Development of framework for facial expression analysis using representation learning

A Thesis

submitted in fulfillment of the requirements for the award of the degree of

DOCTOR OF PHILOSOPHY

in

Electronic and Communication Engineering

By

Vivek Singh

Regn. No.: 901506023

Under the supervision of

Dr. Vinay Kumar

Associate Professor

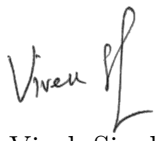
Department of Electronics and Communication Engineering,
Thapar Institute of Engineering & Technology,
(Deemed to be University),
Patiala (147004), India.

December 2019

Declaration

I, **Vivek Singh**, hereby certify that the work, which is being presented in the thesis, entitled "**Development of framework for facial expression analysis using representation learning**" submitted in Thapar Institute of Engineering and Technology, Patiala in partial fulfillment of the requirements for the award of the degree of Doctor of Philosophy in Electronics and Communication Engineering, is an authentic record of my own research work carried out during the period Jan 2016 to May 2019 under the supervision of **Dr. Vinay Kumar**.

I have also cited the reference about the text(s)/figure(s)/table(s) from where they have been taken. The matter presented in this thesis has not been submitted elsewhere for the award of any other degree or diploma from any institution.



Date: December 6, 2019

Vivek Singh

Candidate

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Date: December 6, 2019

Dr. Vinay Kumar

Associate Professor

Supervisor

Abstract

Facial expressions play a crucial role in human social interaction; and this is the primary component needed to be integrated into machines to make human computer interaction more user friendly. Although, humans are very efficient in recognizing even the minute changes in facial expression but for machines it is a very complex task. Recently, this area of research has attracted much needed attention due to its broad spectrum of application. However, expression analysis in an unconstrained environment is a very difficult task. Variations in illumination, facial features, head pose and changes in background make it very difficult to correctly recognize emotions in an open setup for commercial applications.

This thesis develops deep learning based representation learning methods for analyzing facial expressions. In this work, multiple frameworks are developed for different applications of facial expression analysis. First proposed framework analyzes the emotional sentiment represented by an image based on its content. The proposed system investigates the faces and background in the image, and extracts facial and scene features from them, respectively. Two different convolutional neural networks are used to extract facial and scene features. Conditional occurrence of these features is modeled using long short term memory networks to predict the sentiment represented by the image.

The second framework, proposed in this work, predicts likability of the multimedia content based on the facial expression of the viewer. A database with two different sets of video samples was collected for the task under unconstrained environment. First set of samples consists of videos to be watched by recruited subjects called as stimulants. Second set of samples are recordings of facial expressions of subjects while watching stimulants. The proposed framework is a multimodal system which learns spatio-temporal features from the videos of subject to predict the likability. Combination of '3D convolutional neural network' and 'convolutional neural network - long short term memory network' models are used to extract the features from

spatial and temporal variations of face. Another model (designed using long short term memory networks) was used to analyze motion of facial landmarks in the videos of subjects.

Lastly, an activation function (Linearized Sigmoidal Activation) is proposed to improve the learning capacity of the deep learning models. Modeling non-linear dependencies in the data is very complex task. Higher nonlinearity activation function tend to have the problem of vanishing and exploding gradients. While, linear activation functions, like ReLU, do not have enough learning capacity. The proposed activation function have segment wise nonlinear behavior. It has linear characteristics inside a given segment, whereas nonlinear relationship exists among the segments.

Acknowledgment

The research work related to this thesis has been carried out at Department of Electronics and Communication Engineering, Thapar Institute of Engineering and Technology, Patiala, India, during the years 2016-2018. Without the guidance, inspiration, and support of a number of people, this thesis and my PhD journey would not have been possible. So it would be my pleasure to express my appreciation and gratitude to everyone who has been a part of this journey to make it possible.

First and foremost, I would like to express my gratitude to my principal supervisor, Associate Professor Dr. Vinay Kumar for the opportunity to carry out this research and for his invaluable guidance, support, encouragement, and life teachings. He has been with me since the beginning of my PhD and has been through my ups and downs in my PhD life. Without his help and patience I will not be able to complete my PhD study.

This work could not have been possible without the financial support offered by Department of Electronics and Communication Engineering. I would like to thank Head of the Department, Professor Dr. Alpana Aggarwal for funding my work and for support. My special thanks to all the members of the my doctoral committee, Dr. Shailni Bhatra, Dr. Sanjay Kumar, Dr. Amit Mishra. Their continuous feedback on the progress of my work helped me maintain the high quality of research work.

Finally and most importantly, my deep appreciation goes to my family and all the friends. Their encouragement, moral support, understanding, constant love, and prayer become my inspiration, and this thesis is my gift to all of you.

Contents

List of Figures

List of Tables

Nomenclature

ALiSA Adaptive Linearized Sigmoidal Activation

AU Action Unit

BOW Bag Of Words

CNN Convolutional Neural Networks

DCNA Deep Composite Neural Architecture

ELU Exponential Linear Unit

FACS Facial Action Coding System

FAST Features from Accelerated Segment Test

FER Facial Expression Recognition

GPU Graphical Processing Units

HCI Human Computer Interaction

HOG Histogram of Oriented Gradients

LBP Local Binary Patterns

Leaky ReLU Leaky Rectified Linear Unit

LFDA Local Fisher Discriminant Analysis

LiSA Linearized Sigmoidal Activation

LPQ Local Phase Quantization

LSTM Long Short Term Memory

MLP Multi Layer Perceptron

PHOG Pyramid Histogram of Oriented Gradients

PReLU Parametric Rectified Linear Unit

ReLU Rectified Linear Unit

RNN Recurrent Neural Network

SELU Scaled Exponential Linear Unit

SIFT Scale Invariant Feature Transform

SURF Speeded Up Robust Features

SVM Support Vector Machine

1 Introduction

1.1 Overview

With the recent advancements in the technology, humans have started encounter machines at different levels of their lives. This interaction with technology is on rise and have become integral part of our daily routine. Interestingly, many people end up spending more time interacting with machine than with their human counterparts. No one can avoid this evolution of technology and have to adapt to this new era of technology. In spite of such a deep Human Computer Interaction (HCI) at every level of society, machines have not been able to process the emotional part of the message restricting effectual communication between the two. The competence of such a communication relies on the ability of the machine to perceive and interpret emotional state of user. Additionally, the state of mind of the person or the intention of communication is being given is not specified clearly through a speech signal or an input text message to the computer. Presently, non-verbal information does not have any significant role in HCI based systems.

Both verbal and non-verbal communication is vital for our everyday interaction and nurtures interpersonal relationships. Non-verbal communication [10] includes non-verbal queues; like, gestures, facial expressions, eye contact and posture etc. Facial expression is most convincing form of non-verbal communication, as it provides most accurate information about affective state, temperament or intention of the message [11]. Apart from molding the nature of conversation, facial expressions are an ingenuous method of communicating generous amount of information with only a simple facial gesture [12]. [13] explains that while decoding information, if the facial expressions are not in accordance with the verbal speech, then facial expression information takes lead over vocal communication.

The emerging trends in computing environment are leaning progressively towards communicating substantial amount of information through affective states or expres-

sions by contriving human-centered designs instead of computer centered designs [14, 15, 16]. A tremendous amount of information divulged through those affective states is usually ignored by traditional HCI systems that only convey user's intentional input. With this paradigm shift, humans will interact with machines not only through the intentional inputs but also through their manner of conduct i.e. affective states [17]. Therefore, automatic facial expression recognition has emerged as a leading field of research within computer vision research community in last decade.

Facial Expression Recognition (FER) finds applications in a wide range of areas; like, human computer interaction, entertainment, deceit detection, behavior monitoring, medical treatment etc. The recognition of facial expressions, one of the most important forms of non-verbal communication, is not new to the evolutionary journey of human evolution. Ever since the dawn of civilization, humans are able to understand various facial expressions everyday without any extra effort. Many great philosophers and thinkers like Aristotle and Stewart studied facial expressions to understand human behavior. Darwin extended this study of facial expressions to an empirical realm through his acclaimed theory of evolution. Ekman in 1971 [18], defined facial expression recognition for computer vision as the task of classifying input facial features in one of the six basic or universal emotions: happiness, sadness, fear, disgust, surprise and anger. Recognition of these facial expressions by computes is still a very tedious task due to change in facial appearance caused by pose variations, illumination variations, camera quality and angle changes.

1.2 Applications

Facial expression analysis has applications in a wide spectrum. Some areas that benefit from the analysis of affective state of user are:

1. Psychological disorder diagnosis and treatment: Facial Action Coding System (FACS) are able to recognize minor muscle movement, which get ignored by human eye. FACS has been in use for diagnosis and treatment of depression, schizophrenia, and other psychopathological disorders [19]. FACS based expression analysis system is also used to identify alcohol related intoxication [20].
2. Pain monitoring for patients: Pain monitoring is a very critical task and needs a manual handing currently. Lack of health personals requires this task to be

automated with FER based system. [21] reports that facial expressions can be used to measure the level of pain.

3. Commercial application: FER also holds a great potential in commercial applications. For example, expression analysis system can be installed at marts to get an automated feedback on customer experience. It can also be used to test the user response to advertisements, movies and pilot TV series, etc [22].
4. Smart homes: Smart homes find a broad range of applications related to FER. [23] shows some interesting application of FER for human centered systems. FER system can be used to control type of music, lighting condition and room temperature etc. through expression observation.
5. Robotics: Expression analysis can be useful in producing robots with better interactive skills. Consumer robots with ability to analyze expression can be highly effective in communication and interaction [17], as facial expression are most important type non-verbal communication.
6. Realistic gaming: Gaming companies are trying to enhance user experience by providing more realistic virtual environment and highly human like avatars. The industry can infuse facial expression recognition to provide more realistic avatars for their customers. Facial expression can be recorded using the webcam and later can be analyzed to provide various other features; like, difficulty control, game music etc.
7. EmotiChat: An interesting application of the facial expression analysis is in chatting applications. [24] presented a framework which automatically analyzes expressions of user and generates corresponding emoticon.

1.3 Challenges

There has been a significant amount of research on FER and various methods has been proposed. But there are still some inherent problems that need to be addressed by the research community. Following are some of the challenges that need attention:

1. Although methods for automatic facial expression recognition produce good results on different datasets, but they fail drastically for real-time applications. It happens because apart from being computationally expensive they require high dimensional feature vector to complete the task.

2. There is a need for novel methods that uses low resolution images as input which can adequately support many real world applications such as smart meeting, video conferencing and visual surveillance etc. based on facial expression recognition system.

3. Out of several facial expression recognition methods, very few provide good results on low resolution images.

4. Apart from recognizing the six classical emotions, there is a dire need of new methods to recognize more complex facial expressions (such as fatigue, pain, and mental states, such as, agreeing, disagreeing, lie, frustration, thinking). This may open ways for realizing numerous novel application in Human Computer Interaction.

5. Most of the facial expression analysis datasets are developed in controlled environment. Models designed on these datasets fail to provide any significant result in real world scenarios.

6. Several other challanges such as expression intensity estimation, spontaneous expression recognition, micro expression recognition (brief involuntary facial expression that lasts only 1/25 of a second), mis-alignment problem, illumination, and face pose variation etc. need attention from research community.

1.4 Objectives of the dissertation

This thesis work was conducted with main focus on following three objectives:

1. To study, explore and identify various available techniques, architectures and frameworks for facial expression analysis.

2. Development of the framework for facial expression analysis for understanding emotion using representation learning.

3. To evaluate the proposed framework for facial expression analysis using performance parameters; like, classification accuracy, computation cost, time per frame etc.

1.5 Structure of the dissertation

Chapter 1 provides overview of affective computing, and presents the application and challenges related to the field. It also presents the contributions of the thesis and list of publications produced under this research work.

