done using traditional means such as customer satisfaction questionnaires. However, due to high costs and issues with availability related to obtaining feedback, organizations explored other avenues into their customers' opinions. One such avenue into understanding other's opinions or sentiment is the use of sentiment analysis

With the increase of media posted to social media websites, organizations, such as telecommunications enterprises, have the opportunity to gain insight into their customers' opinions [2]. This allows them to identify their strengths and weaknesses, as well as to identify new opportunities and threats. Previous research has been done to determine how telecommunications enterprises can make use of text-based sentiment analysis to improve customer satisfaction and improve the overall user experience [2-4].

Sentiment analysis is traditionally done using text-based reviews published to the Internet [5]. Nevertheless, the increase in popularity of social media platforms and the sharing of opinions on these platforms in forms other than text, such as video, audio, and images, has made it necessary to explore these other modalities to use for performing sentiment analysis [6].

According to [7] the use of videos for performing sentiment analysis has the advantage of including a magnitude of behavioural cues to detect the affective state of the opinion holder. The field of affective computing can help with the identification of human emotions, since it entails computing that "relates to, arises from, or influences emotions" [8].

Based on this information an approach to accurately perform sentiment analysis using affective computing to identify the emotions of opinion holders in videos based on their facial expressions, and a deep learning multilayer perceptron (MLP) neural network, is presented.

A literature review covering the topics of sentiment analysis, affective computing, and deep learning neural networks are presented in Section II, III, IV of this paper, respectively. In Section V the experimental design is discussed with regards to the data collection process and the architecture of the resulting MLP. The results obtained from the MLP, as well as an evaluation of the performance of the MLP, are presented in Section

VI. Some concluding remarks and possible future work are discussed in Section VII.

II. Sentiment Analysis

Opinions of individuals are a vital factor that needs to be considered by organizations before they make decisions [9]. People often review products or provide their opinions on specific issues through blog posts, or discussion forums. Because of the increased availability of content on the Internet and social media there exists a wealth of data that may be used by them.

Data scientists can analyse this data to mine for patterns that could be of interest to organizations. These patterns are used to understand customers, and to improve an organization's sales and marketing strategies [1]. Sentiment analysis refers to discovering the opinions, or sentiment towards a certain object, fact, or attributes. A sentiment consists of four components, i.e. the entity, the aspect, the opinion holder, and the aspect's sentiment [10]. Thus, sentiment analysis should successfully extract these components from the given source.

The application of text-based sentiment analysis has been well researched within a broad range of fields. Yet, it remains a challenge as it requires a deep understanding of language, both in terms of semantics and syntax [11]. Recently a shift in the media published to the Internet has been noticed which also opens new doors for the field of sentiment analysis. There are three main lines for multimodal sentiment analysis, i.e. sentiment analysis in (1) spoken reviews, (2) human-machine interaction and human-human interaction, and (3) images and their associated tags [10]. In this study, videos will be used to detect the participants' sentiment using their facial expression-based emotions.

The combination of computer vision with sentiment analysis is a recent avenue of research being undertaken [10]. This task can be accomplished by detecting, modelling, and leveraging the sentiment expressed by facial or body gestures. The extraction of emotions for use in sentiment analysis is a common task done in the field of affective computing.

Poria *et al.* [6] utilized YouTube videos to perform sentiment analysis based on visual, audio, and textual aspects. They made use of facial expressions as part of their visual features fused to the audio and textual data. The training sets were used with a support vector machine (SVM), an artificial neural network, and an extreme learning machine. In another study done by [12] 40 visual features obtained from frame level, were used with audio and text features. Experiments using an SVM were run using one, two, and three modalities at a time.

III. Affective computing

The field of affective computing is concerned with giving computers the ability to detect, process, express, communicate and respond to human emotions [13]. Emotions play a major role in humans' everyday lives. It influences cognition, perception, communication, and rational decision making [14]. Therefore, it is an important aspect to consider when designing computer systems.

Humans interact with each other using their facial expressions, body gestures, and speech. By incorporating these methods of communication affective computing leads to advancements in the area of human-computer intelligent interaction. Thus, interfacing with a computer can move beyond the traditional use of a keyboard and mouse.