Chapter 2 discusses the basics emotions, their interpretation and types. It also discusses various approaches available to extract the emotion from raw facial image. Basic deep learning architectures like convolutional neural network and recurrent neural networks are also presented in this chapter.

Chapter 3 provides the state of the art literature available to the various problems studied in this thesis. This chapter provides the literature on different approaches used to study affective state and application of the affective state in various commercial and social applications. We also present recent developments on the architectural aspects of the deep learning algorithms.

Chapter 4 presents our work on emotional sentiment analysis represented by the image. This chapter provides the implementation and the comparative analysis of the proposed framework against the state of the art methods.

Chapter 6 presents a novel activation function which was design during this thesis work to boost the representation capability of the deep learning algorithms. Proposed activation function help convolutional neural networks in learning the better feature representations for correct classification of emotions. Proposed function is evaluated on other tasks in addition to facial expression analysis.

Chapter 5 presents a novel framework designed in this period for prediction of the likability of any multimedia video based on the facial expressions of the viewer. This framework is a multimodal system and each component of the system is evaluated against each other to present the clearer contribution of each submodule in the final accuracy of the system.

Chapter 7 presents the final conclusion of the thesis. This chapter discuss the contribution of this thesis and future scope for the further research.

1.6 Contributions

The contributions of this dissertation are as follows:

1. A novel system is developed to label the sentiments represented by an image into three basic emotions: positive, neutral and negative. A multimodal system is developed with the combination of global and local features. Global features incorporate low dimensional representation of the scene presented in the image while the local features are extracted from the faces of the people present in image. Representational learning approach based on deep convolutional neural networks is used to learn both the feature representations.
2. It also presets a novel activation function to model non-linear spatial dependencies in data such as speech, images and videos. The proposed activation function is divided into multiple segments, each having linear behavior within its range, but have non-linear relationship amongst all the segments. This helps in increasing learning capacity of deep learning model. Multiple experiments are conducted to show the achievement of higher performance on tasks, like facial emotion recognition, by proposed function.
3. A novel video based framework is developed to automatically produce user feedback based on his/her expressions against the video being watched by subject. A database is created by recruiting the subjects to watch the preselected set of videos and provided their response against each of them. A multimodal system is developed, which uses the geometric as well as the appearance information from the subject's face, to predict the label for the shown multimedia content.

1.7 Publications

- V. S. Bawa and V. Kumar, "Emotional sentiment analysis for a group of people based on transfer learning with a multi-modal system," Neural Computing and Applications, pp. 1-12, 2018.
- V. S. Bawa and V. Kumar, "Linearized sigmoidal activation: A novel activation function with tractable non-linear characteristics to boost representation capability," Expert Systems with Applications, vol. 120, pp. 346-356, 2019.
- V. S. Bawa and V. Kumar, "An automatic multimedia likability prediction system based on facial expressions of the observer," Under review.

2 Basics of affective computing

2.1 Emotion and its interpretation

Humans realized importance emotion and facial expressions in daily social interaction, and since then, their analysis and utilization for different applications has become eminent area of research. According to some researchers, over the course of evolution many mammals, including humans, developed emotions as an evolutionary process to support their vital life tasks by adapting to the environmental and psychological changes around them [1].

The way we behave, make decisions and communicate with others are influenced by their affective states and facial expressions, which are an important part of human life [25]. Individual's actions are influenced by his/her affective state and the affective states of people around them. Unfortunately, psychologists themselves have not reached to a consensus on the definition of emotion and affect. However, some of the most acclaimed approaches to define it in psychological research are based on discrete categories and dimensional representations [26, 27, 28]. All these theories provide information about the expression of an affect and its interpretation. Therefore, they form a reliable premise to understand the idea of affect for the purpose of automatic affect recognition.

2.1.1 Emotion categories

According to Paul Ekman, emotions can be segregated in terms of discrete categories [1]. He assorts various emotions in six categories: happiness, sadness, surprise, fear, anger and disgust (shown in fig 2.1). He also suggested that all these emotions evolved in the similar manner for the whole mankind so that their perception and interpretation stay similar.

Figure 2.1: Seven basic classes of emotions: happiness, sadness, fear, anger, surprise, disgust and neutral (left to right) [1].

More emotion categories are introduced in the later works; such as, boredom, contempt and engagement etc. [29, 30]. [2] extended basic emotions further into compound emotions, which are combination of basic six emotions. Compound emotions show signs of two of more basic emotions resulting in a new affective state of subject (shown in fig. 2.2). [31] studied facial expressions corresponding to the emotions like amusement, pride and relief in addition to basic emotions.

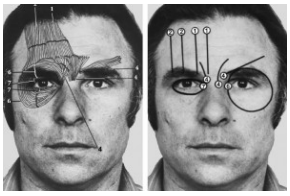
Figure 2.2: Facial expression corresponding to compound emotions: Happily surprised, happily disgusted, sadly fearful, sadly angry, sadly surprised, sadly disgusted, fearfully angry, fearfully surprised [2].

Many cross-cultural studies suggest that regardless of culture, the facial expressions of these prime emotions are perceived in the same way. Paul Ekman carried out ground breaking research in the field of facial expression recognition in his landmark papers based on basic emotions and the Facial Action Coding System (FACS) [32].

2.1.2 Facial action coding system

Facial Action Coding System (FACS) is developed by Paul Ekman to taxonomize the movement of facial muscle into different categories of expression and emotions. FACS map the facial muscle movement into different categories (refer fig. 2.3a). These categories are called as Action Units (AUs). Some examples of the action units are shown in fig. 2.3b. Each action unit corresponds to a specific state of a single facial muscle region. For example, AU1 represent the rise of inner eye brow while AU2

represents the rise of outer eye brow. Combination of these action units are used to identity the emotion represented by the face. FACS is most widely used system for the analysis of facial expressions to date [33]. This system finds many application, including micro expression analysis for lie detection, psychological disorder diagnosis etc.



(a) The image represents the facial muscle movement corresponding to the experienced emotion.

Upper Face Action Units					
AU 1	AU 2	AU 4	AU 5	AU 6	AU 7
Inner Brow Raiser	Outer Brow Raiser	Brow Lowerer	Upper Lid Raiser	Cheek Raiser	Lid Tightener
*AU 41	*AU 42	*AU 43	AU 44	AU 45	AU 46
Lid Droop	Slit	Eyes Closed	Squint	Blink	Wink
Lower Face Action Units					
AU 9	AU 10	AU 11	AU 12	AU 13	AU 14
Nose Wrinkler	Upper Lip Raiser	Nasolabial Deepener	Lip Corner Puller	Cheek Puffer	Dimpler
AU 15	AU 16	AU 17	AU 18	AU 20	AU 22
Lip Corner Depressor	Lower Lip Depressor	Chin Raiser	Lip Puckerer	Lip Stretcher	Lip Funneler
AU 23	AU 24	*AU 25	*AU 26	*AU 27	AU 28
Lip Tightener	Lip Pressor	Lips Part	Jaw Drop	Mouth Stretch	Lip Suck

(b) Different action unit corresponding to different facial muscle and their movement.

Figure 2.3: Some examples of the action units used in facial action coding system.

2.2 Facial expression recognition system

Block diagram of basic facial expression analysis system is shown in fig 2.4. A FER system contains five steps. First, an image is preprocessed to remove any

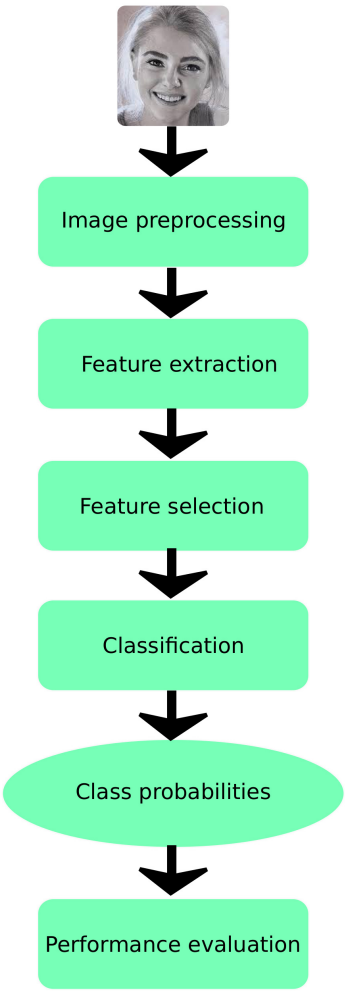


Figure 2.4: Block diagram of basic facial expression recognition system.

type of noise or artifact. Preprocessing is very important due to the variations in illumination, head pose and camera parameters. Preprocessing generally includes contrast stretching, image smoothing, image normalization and face alignment (to correct the tilt in face angle).

Feature extraction is performed to extract the useful structural components from the image to identify the emotion. Feature extraction techniques (used in FER system) are of two types: geometry based features and appearance based feature. Feature extraction is discussed in detail in Section 2.3. In feature selection process, only the informative features are kept. These selected features are passed to classifier, which classifies them on the basis of their distribution. Accuracy of the model depends on

the combination of feature extraction algorithm and classifier. If feature extraction algorithm is incapable of producing distinctive features for different classes, classifier will fail as well. Similarly, if we use a weak classifier (e.g. linear boundary based), even if our feature extraction algorithm is good, classifier would not be able to provide good accuracy, unless those features are linearly separable.

Mostly used classifiers provide class probabilities as output. These class probabilities of the validation data are used to evaluate the performance of designed system. Validation set consists of the data samples that are not used for training the models. These samples are used as universal set to test the performance of model. It is very important that validation set is selected randomly from the available sample set otherwise the evaluated performance of the model will be biased.

2.3 Feature extraction methods

Feature extraction algorithms reduce the dimensionality of very high dimensional data, and make it more manageable and interpretable for the task in hand. Feature extraction methods are divided into two categories based on the approach used by them to extract features. These categories are discussed in the following subsections.

2.3.1 Hand crafted feature extraction approaches

These algorithms are designed by the human based on the application. The hand crafted feature extraction algorithms are designed to discard irrelevant information from the data for given task. In other words, feature extraction algorithms transform high dimensional input data into low dimensional data without loosing relevant information. Hand crafted feature extraction methods can be further divided into two types:

- **Appearance based methods**: The appearance based features use texture, edge information and orientations of edges to generate a feature description about the image. These features do not consider physical structure of the input image (geometry). Most common appearance based feature extraction methods are Gabor descriptor [34], Histogram of Oriented Gradients (HOG) [35], Local Binary Patterns [36] and Scale Invariant Feature Transform (SIFT)[37] etc. Most commonly used methods (like, HOG and SIFT) are based on the

edge orientation features of the image. Orientation based methods are more robust than the texture based methods because of their immunity to the variation in lighting condition and other physical characteristics of subject, like color.

- **Geometry based methods**: Geometry based features extract the geometric characteristics of the data, like placement of ear, nose, eyes etc. Most common approach for extracting the geometry of the face is facial landmark detection [38, 39]. Geometry based methods are not considered reliable due to difficulty in correct detection of facial landmarks. Additionally, developing a robust model, based on the correspondence between these landmarks, is very difficult.

2.3.2 Feature learning approaches

Feature learning methods (also called as representation learning) involve deep learning (neural network) based algorithms. Advantage of the feature learning methods is that we do not need to design a feature extraction algorithm for every new task. Secondly, accuracy of hand crafted feature extraction algorithms depends on the quality of the technique developed by the developer. Whereas, feature learning algorithms are general purpose frameworks which extract relevant features from the provided data and labels. Formally, under supervised setup (when data and target labels are given), if X is the input data and Y is the output of the system. Then, learning algorithm maps the input data to the output labels, represented as:

$$f(X) \rightarrow Y \tag{2.1}$$

The task of the learning algorithm is to learn the function (f) which maps the input data to the given output. In the case of affective computing, X can be interpreted as input facial features or raw pixel values, and Y could be the emotion category for classification model or intensity of emotion for regressive model. From probabilistic point of view, a learning problem is referred to as finding correct distribution of input data corresponding to the given set of labels:

$$Y = P(X \mid \theta) \; \forall \; X \in \mathbb{R}^N \tag{2.2}$$

The eq. 2.2 represents the probabilistic interpretation of a learning system. Here, θ is the model parameters. These parameters are learned during the training to represent the correct relation between the input (X) and output (Y).

2.4 FER databases

There are mainly two type of databases available on affective computing: static image based and video sequence based. One of the most famous dataset based on image is CK+ also known as Extended Cohn-Kanade dataset [40]. This dataset contains 8 different emotions labeled as: neutral, sadness, surprise, happiness, fear, anger, contempt and disgust. Dataset contains 640×490 size gray scale images with posed expressions captured in controlled environment. Japanese Female Facial Expressions (JAFFE) [41] dataset contains images of Japanese female faces. Dataset contains all the basic emotion classes and is developed in controlled environment. AffectNet [42] is the most recent dataset developed for affective state analysis. It contains 450,000 manually annotated and 500,000 automatically annotated samples. The dataset is developed in wild setup to present the challenges faced in real time facial expression analysis.

Belfast database [43] is a famous dataset in video sequence based emotion analysis. Dataset is divided into three sets and contain video of frame sizes 720×576 and 1920×1080. In total, dataset contains 1400 samples recorded in natural environment. Multimedia Understanding Group (MUG) [44] dataset contains 7 basic emotions. The dataset has 1462 video samples of size 896×896 captured at 19 fps. Annotated Facial Expression in Wild (AFEW) [45] database contains small clips cropped from different movies. Dataset contains all basic 7 emotion classes and 1426 video sequences.

In present thesis, we use Static Facial Expression in Wild (SFEW) [45], which contains statics images extracted from movies. This dataset was developed by extracting single frame from the movie clips collected for AFEW dataset. Dataset has wild setup because movie scene have random and uncontrolled backgrounds. We also used Group based Emotion Recognition (GER) dataset [46]. This dataset contains images with groups of people. Our target is to identify emotion represented by group of people present in image. Dataset has three classes: Positive, Negative and Neutral.

2.5 Basic of deep learning

Deep learning algorithm have been deployed in wide range of application with very successful results. Hence, affective computing researcher has also used different variants of these models in various research problems. This section provides insight into various successful deep learning architectures in affect analysis.

2.5.1 Convolutional Neural Networks

Convolutional Neural Networks (CNN) [47] are a special type of neural network which are very efficient in learning features from data with fixed spatial structural geometry. CNNs use two dimensional kernels for convolution with smaller local set of data points. Fig. 2.5 depicts the basic working principle of the CNN. In a CNN architecture, input image (I)[1] is convolved with kernel (K) to produce the output feature map (F). Mathematically this operation can be represented as:

$$F(i,j) = \sum_m \sum_n I(i+m, j+n) \times K(m,n) \tag{2.3}$$

where, i and j are coordinates of output feature map and input image, and m and n represent size of the kernel for given layer.

Each layer has multiple number of kernels convolving with same set of feature maps; i.e., for p number of kernels denoted by K_l^p, output feature map F_l can be represented as eq. 2.4.

$$F_l^p(i,j) = \sum_m \sum_n \sum_d F_{l-1}(i+m, j+n, d) \times K_l^p(m,n,d) \tag{2.4}$$

where, parameter l represents layer number in the model, and d is number of feature maps produced by previous layer (input depth of current layer).

The output of the convolution layer (F_l^p) is then passed through a non-linearity also called as activation function. Rectified Linear Unit (ReLU) [7] is the most

[1]We are assuming input dataset to be consisted of images only.

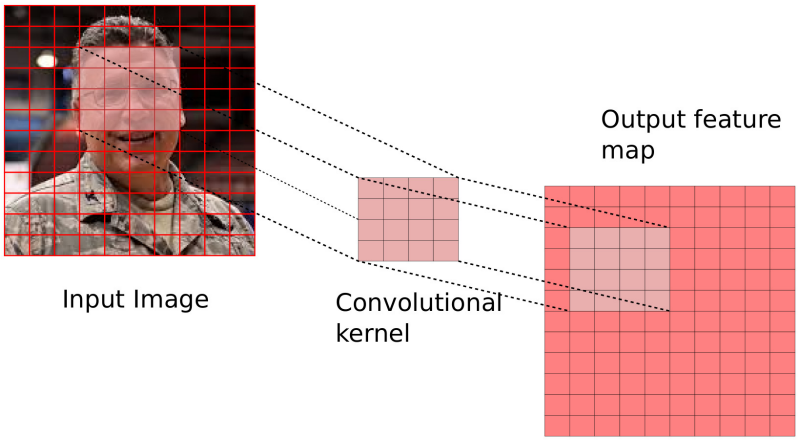


Figure 2.5: Basic working principle of convolutional neural networks.

commonly used type of non-linearity. The output of the activation function (O_l^p) can be represented as below:

$$O_l^p\left(i,j\right)=\alpha\left(\sum_m\sum_n\sum_d F_{l-1}\left(i+m,j+n,d\right)\times K_l^p\left(m,n,d\right)\right) \tag{2.5}$$

where α represents the activation function.

Most commonly used activation functions are:

1. ReLU activation function:

$$\alpha(x)=\begin{cases}0 & \text{if } x<0\\ x & \text{if } x\geq 0\end{cases} \tag{2.6}$$

2. Sigmoid activation function:

$$\sigma(x)=\frac{1}{1+e^{-x}} \tag{2.7}$$

3. Hypertangent activation function:

$$tanh(x)=\frac{e^x-e^{-x}}{e^x+e^{-x}} \tag{2.8}$$

Pooling is also one of the core concepts of the CNN architecture. Pooling layer is used

for reduction of the feature vector, which leads to compressed feature representation corresponding to a very high dimensional data. There are different types of pooling operations, e.g. average pooling, max pooling, global average pooling etc.

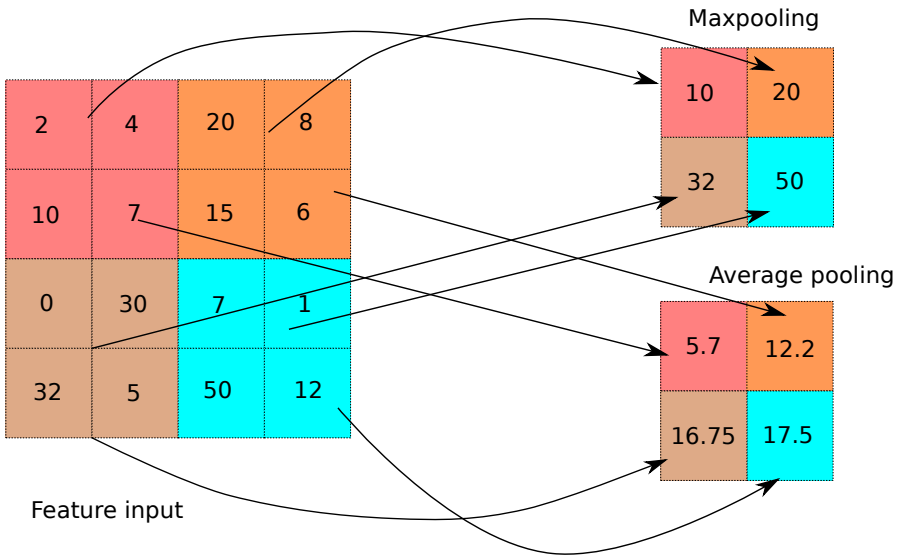


Figure 2.6: Pooling operations used in CNN models, two examples are depicted here.

If the size of a pooling kernel is $(k \times k)$, then that kernel produces single output corresponding to $(k \times k)$ features as shown in fig. 2.6. Hence, it will be able to provide the feature compression ratio of $1 : k^2$. As shown in fig. 2.6, different type of operation can be used for pooling of features. Most common pooling operation are max pooling and average pooling. The max pooling operation reflect only the feature with highest activation values while all other features are discarded. Intuition behind this operation is that only the strong activity are the most important for correct classification. Additionally, this operation helps in extracting robust and invariant features. Similarly, if average pooling is used, it generate the average value for each of the pooling region. Hence, each feature has equal contribution in the output map.

2.5.2 Recurrent Neural Networks

Recurrent Neural Network (RNN) [47] are another special type of neural networks. These networks are good at analyzing the conditional distribution of various input parameters. A simple RNN architecture is shown in fig. 2.7. x_t, x_{t+1}, $x_{t+2} \ldots$

represent the inputs, H_t, H_{t+1}, $H_{t+2}\ldots$ are the hidden states of recurrent cell, and y_t, y_{t+1}, $y_{t+2}\ldots$ are the their corresponding outputs at time $t, t+1, t+2\ldots$, respectively.

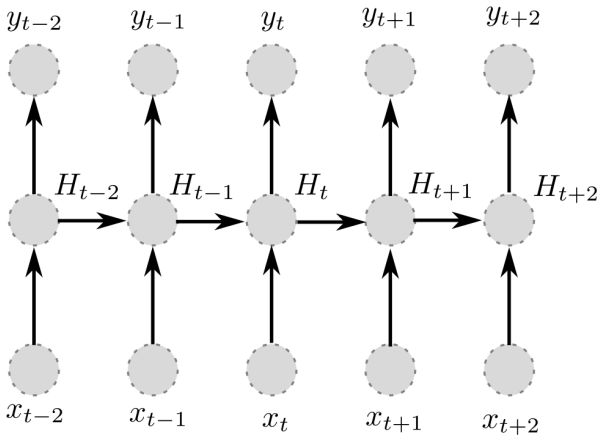


Figure 2.7: RNN based architecture for showing relation between input and output of the model at time $\ldots, t-1, t, t+1, \ldots$.

Probability of receiving an output vector, given the input vectors, is represented by following equation:

$$P(y_1, y_2 \ldots y_{T'} | x_1, x_2 \ldots x_T) = \prod_{t=1}^{T} P(y_t | x_1, x_2 \ldots x_T) \tag{2.9}$$

where $y_1, y_2 \ldots y_{T'}$ represent output vector and $x_1, x_2 \ldots x_T$ represent input vector for RNN with sequences length of T' and T, respectively. Physical implementation of the probability estimation using RNN is shown in fig. 2.7. At the time of implementation, eq. 2.9 is represented as:

$$y_t = f_\alpha(H_t, x_t; \theta) \quad \forall \quad t = 1, 2, 3 \ldots T \tag{2.10}$$

$$H_t = f_\beta(H_{t-1}, x_t; \theta) \quad \forall \quad t = 1, 2, 3 \ldots T \tag{2.11}$$

$$H_{t-1} = f_\gamma(H_{t-2}, x_{t-1}; \theta) \quad \forall \quad t = 1, 2, 3 \ldots T \tag{2.12}$$

where, y_t is output of the model at time t. This output is function of current hidden state (H_t) and current input vector (x_t) when the model parameters (θ) are given.

A hidden state is represented as a function of previous hidden state (H_{t-1}) and corresponding input (x_t) (refer fig. 2.7). From eqs. 2.10, 2.11 and 2.12; the output can be represented as a function of all previous inputs (eq. 2.13).

$$y_t = f_\alpha(f_\beta(f_\gamma(\ldots(\ldots x_{t-2}, x_{t-1}, x_t; \theta)))) \quad \forall \quad t = 1, 2, 3 \ldots T \tag{2.13}$$

Long short term memory networks

Long Short Term Memory (LSTM) network is one of the most commonly used RNN architectures. This architecture was proposed by Hochreiter et al. in 1997 [48]. But the successful use of these networks came almost a decade later. Alex Graves popularized these networks by successfully using them on hand writing recognition [49] and speech recognition [50]. LSTM concur the most fundamental problems of RNNs which is their inability to learn long term dependencies. In vanilla RNNs, gradient gets over amplified or compressed to vanishing point as the length of the input sequence increases. The gated structure of LSTM assists in controlling the flow of gradient in very long sequences. Hence, These networks, unlike vanilla RNN, do not face the problem of vanishing or exploding gradients in case of data with long term dependencies [47].