Computers can be made aware of emotions by detecting emotions through facial expressions, body gestures, speech, and other physiological signals. For this paper, the Affectiva© software development kit (SDK) [15] was used to detect facial expressions.

The Affectiva© system was trained on a facial expression data set which consisted of 242 videos of faces, with 168359 frames of people watching Super Bowl commercials, which were recorded using their webcams [16]. The system has four main components for the classification of emotional states, i.e. (1) detection of faces and facial landmarks, (2) feature extraction from face texture, (3) classification of facial actions, and (4) modelling emotion expression.

The Viola-Jones face detection algorithm is used to detect faces in each frame, after which 34 landmarks are identified in each facial bounding box. Should a specific threshold for the confidence of the landmark detection not be reached, that bounding box is ignored. A screenshot of the iOS Affectiva© SDK demo application indicating the facial landmarks is shown in Fig. 1.

Figure 1: Screenshot of iOS SDK demo application indicating facial landmarks

The identified landmarks are used to extract Histogram of Oriented Gradient features from image regions of interest. These features are processed by an SVM classifier to assign a score of 0 to 100 for each facial action. The combination of the resulting scores is then used to determine emotion expressions. Finally, the emotion expressions are each assigned a score of 0 to 100, where 0 indicates that it is absent and 100 indicates that it is strongly present.

Ekman [17] identified six emotions that are universal among all humans, i.e. happiness, sadness, anger, fear, surprise, and disgust. These emotions are included within the metrics produced by the Affectiva© system, along with an additional emotion, i.e. contempt, 21 facial expressions, such as open mouth, smile, brow

raise, etc., 13 emojis, such as a wink, smiley, stuck out tongue, etc. It also contains values for valence and engagement, resulting in a total of 42 metrics.

The primary focus of this study is making use of the emotions identified by [17] to perform sentiment analysis.

IV. Deep learning neural networks

Deep learning emerged from the field of machine learning in 2006 [18]. It broadly refers to machine learning techniques using neural networks which (1) consist of multiple layers of nonlinear nodes, and (2) which uses supervised or unsupervised learning methods at consecutive higher levels of abstracted layers for feature representation.

Supervised deep learning refers to deep learning techniques where the collected data is labelled before feeding it to the deep learning neural network [18]. On the other hand, unsupervised deep learning can be classified as the techniques that do not require a human to label the data set and are usually focused on the capturing of the high-order correlation within the observed data [19].

The most basic form of a deep learning neural network is known as the multilayer perceptron (MLP) neural network, which is based on the Perceptron proposed by Rosenblatt in 1950 [19, 20]. An MLP consists of an input and output layer and contains multiple hidden layers in between these layers. It receives an input x which is then mapped to a category y, by successively sending the input values from one layer of nodes to the next layer of nodes. Thus, it can be expressed as

$$y = f(x, \theta), \tag{1}$$

where θ indicates the parameters, i.e. connection weights and biases, used by the MLP to learn. It is important to note that an MLP does not contain any connections which send the output values of a higher level to nodes on a lower level. Each layer of nodes contains certain parameters that help the MLP to learn.

Learning refers to the adjusting of connection weights within the MLP to minimize the error between the desired output and the produced output [20]. A method that is popularly used for training an MLP is the backpropagation algorithm [21]. The algorithm receives a set of inputs,

$$\{\boldsymbol{p_1}, \boldsymbol{t_1}\}, \{\boldsymbol{p_2}, \boldsymbol{t_2}\}, \dots, \{\boldsymbol{p_Q}, \boldsymbol{t_Q}\}, \tag{2}$$

where each input contains an input value ($\boldsymbol{p}_Q$) mapped to a target output value ($\boldsymbol{t}_Q$). As the MLP processes each of these inputs, it adjusts its parameters based on the calculated mean square error. This process can be summarised into the following three steps:

1. Forward propagate the inputs through the MLP.
2. Calculate and backwards propagate the sensitivities through the MLP.
3. Modify the MLP's parameters accordingly.

For the first step, the outputs of a layer which is then used as input for the subsequent layer is expressed as

$$\boldsymbol{a}^{m+1} = \boldsymbol{f}^{m+1}(\boldsymbol{W}^{m+1}\,\boldsymbol{a}^{m} + \boldsymbol{b}^{m+1}) \; for m = 0, 1, \dots, M-1, \tag{3}$$

where $\boldsymbol{f}$ is the activation function, $\boldsymbol{W}^n$ and $\boldsymbol{b}^n$ refers to the weight vector and bias of layer n, respectively, M is the number of layers within the MLP, and its starting point is given by

$$\boldsymbol{a}^0 = \boldsymbol{p}, \tag{4}$$

where p represents the original input vector and the MLP's last layer's output is the output of the MLP, i.e.

$$\boldsymbol{a} = \boldsymbol{a}^M. \tag{5}$$

To calculate the sensitivities in the second step the following equations are applied:

$$\boldsymbol{s^M} = -\boldsymbol{2\dot{F}^M}(\boldsymbol{n^M})(\boldsymbol{t} - \boldsymbol{a}), \tag{6}$$

where $\boldsymbol{n}$ refers to the net input, $\boldsymbol{t}$ indicates the target or expected outputs, and

$$\dot{\boldsymbol{F}}^m(\boldsymbol{n}^m) = \begin{bmatrix} \dot{f}^m(n_1^m) & 0 & \cdots & 0 \\ 0 & \dot{f}^m(n_2^m) & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & \dot{f}^m(n_{S^m}^m) \end{bmatrix}. \tag{7}$$

where

$$\boldsymbol{s}^m = \dot{\boldsymbol{F}}^m(\boldsymbol{n}^m)(\boldsymbol{W}^{m+1})^T \boldsymbol{s}^{m+1}. \tag{8}$$

Finally, the biases and weights of the MLP can be adjusted. This is done using the mean square error and is calculated as follows:

$$\boldsymbol{W}^m(k+1) = \boldsymbol{W}^m(k) - \alpha \boldsymbol{s}^m(\boldsymbol{a}^{m-1})^T, \tag{9}$$

and

$$\boldsymbol{b}^m(k+1) = \boldsymbol{b}^m(k) - \alpha \boldsymbol{s}^m, \tag{10}$$

at iteration k, with a learning rate represented by α.

V. Experimental design

The collection of data, the grouping of the data into training, validation, and test data sets, as well as the setup of the deep learning MLP neural network, are discussed in this section.

A. Data collection

The proposed approach makes use of affective data that are extracted from video recordings. Therefore, affective data (specifically facial expressions), were collected from nine participants, consisting of Honours and Masters Computer Science students. The group consisted of 1 female and 8 male participants.

Three text passages were identified that would evoke one of three main reactions, or sentiment, from the participants. The three categories for the passages were classified as positive, neutral, and negative. The participants were informed that they would be recorded as they read the passages. However, they were only informed afterwards about which metrics were to be extracted from the videos (i.e. emotions), as to prevent them from possibly being self-conscious about their reactions, and to not be influenced to react in a certain way.

After recording the participants, each video was processed using the Affectiva© SDK to extract the 42 metrics indicating a probability that the participant is experiencing that metric. Therefore, 42 input values for the MLP were generated at each frame that was analysed. Table I provides an example of values for four of the extracted metrics at five consecutive frames.

TABLE I

SAMPLE OF DATA EXTRACTED FROM PARTICIPANT VIDEOS

Time-stamp (s)	Disgust (%)	Fear (%)	Surprise (%)	Brow Raise (%)
8.48	99.92004	99.57072	95.69440	28.01171
8.52	99.91998	99.57033	95.68865	27.95599
8.56	99.91998	99.57021	94.37222	28.73918
8.60	99.92001	99.57047	93.46150	29.60226
8.64	99.91998	99.57097	91.47912	29.15755

The pre-processing of all the videos created a data set consisting of 132261 data points. The number of data points extracted from each video, as shown in Table II, varied for each participant. This may be due to each participant having a different reading speed. However, it is believed that this did not influence the results. Before it was presented as input to the MLP, the order of the extracted data points was randomised, and the data were grouped into three data sets, i.e. training-, validation-, and test data sets, in a relation of 70%, 20%, and 10%, respectively.