One memory element of LSTM network is shown in fig 2.8. LSTMs are slightly different in architecture and operation than common recurrent neural networks; but, their performance is much better due to their ability to exploit long term temporal dependencies. LSTM consists of multiple gates namely input gate (I_t), output gate (O_t), forget Gate (F_t) which control flow to and from the memory cell (C_t). These gates are used to control the output and internal state of memory cell.

In fig 2.8, connectivity of input data (X_t) to the state of memory cell is controlled by input gate (I_t) and state of input gate is decided by the mechanism given in eq.

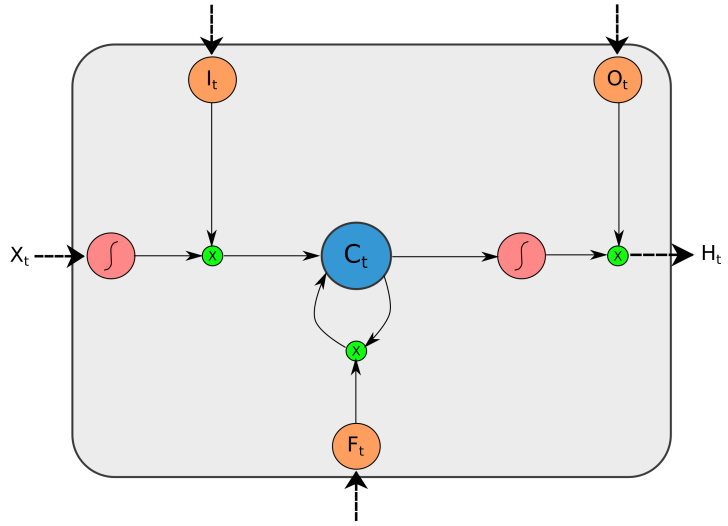


Figure 2.8: Single elementary block of Long Short Term Memory (LSTM) network.

2.14. Input gate acts as switch, controlled by output of cell (H_t) and input value (X_t).

$$I_t = \sigma(W_i X_t + U_i H_t + b_i) \tag{2.14}$$

where, H_t is the output of cell and W_i, U_i and b_i are trainable model parameters.

When input gate is switched on, input gets through the gate and provides an auxiliary state ($\tilde{C}_t$) to the cell. The auxiliary state is given by eq. 2.15 (this is not the final state of cell). Here, W_c, U_c and b_c are trainable model parameters.

$$\tilde{C}_t = tanh(W_c X_t + U_c H_{t-1} + b_c) \tag{2.15}$$

If input gate (I_t) and forget gate (F_t) are closed, model maintains cell state and keep looping at every new iteration. Forget gate (F_t) is used to erase the cell state when information becomes irrelevant. The erasing mechanism is given by eq. 2.16, where W_f, U_f and b_f are trainable model parameters. Sigmoidal (σ) non linearity is used to control the state of this gate. Actual state of the cell is calculated using

state of I_t and F_t, refer eq. 2.17.

$$F_t = \sigma(W_f X_t + U_f H_{t-1} + b_f) \tag{2.16}$$

$$C_t = I_t \times \tilde{C}_t + F_t \times C_{t-1} \tag{2.17}$$

The output gate (O_t) is used to control the output of each memory cell, which in turn affects the contribution of each cell at each iteration. The state of O_t is calculated using eq 2.18. O_t is used to calculate final output of the model (H_t), refer eq. 2.19.

$$O_t = \sigma(W_o X_t + U_o H_{t-1} + V_o C_t + b_o) \tag{2.18}$$

$$H_t = O_t \times tanh(C_t) \tag{2.19}$$

3 Literature review

Even after many decades of research, efficient and accurate detection of human facial expression has not been achieved. Fully automated facial expression and emotion analysis systems find applications in numerous fields [51, 52, 53]. The relevant literature is discussed in this chapter on expression analysis technique.

3.1 Basic affective computing approaches

Different approaches to design a facial expression analysis system are discussed in this section. These approaches can be divided into three fundamental categories: statics (image) based, dynamics (video) based and multi-modal.

3.1.1 Static image based methods

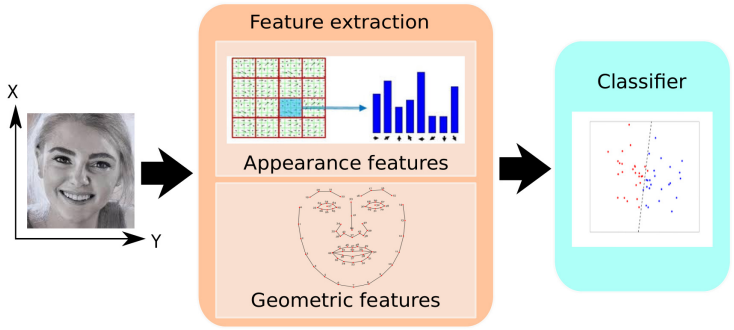


Figure 3.1: Block diagram of static image based FER system.

These approaches use single image to infer the represented emotion. The basic building blocks of static methods are shown in fig. 3.1. Static methods use spatial feature (discussed in Section 2.3) to generate a low dimensional feature vector. This

feature vector comprises hand crafted or learned information necessary for correctly classifying the image. Image based methods generally have inferior performance due to their reliance on just current state of the facial expressions. Hence, they are not suitable in identification of complex and hybrid emotions. Yet their low computational requirement makes them a suitable alternative for identification of basic emotions in commercial applications [54, 55].

[56] showed that facial landmarks have Grassmannian properties in affine shape space. They also demonstrated that Grassmannian manifold can be used to represent all the facial deformations created by facial expressions. This technique removes the effect of variation in camera coordinate system on geometric features, and helps in dealing with facial alignment issues.

[57] explored the Bag of Words (BOW) based approaches for facial expression analysis. They used the combination of spatial pyramid framework, augmented by multiscale dense SIFT features, locality-constrained linear coding and max-pooling features in BOWs. The authors states that combination of these features is capable of improving the classification accuracy for a linear classifier as well.

[58] developed a Local Fisher Discriminant Analysis (LFDA) based method to recognize facial expressions in encrypted domain. According to authors, this methods has the benefit that expression recognition can be performed with images in their encrypted form without revealing any content.

[59] presented a patch based approach for facial expression analysis. The patches are extracted from prominent facial areas based on location of the facial landmarks. Further processing on these extracted patches provides the saliency patches, which contribute to the correct classification of expressions, rest of the patches are discarded. Patch based approach reduces the computation cost and improves classification accuracy for low resolution images.

Yu et al. [45] developed a ensemble based model to categorize the facial emotions in SFEW dataset [45]. SFEW dataset contains images from the movie scenes. Task requires the classification of the facial emotion of the main subject into seven basic emotional categories. The paper uses ensemble of CNNs with different architecture to classify the emotion. All the models are pretrained on the large Facial Expression Recognition (FER-2013) dataset [60]. Hinge loss and log likelihood are used as loss functions for optimization of the models. Output of the models in ensemble is averaged to get the final prediction.

[61] developed an images based system to assess the level of pain for the patient under observation. The paper presents a novel feature descriptor for assessing the pain from facial activity of the patient. The feature descriptor uses combination of both geometric and appearance based features to accurately model the facial activity in reference to level of pain.

3.1.2 Dynamics (video) based methods

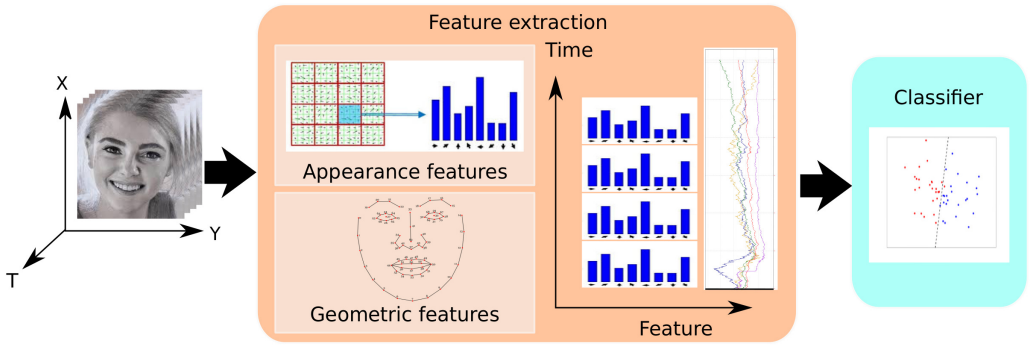


Figure 3.2: Block diagram of dynamics (video) based methods.

In dynamic approaches, combination of spatial as well as temporal activity is used to model the emotions represented by a face (refer fig. 3.2). Dynamic methods are able to overcome the limitation of the static approaches by analyzing the variation in facial activity with time. These methods tend to be computationally expensive and can not be deployed on mobile devices for real time applications.

[62] presented a dynamic Bayesian network based model to track and extract the facial activity. Model learns the evolution of facial expressions, and interaction of local and global facial features with each other. Bayesian model combines the prior knowledge with learnt features from training data to provide a better modeling capability.

Long et al. [63] developed an unsupervised approach to classify video from unlabeled video data. First, Independent Component Analysis (ICA) is used to extract spatio-temporal features from the video, and then, these features are used to classify the video for specific application. According to authors, this approach is able to perform better than the statistical approaches for facial expression analysis task.

A novel feature extraction technique called as Local Gabor Binary Pattern from Three Orthogonal Planes (LGBP-TOP) was proposed by [64] to extract spatial as well as temporal features, simultaneously. The proposed feature extraction method was combination of two previously successful methods: LGBP and TOP. The methods use Gabor filter independently in XY, XT and YT planes. Results showed that these features are very robust to misalignment of the facial axis in consecutive frames.

[65] presented another feature extraction method for extraction of spatio-temporal features. According to the paper, proposed Spatio Temporal Local Monogenic Binary Pattern (STLMBP) features are invariant to orientation, phase and illumination variation. Hence, they are good at extracting features in unconstrained environment. The monogenic filters were used to extract magnitude, phase and orientation of images. Multiple kernel based learning approaches were used on these features for correct classification.

Fan et al. [66] developed a hybrid network for emotion recognition in videos. Network contained two input pipelines. First network was 3D convolutional neural network (3D-CNN), which has capability to extract spatio-temporal features simultaneously. Second pipeline had CNN paired RNN. CNN extracted the spatial features and RNN learned the temporal relationship between the features of successive frame.

3.1.3 Multiple modality based methods

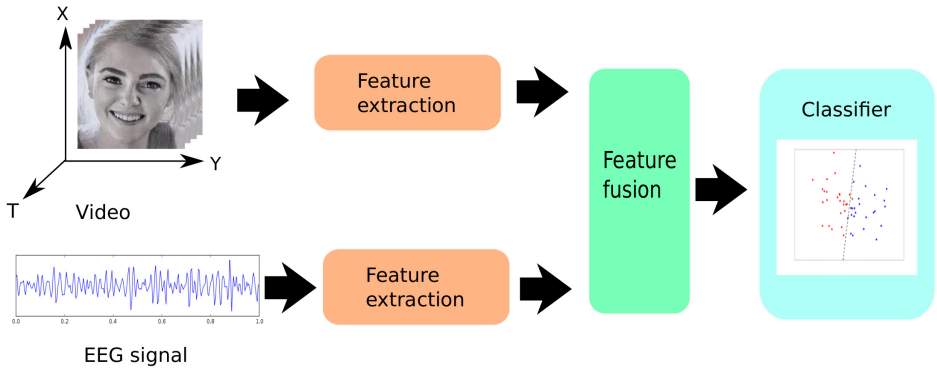


Figure 3.3: Block diagram of multimodal systems for FER.