B. MLP architecture

As previously mentioned, the primary focus of the study was to make use of the six universal emotions identified by [17] to perform sentiment analysis. Thus, an MLP was developed and trained on the data set consisting of the affective data as discussed in the previous section. The input layer was followed by five hidden layers each consisting of a different number of nodes which made use of the Rectified Linear Unit (ReLU) activation function. The output of the last hidden layer was propagated to the nodes in the output layer utilizing the softmax activation function. This function assigns a probability distribution to each of the three sentiment categories, i.e. positive, neutral, or negative. The architecture indicating the number of nodes in each hidden layer is shown in Fig. 2.

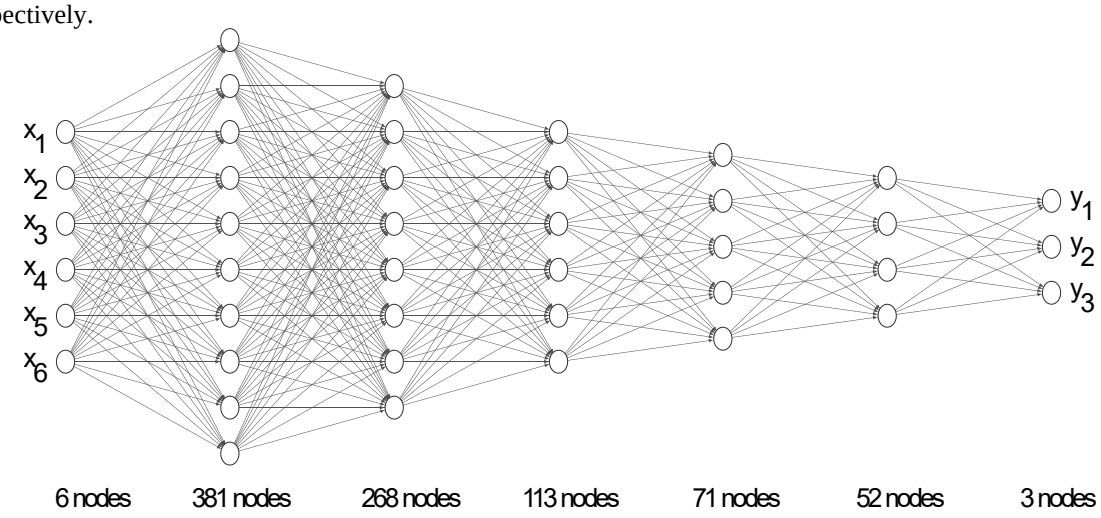


Figure 2: Architecture of MLP based on Ekman's six emotions

TABLE II

NUMBER OF SAMPLES COLLECTED PER PARTICIPANT PER PASSAGE

Participant Number	Passage 1	Passage 2	Passage 3	Total
1	4650	6324	4556	15530
2	5870	5746	5055	16671
3	2965	2757	2881	8603
4	2706	4327	3563	10596
5	5189	6352	4417	15958
6	2238	7633	3307	13178
7	498	5964	5857	12319
8	7025	9769	6189	22983
9	4437	5877	6109	16423
Total	35578	54749	41934	132261

The MLP was implemented using the Keras high-level deep learning API's [22] RMSProp optimizer, and its categorical cross-entropy loss function. A batch size of 2048 was used for training the model. Additionally, a total number of 147437 trainable parameters were used in the model. The distribution of these parameters over the layers is shown in Table III.

To determine if the first MLP model's accuracy could be improved a second MLP architecture was developed which included additional metrics produced by the Affectiva© system. This model consisted of 42 input nodes, five hidden layers utilizing the ReLU activation function, and three output nodes using the softmax activation function. The numbers of hidden nodes in each of the hidden layers were kept the same as in the first model. The number of weights for the connections between the input layer and the first hidden layer increased from 2286 to 16383, resulting in a total of 161534 trainable parameters.

The results obtained using the collected affective data and the two above-mentioned MLP architectures are presented in the following section.

TABLE III

PARAMETERS USED IN MLP WITH SIX INPUT NODES

Layer	Number of Weights	Number of Biases	Number of Parameters
Hidden layer 1	2286	381	2667
Hidden layer 2	102108	268	102376
Hidden layer 3	30284	113	30397
Hidden layer 4	8023	71	8094
Hidden layer 5	3692	52	3744
Output layer	156	3	159
Total	146549	888	147437

VI. RESULTS

Both the models described in Section V were trained and evaluated on an Intel Core i7-6950X computer with 64GB RAM, and NVidia GTX 1080 GPU. The duration of training the first model was 3 hours and 13 minutes, while the second model trained in three 3 hours and 10 minutes.

The historical learning curve of the validation accuracy compared to the loss for this model over 35000 epochs is shown in

. It can be seen that the model had a sharp initial learning curve for the first thousand epochs but started to increase only slightly until it reached its peak at 33181 epochs. A validation accuracy of 69.98% was obtained at this epoch.

The MLP was further applied to the test data set to predict the out-of-sample class of each data point. The results obtained from the predictions using the MLP trained with only six input nodes are presented in Table IV. The model predicted 1274 out of 3468, of the cases labelled as positive correctly, resulting in a prediction accuracy of 52.10%. It further had a prediction accuracy of 81.75% for the 5464 cases presented to it labelled as neutral. For the 4294 data points labelled as negative an accuracy of 71.87% obtained, by classifying 3086 data points correctly. This resulted in an overall accuracy of 66.74% and a loss score of 0.74.

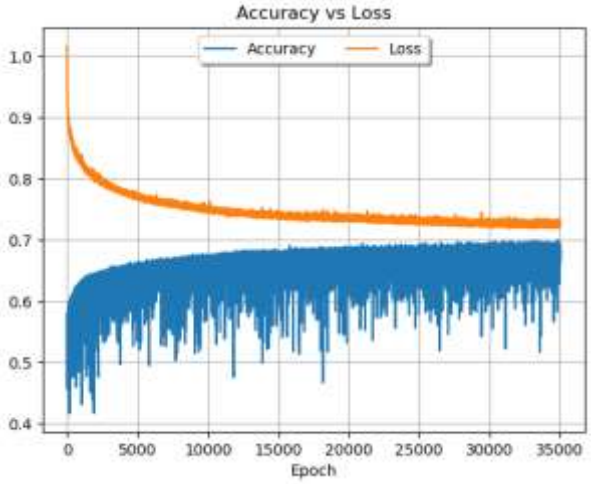


Figure 3: Validation accuracy versus loss for 35000 epochs for the MLP with six inputs

TABLE IV

PREDICTION RESULTS OF MLP WITH SIX INPUT NODES

	Predicted Category			
True Category	Positive	Neutral	Negative	Accuracy (%)
Positive	1274	1807	387	52.10
Neutral	526	4467	471	81.75
Negative	316	892	3086	71.87

The second model's validation accuracy compared to the loss during the training process for 35000 epochs is shown in Fig. 4. The highest value for the validation accuracy, i.e. 99.77%, was obtained after 34264 epochs. The model had an initial sharp increase in its accuracy but started to converge after about 1500 epochs.

The prediction results for the MLP trained with 42 input nodes are shown in Table V. Of the 7157 records that were labelled as positive, 7131 cases were predicted correctly, giving an accuracy of 99.64%. For the neutral cases, 10857 out of 10888 were correctly predicted with an accuracy of 99.72%. Finally, a classification accuracy of 99.70% was obtained for the records that were labelled as negative, where 8382 out of the 8407 cases were correctly predicted.

The overall accuracy obtained using the test data set was similar to the validation accuracy at 99.69%, and a loss score of 0.01. These results will be discussed in the next section.

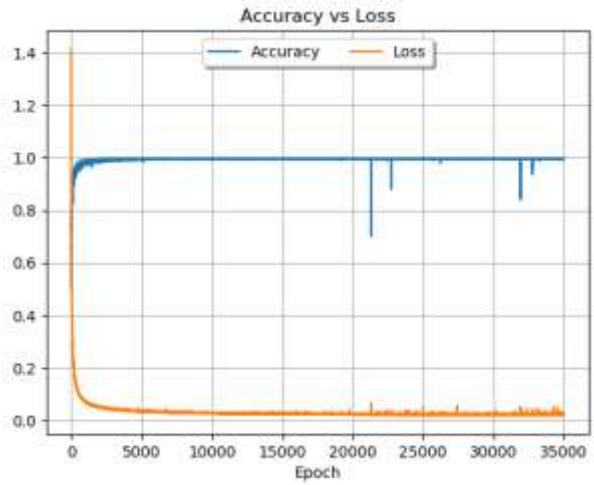


Figure 4: Validation accuracy versus loss for 35000 epochs for the MLP with six inputs