Multimodal approaches combine image or video data along with other physical signals captured from subject; such as, audio, EEG, ECG etc. Addition of other physical signals provide more concrete information on the state of the subject, leading to a better hypothesis. Fig. 3.3 shows the block diagram of a general multimodal system.

Han et al. [67] uses multimodal regression approach to analyze the valence and arousal activity on face. The paper uses strength modeling technique which hierarchically combine models based on their strengths. Support vector regression and Bidirectional Long Short Term Memory (BLSTM) network were used to learn and extract the features from audio and visual modalities.

[68] developed a Brain Computer Interaction (BCI) system for emotional analysis using EEG signal. They employ multiple prepossessing and feature extraction methods to accurately classify the emotion. Kernel based classifier is used to classify the valance and arousal activity from physiological signal. [69] developed a sentiment analysis system based on textual and multimedia information present in the social media posts. The paper employs multi kernel learning approach to train different models simultaneously for different modalities of input data. CNNs are used to extract the features from video and text inputs while openSMILE [70] is used to extract the audio features.

A deep belief network based system was developed by Zheng et al. [71] to classify the emotions in positive, negative and neutral categories. The EEG signals were collected by providing external stimuli to subjects with different types of videos. Experiments were conducted with different number of EEG profiles; i.e. 4, 6, 9 and 12 channels.

[72] developed a multimodal system with specialist models learning specific modalities from the input video sequence. The system was designed for the video based emotion recognition challenge in the the emotion recognition in wild (EmotiW) competition. In this ensemble of models, CNN focused on visual spatial features from input frame, deep belief net learned the audio modalities and K-mean clustering algorithm extracted features from the mouth regions of each frame. These features were combined and passed to final classifier to predict the emotion represented by the image.

Kaya et al. [73] developed a multimodal system for EmotiW competition. The paper used combination of both feature learning and hand crafted feature extraction

methods. CNN was used to learn spatial activity corresponding to different emotions. Conventional (hand crafted) feature extractors; like, SIFT-FUN, LPQ-TOP, LGBP-TOP etc., are also used to extract visual structure from the face. The paper uses openSMILE library [70] to extract the audio features from the input video. Two different models (Partial Least Square (PLS) and Extreme Learning Machine (ELM)) are trained independently on these extracted features. Later, The output scores of these two models are combined using weighted averaging.

Turker et al. [74] used audio and video features to identify the laughter in the natural environment. Dataset was manually labeled for laughter sound and other environmental noises. Combination of head pose and facial features are used along with raw audio to identify the required event. Different modalities are fused and classified with the help of SVM classifier. Deep neural network was also infused into the system to increase the detection accuracy of the system.

3.2 Affective computing for sentiment analysis

Affective computing has huge scope in analysis of sentiment from human facial expressions in social media and other applications. With recent growth of the internet and social media platforms, these platforms have become accessible to huge amount of population. Hence, companies use them for advertisement and retrieving user feedback (sentiment). But most the data uploaded by users contains multimedia images. Sentiment of these images can not be extracted by traditional NLP based sentiment analysis systems. Hence, new approaches are being developed to extract the sentiment from images. Most of these image based sentiment analysis systems use hand crafted feature extraction approaches, such as, Local Binary Patterns (LBP) [75], Histogram of Oriented Gradients (HOG) [35], Scale Invariant Feature Transform (SIFT)[76], etc.

CENTRIST [77] (a scene level feature descriptor) was used by [9] to extract sentiment represented by the input image. CENTRIST feature descriptors were extracted to encode the scene level sentiment information. Then, Pyramid Histogram of Oriented Gradients (PHOG) and Local Phase Quantization (LPQ) feature extractor were used, on the segmented faces of people in the scene, to identify their facial expression. A multikernel classifier is used on fused features to predict the sentiment of the image.

Li et al. [78] developed a system to predict happiness intensity represented by an image. The system predicted increasing level of happiness with score in between 0 and 5. The system contained four submodules arranged as four stages of processing. First and second stage were used to extract scene level and facial level features, respectively. Third stage uses LSTM model, which was trained on extracted features to learn conditional dependence between all the features. Last stage applies regression model on the output of third stage to predict the happiness intensity between defined levels.

Geometric feature based model was developed by Vonikakis et al. for happiness intensity prediction [79]. This paper used forty nine facial keypoints to learn facial muscle movement corresponding to the level of happiness represented by the image. In another work by Sun et al. [80], hand crafted feature methods (like, Dense SIFT) are combined with the featuring learning approach to extract much robust but task specific feature representations. LSTM model is used on these extracted feature representations to learn their joint probability distribution. A densely connected neural network is used on the output of LSTM as regression model to predict the happiness intensity.

You et al. [81] used CNN based progressive learning approach to learn the target sentiment from a weakly labeled dataset collected from Flickr. They also used fine tuning to make model more efficient on the sentiment classification. The CNN model was pretrained using large weakly labeled Flickr dataset, and later fine tuned using manually labels images collected from twitter. [82] studied social media images to analyze the social interaction between individuals, and how it influences individuals on the social media platform. A probabilistic graph based model was proposed by them to analyze the influence of one individual over another in the image. This influence graph was used to predict the emotion represented by the image.

A very illustrious frame for sentiment analysis known as Sentribute was proposed by Yuan et al. [83]. The framework predicts the emotion represented by the image based on the content of that image which doesn't necessarily have to have humans. It takes low level features from the image and converts them into the mid level features using set of classifier. They also extract face features using Eigenface model, which are later fused with mid level scene features (if the image contains the faces). The final classifier predict the sentiment represented by the image using these combined set of features. SentiBank [84] is another emotional sentiment analysis framework

developed for sentiment prediction from images in social media. SentiBank links each social media image to one of the 1200 Adjective Noun Pair (ANP); these ANPs are then used for prediction of final emotion.

3.3 Deep learning architectures

Last decade has seen huge amount of research in deep learning due to its success in wide range of applications [85, 86, 87, 88, 89]. One of the main reason for this recent success of neural networks is availability of large datasets produced by industrial giants. Secondly, induction of Graphical Processing Units (GPU) into deep learning research has greatly increased computation power and reduced the development time for new test architectures. Deep learning algorithms are now performing significantly better than conventional learning algorithms in computer vision applications; such as, image classification [90, 6, 3], image segmentation [91], object detection [85, 86, 87], image enhancement [88, 92], image retrieval [89, 93, 94] and tracking [95]. Additionally, machine learning approaches are now being used in different multidisciplinary domains with very good results [96, 97, 98, 99].

First big breakthrough in deep learning was by [3]. In year 2012, the deep convolution neural network proposed by Krizhevsky et al., called as AlexNet (fig. 3.4), was able to outperform the conventional learning algorithms in ImageNet competition [100] for the classification task. The competition requires computer vision algorithm to correctly classify the images into one of the thousand classes. The model was able to achieve 15.3% error rate against its nearest competitor with error rate of 26.2%. This network used dropout [101], minibatch learning along with momentum based SGD optimization [102]. This event triggered the interest of research community back into neural network research.

In year 2014, [4] proposed VGGNet architecture (shown in fig. 3.5) with more depth and higher kernel count to beat the AlexNet on ImageNet dataset. The key idea of this paper was that they replaced larger sized kernels (11×11) by smaller sized and constant sized kernels (3×3). This model also had higher depth of 16 layer in comparison to AlexNet. According to authors, smaller sized kernels and higher depth provided model the capability to learn more robust features.

Same year, [5] proposed an inception based CNN architecture. It has also gained a lot of popularity in computer vision application due to its robust and general purpose

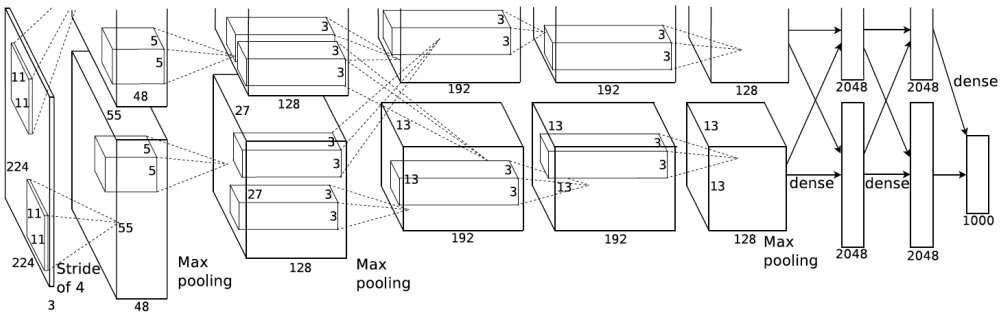


Figure 3.4: Architecture of the AlexNet CNN model [3].

Figure 3.5: Architecture of VGGNet CNN model [4].

learning architecture. Instead of using constant size kernels in a given layer, as used by its predecessors, inception architecture used different sized kernels as shown in fig. 3.6. Each layer has multiple kernels of different sizes, convolving with same input data parallelly. Fig. 3.6 shows single layer of the inception based architecture. These blocks are stacked together to build the required depth for a given task. This architecture had 22 layer and 4 million parameters. On the contrary, AlexNet and VGGNet had 60 million and 138 million parameters, respectively (even with much lower depth). This reduction in parameter count was due to removal of dense layer, which were used earlier as part of classification module at the end of feature reduction process.

Another very commonly used CNN architecture, called as Residual Neural Network (ResNet), was proposed by [6]. The ResNet architecture used skip connection approach (refer fig. 3.7). ResNet model is divided in blocks called as residual blocks (shown in fig. 3.7). Each block uses skip connection which connects the input (x) to output ($F(x)$) of that block and every block contains multiple convolutional layers. This skip connection or residue addition approach helps to avoid vanishing gradient problem in architectures with very high depth. The paper was able to optimize the

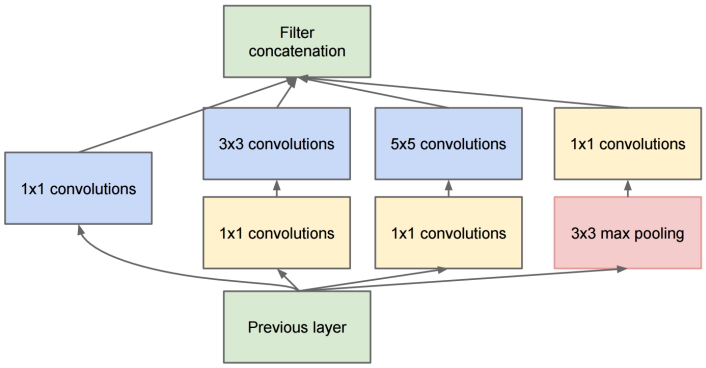


Figure 3.6: Inception block used in CNN architecture proposed by [5].

model with a depth of 150 layers using the residue learning approach.

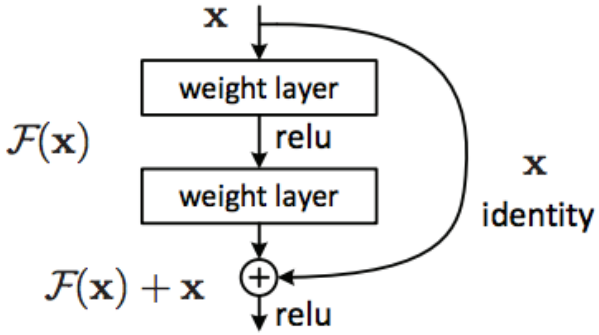


Figure 3.7: Architecture of residual network [6].

[103] proposed DenseNet CNN architecture using residual learning approach from different prospective. DenseNet connects current layer of the model with all its preceding layers using residual connections. According to authors, using output of previous layers at each layer helps in concurring the problem of vanishing gradient, and also helps in reducing parameter count.