TABLE V

PREDICTION RESULTS OF MLP WITH 42 INPUT NODES

True Category	Predicted Category: Positive	Neutral	Negative	Accuracy (%)
Positive	7131	6	20	99.64
Neutral	15	10857	16	99.72
Negative	14	11	8382	99.70

VII. DISCUSSION

This paper explored the use of deep learning models trained on affective data to perform sentiment analysis. The two deep learning models developed for this purpose were similar in architecture and made use of the same data set of affective data. However, they differed in the number of features selected as input to the deep learning model. The first only made use of the six universal emotions proposed by [17], whilst the second model also took facial expressions into account.

The results obtained from the first model, show that the model had difficulty in classifying the given data points labelled as positive. In contrast, the second model performed considerably better in this task, having only a 0.08% difference in accuracy from the neutral class that was classified the most accurately in both cases. Furthermore, the overall accuracy of the model trained on the 42 metrics was also much higher than the model trained on only the six emotions. This suggests that the use of emotions on its own for performing sentiment analysis is insufficient. Rather, the data based on emotions should be augmented with facial expression data to classify sentiment more effectively. Though, it was not in the scope of this study to determine the minimum features needed for performing sentiment analysis.

As mentioned in Section V only a small number of participants with similar backgrounds were used to collect data. Thus, the resulting data set may have contained little variation which could have led to the high accuracies obtained with the second model.

This paper is concluded in the next section, and possible future work is discussed.

VIII. CONCLUSION

Telecommunication enterprises can make use of sentiment analysis to gain a better understanding of their customers' opinions. Since the number of non-text media, such as videos, and audio, being published to the Internet has drastically increased with the use of social media, new modalities of performing sentiment analysis should be explored. Possible improvements can be made to current applications of sentiment analysis by making use of affective data and deep learning neural networks.

This paper proposed the use of affective computing and a deep learning MLP to perform sentiment analysis. To obtain a data set consisting of affective data for performing sentiment analysis, video recordings were made of nine participants while they read three text passages. This resulted in a data set consisting of 132261 points. Two MLPs were developed; the first using only the six universal emotions as input, and the second using all 42 metrics. The latter was created to determine whether it is feasible to include more metrics to improve the accuracy of the first model. The accuracy obtained for the two models were 69.98% and 99.77%, respectively.

The results indicate that using affective data and a deep learning MLP to perform sentiment analysis is feasible. However, it should be further investigated as to which metrics would produce the best results for performing sentiment analysis.

Possible future work includes performing further experiments in terms of the architecture and number of inputs used by the MLP. Also, as only nine participants were used, the number of participants can be increased to create a larger data set that is more representative. Alternative methods to extract other forms of affective data from the videos may also be explored. Furthermore, the problem can be approached using other types of neural networks, such as recurrent neural networks, convolutional neural networks, as well as ensembles of neural networks, as to determine the most suitable type for this problem.

REFERENCES

[1] M. Farhadloo and E. Rolland, "Fundamentals of sentiment analysis and its applications," in Sentiment Analysis and Ontology Engineering: Springer, 2016, pp. 1-24.

[2] E. Afful-Dadzie, S. Nabareseh, Z. KomínkováOplatková, and P. Klímek, "Enterprise Competitive Analysis and Consumer Sentiments on Social Media," in Proceedings of 3rd International Conference on Data Management Technologies and Applications, 2014, pp. 22-32: SCITEPRESS-Science and Technology Publications, Lda.

[3] A. M. Qamar, S. A. Alsuhibany, and S. S. Ahmed, "Sentiment classification of twitter data belonging to Saudi Arabian telecommunication companies," International Journal of Advanced Computer Science and Applications (IJACS), vol. 1, no. 8, pp. 395-401, 2017.

[4] S. Ranjan, S. Sood, and V. Verma, Twitter Sentiment Analysis of Real-Time Customer Experience Feedback for Predicting Growth of Indian Telecom Companies. 2018, pp. 166-174.