The Xception CNN architecture [104], proposed by Chollet, improves learning mechanism in CNN architectures. The paper proposes depthwise separable convolution operation. Although, general architecture of the proposed model is same as [5] with a single exception that instead of performing convolution for the complete depth of the model, Xception architecture only considers a part of feature depth for convolution operation. The mechanism helps in reducing the parameter count of the model.

[105] proposed network-in-network architecture to increase representation capability of the CNN model. It places multilayer perceptron (MLP) network between two consecutive convolution layers. Thus, instead of directly passing convolved features from the previous layer, they are passed through MLP. MLP provides the nonlinear local relationship between the features of current layer.

[106] proposed maxout based CNN architecture. The architecture keeps only maximum value of the output feature maps in a layer. Model compares all the feature values produced by different kernels for a particular location in the feature map and retains only highest value for that location while discarding the rest. Thus, only most dominant features of the input data are considered for further processing. This procedure improves generalization capability of the CNN model.

3.4 Non-linear data modeling with deep learning

A simple perceptron is a linear modeling function given by equation below:

$$y = f(\boldsymbol{W}\boldsymbol{X} + \boldsymbol{B}) \tag{3.1}$$

where W and B are the weights and biases of the model, and X and y are input and output. The activation function (f) plays a very important role in nonlinear modeling of data for neural networks.

The earlier activation, like sigmoidal and hyper tangent function, had problem of exploding and vanishing gradients. Hence, the older neural networks were unable to scale the depth limiting their representational capabilities. In 2010, Niar et al. proposed a new activation function called as Rectified Linear Unit (ReLU) [7]. Till date, ReLU is an acclaimed activation function, and is used in most of the applications. Reason for the success of ReLU is its linear behavior and non-saturating nature (shown in fig. 3.8). ReLU clips negative part of the input and transfers positive part to output without any alteration (eq. 3.2). This helps linear flow of gradient during back propagation in deep models. On the other hand, when unit is stuck with a negative value of input due to random initialization, it never gets a chance to change the connected weights (as no gradient will flow through that unit).

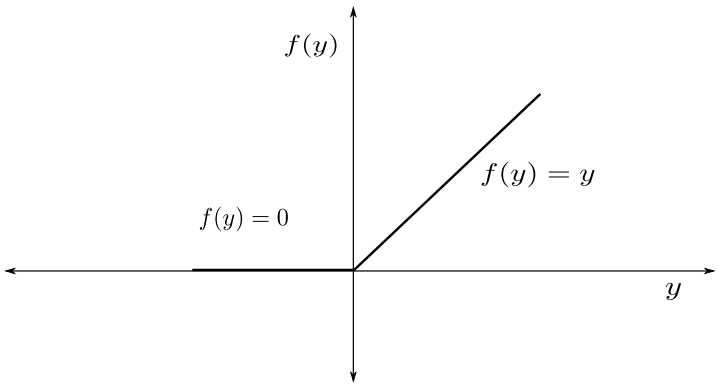


Figure 3.8: Rectified linear unit activation function [7].

$$f(y) = \begin{cases} y, & \text{if } y > 0 \\ 0, & \text{if } y \leq 0 \end{cases} \tag{3.2}$$

To overcome the problem of dead units in ReLU, a number of other activation functions have been proposed. Leaky ReLU [8] (shown in fig. 3.9) resolves the predicament by assigning non-zero slope value to the negative section of the input, while positive part of the signal is treated similar to ReLU (eq. 3.3).

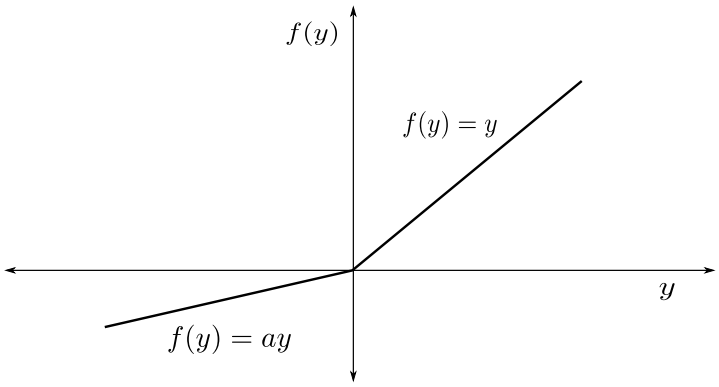


Figure 3.9: Leaky rectified linear unit activation function [8].

$$f(y) = \begin{cases} y & \text{if } y > 0 \\ ay & \text{if } y \leq 0 \end{cases} \tag{3.3}$$

Parametric rectified linear unit (PReLU) [107] uses trainable parameter as the slope coefficient instead of predefined value for the negative part of the input (eq. 3.4). Negative slope coefficients are trained along with the model weights during training. Activation functions in all of the layers learn different parameter values, and therefore they show unique activation behavior for the negative part of input in each layer. [107] claims that PReLU is the major factor in surpassing human level classification accuracy on ImageNet [100]. [108] proposed Randomized Leaky Rectified Linear Unit (RLReLU), which uses non-linear random coefficient instead of linear. Negative slope coefficient is sampled from the distribution around given slope value. It helps in avoiding chances of over fitting over the training data.

$$f(y) = \begin{cases} y^m, & \text{if } y > 0 \\ a^m y^m, & \text{if } y \leq 0 \end{cases} \tag{3.4}$$

Some other activation units use non-linear functions to model the activity of neuron. [109] proposed exponential linear unit (ELU), which uses exponential function to model the negative part of the input (eq. 3.5). ELU provides solution to output variance and bias problems. Saturation in negative part pushes activity of neurons toward zero mean, and pulls down the variance in the negative part. It was advanced by Parametric Exponential Linear Unit (PELU) [110], which used trainable parameters to control behavior of the exponential function.

$$f(y) = \begin{cases} y, & \text{if } y > 0 \\ a(e^y - 1), & \text{if } y \leq 0 \end{cases} \tag{3.5}$$

Goodfellow et al. [106] proposed maxout activation function, which takes maximum value out of outputs produced by several neurons. Adaptive Piecewise Linear (APL) unit proposed by Agostinelli et al. [111] uses hinge shared linear function as activation layer. It calculates piecewise linear function independently for each neuron and learn them during training process. [112] also proposed a trainable activation function, whose shape can be learned using linear regression model. The activation function can take any shape according to the given data. A non-linear activation function was proposed by [113] to solve time varying non-linear equations. Authors

also proposed that activation can help neural network converge in a finite time. Mathematical formulation for the upper bound on convergence time was presented in the paper. [114] designed an activation function specially for recurrent neural dynamics. The paper tries to find the roots of non-linear equations using recurrent neural network based on this activation function.

4 Sentiment analysis from social media images

4.1 Introduction

Analysis of social media content is helping companies to analyze customers' behavior and design their products and services accordingly. Sentiment analysis of textual data, from various social media sources, provides a convenient way to collect customer's experiences and is used by companies to make their products better. In recent years, a sharp surge has been observed in multimedia content on social media platforms compared to textual. It creates the need for a system capable of analyzing the multimedia content and produce relevant information. Social media images mostly contain multiple individuals in a wide range of social settings. Extracting emotional sentiment from these images is a very complex task due to variation in background conditions and number of subjects in the image.

There are multiple independent factors affecting perceived emotion in a group image. Some times different local regional features may convey different information; such as, there could be very high variation in facial expressions of individuals within a single image. For example, if we look at fig 4.1, face labeled as FACE-1 represents neutral emotion whereas face FACE-2 shows positive (happy) emotion, but whole image is labeled as negative emotion in the dataset [9]. Therefore, it is not correct to predict the sentiment in an image based on just a few faces. Surroundings of individuals also play a key role in final group emotion prediction. Hence, a multi modal system, which can capture various independent features from the given scene, is required for such a task.

We use EmotiW Group Effect Database (GED) 2.0 [9], developed by EmotiW 2017 competition organizers. In this dataset, images of a group of people are divided into three categories representing positive, negative and neutral emotions. The dataset

contains images from a wide range of social scenarios. The objective is to predict correct emotion on test data by considering all possible local and global factors contributing to final perceived emotion in an image.

Present chapter proposes a novel method to predict the sentiment represented by an image by combining local features (facial expressions of subjects) and global features (scene level physical characteristics) in an image. Joint probability distribution of physical and facial features provides the sentiment represented by the image. This unknown distribution over facial and scene features is learnt using Long Short Term Memory (LSTM) networks. We use Tensorflow [115] library for designing different types of deep learning models, and OpenCV [116] for data preprocessing and other image operations.

Figure 4.1: Sample from Group Effect Database 2.0 [9] representing diversity of facial as well as surrounding features.

4.2 Proposed method

In spite of considerable amount of research in facial emotion analysis, social group images based sentiment analysis remains under-explored. This work targets this problem with deep Convolutional Neural Network (CNN) based feature learning instead of hand crafted feature extraction techniques. Hand crafted feature extraction techniques sometimes miss key features contributing to the final label whereas feature learning techniques do not have such limitations [83, 84]. The proposed system contains two CNN based models to learn two different modalities. First model (scene CNN model) learns scene level features to identify the emotional sentiment

represented by the image. In second model (face CNN model), we train a different CNN architecture to learn to predict facial emotions on the faces of subjects in the image.

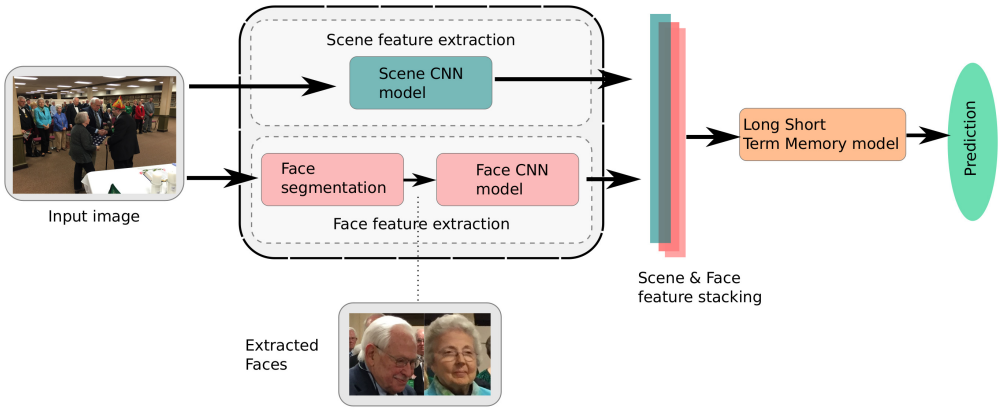


Figure 4.2: Proposed image based group sentiment analysis system. The system extracts features from scene and face separately, and then stacks them for overall score calculation using LSTM.

Block diagram of the proposed framework for group level emotion classification is shown in fig 4.2. The system contains two submodules:

1. Scene feature extraction submodule focuses on scene level spatial components contributing to final label, and
2. Face feature extraction submodule extracts facial expression of people in an image.

LSTM network is used to produce a multivariate joint distribution over the face and scene features. The LSTM model is chosen for the task, based on their better ability to model conditional joint distribution of inputs over other types of recurrent neural networks. Output multivariate joint distribution is used to estimate the probability of co-occurrence of various features in images for a predefined class using Softmax regression model.

4.2.1 Dataset

Group Effect Database (GED) 2.0 [9] contains images from all types of social scenarios we encounter in daily life. The dataset is divided into three sets for training,

validation and testing. The training data has 3630 training images divided into: 1159 samples for negative emotion, 1199 samples for neutral, and 1272 samples for positive emotion. Validation data has 2065 images containing 564, 728 and 773 images for negative, neutral and positive emotions, respectively. The test data contains 772 unlabeled images. Some random samples from GED 2.0 dataset are shown in fig 4.3, which represent the diversity of conditions covered by dataset. With very few training samples in GED 2.0, it is impossible to train a large CNN model without struggling with over-fitting. Therefore, we use ImageNet dataset [100] in this work for model pre-training. Additionally, FER-2013 [117] and SFEW 2.0 [118] facial emotion databases are used for training the Face-model (refer Section 4.2.3).

(a) Negative sample (b) Neutral sample (c) Positive sample

Figure 4.3: Sample images from GED 2.0 dataset

Data preprocessing

The training data contains very low number of images (only 1159, 1199 and 1272 samples of negative, neutral and positive emotion category, respectively) to train a deep CNN model. Therefore, data augmentation techniques are used to increase the number of samples. For augmentation, a random batch of images are read from each category. For each image, a sample is generated by cropping 3/5 region along the height and width. Five such samples are generated per image. Four samples are generated by cropping regions from each of the four corners, and fifth sample is from the center. Then these samples are flipped horizontally. In total, each training image provides 10 samples to train the model, thus providing 3630×10 samples. The pixel values of images in training data are normalized within range $[-1, 1]$ to

maintain a uniform color spectrum throughout the dataset, eq. 4.1.

$$x_{normalized}(i,j) = \frac{2 \times x(i,j)}{x_{max}} - 1 \tag{4.1}$$

here, $x_{normalized}(i,j)$ is the normalized output and $x(i,j)$ is input image. x_{max} represents the maximum pixel value in the given image.

4.2.2 Scene-model

A CNN model based on Inception V3 [119] architecture is developed as base model for learning the scene level contributing features. This model would be referred as Scene-model in rest of this chapter. [119] achieved highest classification accuracy on world's largest image dataset (ImageNet). It makes Inception architecture best feature extractor for any computer vision task. [119] also combines multiple sized kernels, providing a receptive field which combines multiple sized spatial correlation based on target label requirements.

The Inception V3 modules (shown in fig 3.6) are stacked together to get the required depth in the model. "Previous layer" block (fig 3.6) represents output of the preceding layer, which is passed to four parallel pipelines performing multilevel convolution with kernels of size 1×1, 3×3 and 5×5. These different sized kernels are able to capture different sized spatial correlations. Output of all these kernels are finally concatenated at "Filter concatenation" block.

Model training

Scene-model developed for scene feature extraction is based on Inception V3 architecture and contains 11 inception modules, shown in fig 4.4. In fig 4.4, $Block - n$ ($n \in 1 \ldots 11$) represent the Inception V3 modules from fig 3.6. This model is pre-trained on ImageNet dataset to get a generalized initial weight configuration. Pre-training the model on enormous ImageNet dataset helps in achieving generalized weight configuration, suitable for extracting wide variety of features.

This pre-trained model is then fine tuned with task specific dataset (GED 2.0) using data augmentation (refer Section 4.2.1). Instead of fine tuning the whole model from

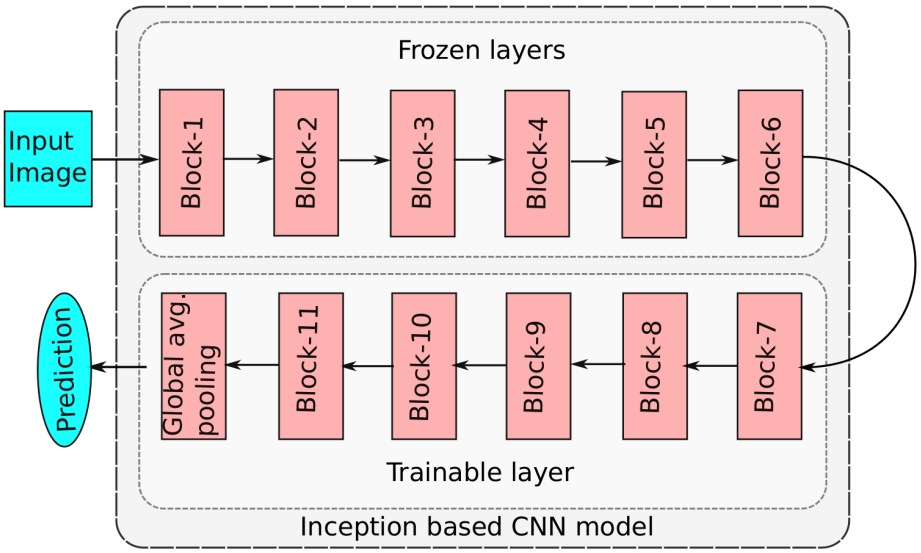


Figure 4.4: Fine tuning of Scene-model architecture to extract scene level features. Some of the layers are frozen and the rest remain trainable for faster and accurate prediction of group emotion.

end to end, we fine tune only certain percentage of the total weights. The choice, whether the weights of a certain layer are going to be fine-tuned or kept frozen, is made empirically. Model is trained multiple times for each set of frozen layers. Table 4.1 shows the maximum accuracy achieved by fine tuned Scene-model with different number of frozen layers. For this experiment, initial layers of models are frozen and end layers are fine tuned. In first experiment, when all the parameters of the model are fine tuned, it achieves lowest validation accuracy of 54.16. Similarly, accuracy also decreases when only last block of model is fine tuned. In general, model is able to achieve highest accuracy when around half of the layers are fine tuned in comparison to full fine-tuned model. Although, there is no specific pattern observed in the fine tuning process. But as a rule of thumb, it is better to search for best fine tuned model around space where approximately half of the layers are frozen. This significantly reduces search space and computation cost. The representation of fine tuning of the model is also shown in fig 4.4.

The experimental results are also supported by the work of Zeiler et al. [120]. According to which, the initial layers of model learn local pixel correlation in receptive field of their respective kernels. But as we go deeper into the model, level of ab-

straction starts to increase and end layers in CNN models generally learn objects or object parts for given data. This implies that local features in all objects are generally same (e.g. edges, texture, etc.), but global features change with data. From this discussion, we can establish the fact that during domain transfer process, there is no need for fine-tuning the whole model because initial model layers learn generic features. Therefore, only backend of the model needs to be fine tuned with application specific dataset. This phenomenon is also substantiated by the results of table 4.1.

Table 4.1: Effect of changing number of trainable weights during fine-tuning on model accuracy.

S.No.	Trainable blocks	Frozen blocks	Model name	Validation accuracy
1	block-1 - block-11	None	FreezeNone	54.16
2	block-3 - block-11	block-1 - block-2	Freeze_1-2	66.19
3	block-5 - block-11	block-1 - block-4	Freeze_1-4	65.29
4	block-7 - block-11	block-1 - block-6	Freeze_1-6	65.70
5	block-8 - block-11	block-1 - block-7	Freeze_1-7	**66.40**
6	block-9 - block-11	block-1 - block-8	Freeze_1-8	64.57
7	block-11	block-1 - block-10	Freeze_1-10	62.49

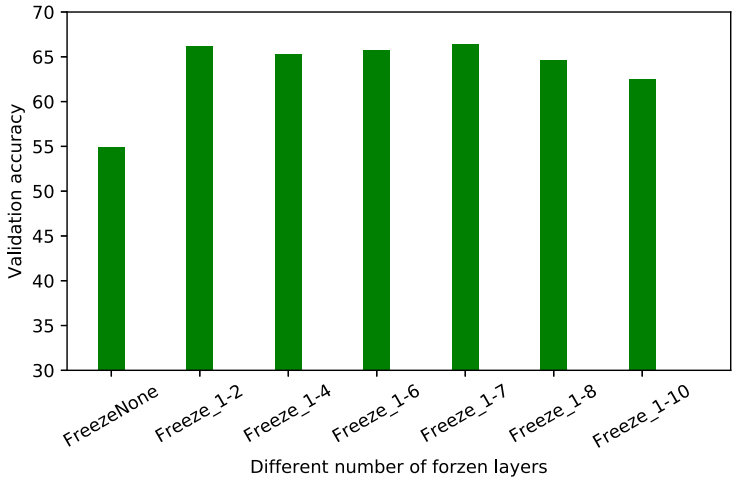


Figure 4.5: Bar chart showing the effect of fine tuning of different number of blocks in Inception based Scene-model.

Bar chart of the model accuracies at different number of frozen layers is shown in fig 4.5 and plot of validation accuracy of the model with respect to number of training epochs for different number of frozen layers is shown in fig 4.6. $freezeNone$

represents the model with all the fine tuned layers and $freeze\_1-2$, $freeze\_1-4$, $freeze\_1-6$, $freeze\_1-7$, $freeze\_1-8$, $freeze\_1-10$ represent models with 2, 4, 6, 7, 8, 10 layers kept frozen during fine tuning, respectively. Plots reveal some interesting facts about the model behavior during fine tuning with different number of layers. The model with lower number of trainable layers have faster convergence rate than the model with higher number of trainable layers. For example, $freeze\_1-10$ (black curve) reaches its best performance within just 10 epochs, while $freezeNone$ (blue curve) takes around 40 epochs to achieve best validation accuracy. Another important obserbation from this experiment is that models with very few fine tuned backend layers show very sharp slope in validation accuracy till they reach plateau. While the models with all or maximum number of layers fine tuned, show very slow progress in validation accuracy.

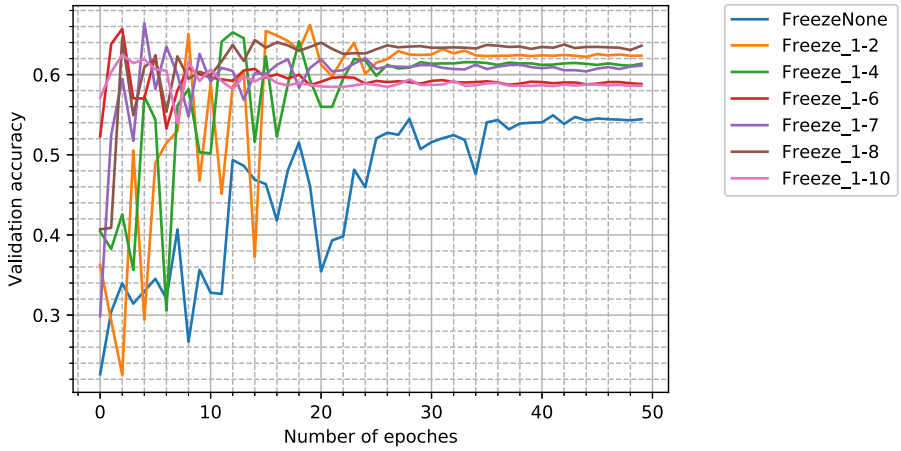


Figure 4.6: Plot of progress in validation accuracy of model with respect to epochs during fine-tuning process for different number of layers kept frozen.

Once fine tuning procedure is established, categorical cross-entropy is used as loss to measure the deviation given by eq. 4.2.

$$L(y', y) = -\frac{1}{N}\sum_{i=0}^{N}[y_i log(y'_i) + (1-y_i) log(1-y'_i)] \tag{4.2}$$

where, $L(y', y)$ is the value of loss, y_i and y'_i denote target and predicted labels, respectively. N is total number of samples per mini-batch.

RMSProp gradient descent optimization method is used to back-propagate the error

through network. If L_t is the loss value for a particular example and θ_t is the parameters' value at time t, and γ and α are decay terms for gradient and leaning rate, respectively. Then, the mean square term (r_t) is given by:

$$r_t = (1 - \gamma) \left[L'_t(\theta_t)\right]^2 + \gamma r_{t-1} \tag{4.3}$$

here, L'_t is derivative of loss with respect to parameters θ_t. γ is the momentum term and can be used to control the contribution of present and past gradients.

Actual update gradient (v_{t+1}) is calculated using following eq:

$$v_{t+1} = \frac{\alpha}{\sqrt{r_t}} L'_t(\theta_t) \tag{4.4}$$

here, α is the learning rate of optimization function.