[5] M. V. Mäntylä, D. Graziotin, and M. Kuutila, "The evolution of sentiment analysis—A review of research topics, venues, and top-cited papers," Computer Science Review, vol. 27, pp. 16-32, 2018.

[6] S. Poria, E. Cambria, N. Howard, G.-B. Huang, and A. Hussain, "Fusing audio, visual and textual clues for sentiment analysis from multimodal content," Neurocomputing, vol. 174, pp. 50-59, 2016.

[7] S. Poria, E. Cambria, R. Bajpai, and A. Hussain, "A review of affective computing: From unimodal analysis to multimodal fusion," Information Fusion, vol. 37, pp. 98-125, 2017.

[8] R. W. Picard, Affective computing. Cambridge: MIT Press, 1995.

[9] A. Pawar, M. Jawale, and D. Kyatanavar, "Fundamentals of sentiment analysis: concepts and methodology," in Sentiment Analysis and Ontology Engineering: Springer, 2016, pp. 25-48.

[10] M. Soleymani, D. Garcia, B. Jou, B. Schuller, S.-F. Chang, and M. Pantic, "A survey of multimodal sentiment analysis," Image and Vision Computing, vol. 65, pp. 3-14, 2017.

[11] E. Cambria, B. Schuller, Y. Xia, and C. Havasi, "New avenues in opinion mining and sentiment analysis," IEEE Intelligent systems, vol. 28, no. 2, pp. 15-21, 2013.

[12] V. Pérez-Rosas, R. Mihalcea, and L.-P. Morency, "Utterance-level multimodal sentiment analysis," in Proceedings of the

51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2013, vol. 1, pp. 973-982.

[13] J. D. Schwark, "Toward a taxonomy of affective computing," International Journal of Human-Computer Interaction, vol. 31, no. 11, pp. 761-768, 2015.

[14] C. Kleine-Cosack, "Recognition and simulation of emotions," vol. 28, 2008.

[15] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," nature, vol. 521, no. 7553, p. 436, 2015.

[16] D. McDuff, A. Mahmoud, M. Mavadati, M. Amr, J. Turcot, and R. e. Kaliouby, "AFFDEX SDK: a cross-platform real-time multi-face expression recognition toolkit," in Proceedings of the 2016 CHI conference extended abstracts on human factors in computing systems, 2016, pp. 3723-3726: ACM.

[17] P. Ekman and D. Keltner, "Universal facial expressions of emotion," Segerstrale U, P. Molnar P, eds. Nonverbal communication: Where nature meets culture, pp. 27-46, 1997.

[18] L. Deng and D. Yu, "Deep learning: methods and applications," Foundations and Trends® in Signal Processing, vol. 7, no. 3–4, pp. 197-387, 2014.

[19] I. Goodfellow, Y. Bengio, and A. Courville, Deep learning. MIT press, 2016.

[20] H. Ramchoun, M. A. J. Idrissi, Y. Ghanou, and M. Ettaouil, "Multilayer Perceptron: Architecture Optimization and Training," IJIMAI, vol. 4, no. 1, pp. 26-30, 2016.

[21] M. T. Hagan, H. B. Demuth, and M. Beale, "Neural network design," 1997.

[22] O. Chapelle, B. Schölkopf, and A. Zien, Semi-Supervised Learning. Cambridge, UNITED STATES: MIT Press, 2006.

Nicolaas Maree received both his B.Sc. and B.Sc. (Hons) in IT with distinction from the North-West University (NWU) in 2016 and 2017, respectively. He is currently pursuing his M.Sc. in Computer Science whilst acting as a junior lecturer in the NWU's School of Computer Science and Information Systems.

The authors gratefully acknowledge the financial support of this study by the Telkom CoE at the NWU and the National Research Foundation under grant nr TP14081892668

Annexure E: Confirmation of language editing

This serves to confirm that I, Isabella Johanna Swart, registered with and accredited as professional translator by the South African Translators' Institute, registration number 1001128, language edited the following dissertation:

Active computing and deep learning to perform sentiment analysis

by

NJ Maree

Dr Isabel J Swart

Date: 26 June 2020

23 Poinsettia Close
Van der Stel Park
Dormehlsdrift
GEORGE
6529
Tel: (044) 873 0111
Cell: 082 718 4210
e-mail: isaswart@telkomsa.net