For set of parameters defined by θ_t at time t, the next set of values (θ_{t+1}) are given by eq. 4.5:

$$\theta_{t+1} = \theta_t - v_{t+1} \tag{4.5}$$

The model is trained for 50 epochs with scheduled learning rate. At the start of training 0.001 learning rate is used, and test loss is observed at the end of each epoch. If test loss does not improve for 5 continuous epochs then learning rate is dropped by a factor of 1/5. This policy is adopted for complete training process.

4.2.3 Face-model

After training the CNN model to classify the sentiment in an image based on scene features, a second model is developed and trained to extract the facial emotions of individuals in the image.

4.2.3.1 Face data preprocessing

FER-2013 [117] and SFEW 2.0 [118] datasets are used to train the model for different classes of facial emotions. Both datasets have seven basic emotion; i.e., angry, disgust, fear, happy, neutral, sad, and surprise. Some samples from both of the datasets are shown in fig 4.7. FER-2013 dataset has cropped (48×48) gray scale face images. While SFEW 2.0 dataset has wide scale images from movies with much smaller region of interest and sometimes even have more than one faces in image. Therefore, initially faces are segmented using Viola-Jones face detector [121] in SFEW 2.0 dataset, and then face, corresponding to the labeled emotion (base face), is identified based on area (face with maximum surface area). These faces are resized to (48×48) gray scale image (same size as FER-2013 dataset) to have a uniform dimensionality of dataset.

(a) Samples from Fer-2013 dataset.

(b) Samples from SFEW 2.0 dataset.

Figure 4.7: Sample images representing nature of data and emotions in facial emotion recognition datasets.

4.2.3.2 Face-model training

A second CNN model (Face-model) designed to classify the facial emotion is based on Xception architecture [104]. The Face-model is pre-trained on the combination of FER-2013 and SFEW 2.0 facial emotion datasets. Training set of FER-2013, and training and test set of SFEW 2.0 are used as training data for the Face-model. Test set of FER-2013 dataset is used to validate the performance of Face-model. This model is designed to predict 7 basic facial emotions.

The Face-model is not fine tuned for task specific dataset (as discussed in 4.1); as in GED 2.0 dataset, locally individual faces might provide wrong information regarding the globally perceived emotion. Therefore, the same model trained on SFEW 2.0 and FER-2013 is used without any fine tuning.

4.2.4 Face-Scene-model

For final model, which exploits both local as well as global features, we use Face-model and Scene-model as feature extractors. These models take raw image pixels and provide mid level feature representations. The features can also be treated as n-dimensional (n is number of feature per image) distribution of data based on different characteristics. Joint probability distribution of these features is modeled by Long Short Term Memory (LSTM) [122] models.

For training of LSTM model, training dataset (images from GED 2.0) is passed through Scene-model to extract scene level features at global average pooling layer (layer before classification layer), refer fig. 4.4. Then faces in training images are extracted (using Viola-Jones face detector) and resized to (48×48) gray scale images. Similar to Scene-model, all extracted faces pass through Face-model, and facial features are acquired from global average pooling layer (layer just before classification layer). Both Scene-model and Face-model provide 2048 dimensional scene and facial features.

Number of faces per image can vary a lot from image to image; therefore, top five faces (based on their percentage area in the image) are selected for feature extraction. This provides a uniform (5×2048) dimensionality of facial feature set. If an image has less number of faces, then output feature set is zero padded with $((5-n) \times 2048)$ sized array; where, n is number of faces detected in an image. Since LSTM networks are order dependent, these features (extracted using Face-model and Scene-model) are stacked together in a predefined fixed order. During feature stacking, scene features are always first vector followed by any random order of facial features.

Features from global average pooling layer in Scene and Face models are concatenated resulting in (6×2048) sized input vector. Single LSTM layer is used with 128 LSTM cells, providing 128 dimensional output vector. The choice of the order of LSTM model is based on empirical evaluation. Softmax regression model is used to convert this 128 dimensional joint distribution of global and local features into

class scores.

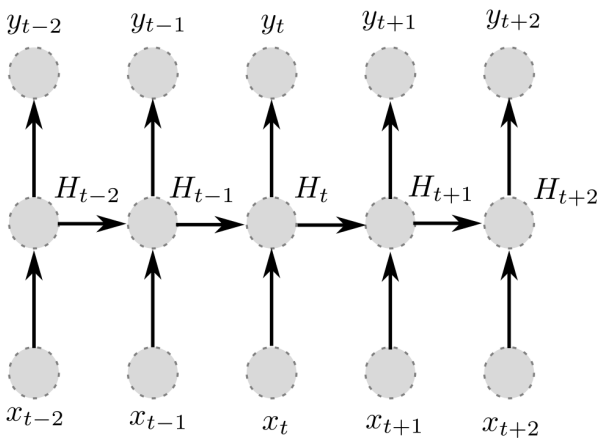


Figure 4.8: LSTM based architecture for showing relation between input and output of the model

The probability of receiving an output vector, given the face and scene feature vectors, is represented by following equation:

$$P(y_1, y_2....y_{T'}|x_1, x_2.....x_T) = \prod_{t=1}^{T} P(y_t|x_1, x_2.....x_T) \tag{4.6}$$

where $y_1, y_2....y_{T'}$ represent output vector and $x_1, x_2.....x_T$ represent input vector for LSTM. The input and output sequences have length T and T', respectively. In proposed model, input is combination of scene and face features; therefore, equation can be rewritten in terms of input scene vector (S) and face feature vectors ($F_1, F_2...F_n$) each with size (1×2048) (eq. 4.7). Hence, conditional probability of output vector ($P(y_1, y_2....y_{T'}|x_1, x_2.....x_T)$) is presented as the product of conditional probability of output given scene feature vector $P(y_t|S)$ and face feature vector $P(y_t|F_1, F_2.....F_n)$, respectively (eq. 4.7).

$$P(y_1, y_2....y_{T'}|x_1, x_2.....x_T) = \prod_{t=1}^{n} P(y_t|F_1, F_2.....F_n) \times P(y_t|S) \tag{4.7}$$

Physical implementation of the probability estimation using LSTM (for eq. 4.7) is

shown in fig. 4.8. At the time of implementation, eq. 4.7 is represented as:

$$y_t = f_\alpha(H_t, x_t; \theta) \quad \forall \quad t = 1, 2, 3.....T \tag{4.8}$$

$$H_t = f_\beta(H_{t-1}, x_t; \theta) \quad \forall \quad t = 1, 2, 3.....T \tag{4.9}$$

$$H_{t-1} = f_\gamma(H_{t-2}, x_{t-1}; \theta) \quad \forall \quad t = 1, 2, 3.....T \tag{4.10}$$

where, y_t is output of the model at time t. This output is function of current hidden state (H_t) and current input vector (x_t), given the model parameters (θ). In fig. 4.8, current hidden state is represented as a function of previous hidden state (H_{t-1}) and current input (x_t). Similarly, previous hidden sate is represented as a function of its input (x_{t-1}) and previous state (H_{t-2}), and so on.

By combing eqs. 4.8, 4.9 and 4.10; we rewrite the output as the function of all previous inputs (eq. 4.11).

$$y_t = f_\alpha(f_\beta(f_\gamma(.....(....x_{t-2}, x_{t-1}, x_t; \theta)))) \quad \forall \quad t = 1, 2, 3.....T \tag{4.11}$$

Combination of LSTM and Softmax regression setup is trained on the training data of GED 2.0 with categorical cross-entropy as the loss function (refer eq. 4.2) and RMSprop algorithm (eq. 4.3-4.5) for gradient back-propagation.

4.3 Results and discussion

We evaluate test data on Scene-model as well as on complete Face-Scene-model. First, performance of Scene-model is analyzed on validation set; and later, it is combined with Face-model to check improvement in model accuracy.

The Scene-model by itself is also able to achieve a decent accuracy of 63.43%. Table 4.2 shows category-wise accuracy achieved by Scene-model. The Scene-model is able to achieve higher accuracy on positive emotion compared to negative and neutral. Confusion matrix for the Scene-model is shown in table 4.3.

Table 4.2: Scene-model accuracy for positive, neutral and negative emotion along with overall accuracy.

Category	Accuracy
Overall Classification Accuracy	63.43%
Positive Classification Accuracy	62.06%
Neutral Classification Accuracy	57.57%
Negative Classification Accuracy	57.43%

Table 4.3: Confusion matrix for Scene-model presenting the number of correctly classified and misclassified samples for each class.

	Positive	Neutral	Negative
Positive	193	88	30
Neutral	29	95	41
Negative	38	88	170

Detailed description of performance of complete Face-Scene-model for each target class is given in table 4.4. Proposed model is able to achieve overall accuracy of 66.34% with highest accuracy of 69.93% for the negative group emotions. Confusion matrix for the Face-Scene-model is shown in table 4.5. By analyzing the model performance, it is observed that model has difficulty in classifying the neutral emotions and is biased toward negative emotions. Inclusion of Face-model has improved accuracy of Face-Scene-model in negative class when compared with Scene-model alone.

Table 4.4: Face-Scene-model accuracy for positive, neutral and negative emotion along with overall accuracy.

Category	Accuracy
Overall Classification Accuracy	66.34%
Positive Classification Accuracy	63.02%
Neutral Classification Accuracy	46.06%
Negative Classification Accuracy	69.93%

Table 4.5: Confusion matrix for Scene-model presenting the number of correctly classified and misclassified samples for each class.

	Positive	Neutral	Negative
Positive	196	66	49
Neutral	31	76	58
Negative	35	54	207

The proposed model is compared against other state of the art techniques used for scene classification [77, 123, 124, 125]. For comparative analysis, we considered scene feature extractor like scale-invariant feature transform (SIFT), Speeded Up Robust Features (SURF) [126] and Oriented FAST, Oriented BRIEF (ORB) [127] and CENTRIST [77]. All of them are known for their local scene feature encoding efficiency. In CENTRIST based classifier, Support Vector Machine (SVM) classifier is trained on CENTRIST features to classify the image [9]. For CENTRIST feature extraction, image is divided into 4×4 non-overlapping regions. Then CENTRIST, a Census transform based feature extractor, is used to extract features of these regions.

In rest of the methods shown in table, three levels of clustering are used to create visual word dictionary. Cluster size of 100, 200 and 400 is chosen for SIFT, SURF and ORB feature descriptors. Both SIFT and SURF provides higher density of spacial keypoints (3000 to 5000). Each keypoint is encoded into 128 and 64 dimensional space for SIFT and SURF, respectively. On the other hand, ORB is more efficient and robust algorithm. It uses FAST (Features from Accelerated Segment Test) [128] corner detector and provides comparatively more robust and lesser number of keypoints. ORB uses BRIEF (Binary Robust Independent Elementary Features) [129] feature descriptor and provides 32 dimensional descriptor.

Feature vector received from all descriptors are concatenated and clustered into 100, 200 and 400 size clusters using k nearest neighbor clustering algorithm. These clusters are used as visual word dictionary to create visual Bag Of Words (BOW) for training classifier. Again, SVM classifier is used to classify these BOW features.

Table 4.6, shows comparative performance analysis of Scene-model and Face-Scene-model against other state of the art techniques. Models using ORB feature with SVM classifier are referred as ORB-100, ORB-200 and ORB-400 (number represents the cluster size); models using SURF feature with SVM are presented as SURF-100, SURF-200 and SURF-400; and models based on SIFT and SVM are denoted as

SIFT-100, SIFT-200 and SIFT-400 in table 4.6. ORB descriptor based models achieve 42.24%, 38.88% and 36.55% accuracy on test set with 100, 200 and 400 cluster size, respectively. SURF based models achieve 49.11%, 47.43% and 46.26% accuracy on test set with cluster size of 100, 200 and 400, respectively. Although SIFT descriptor uses 128 sized feature vector, but SIFT based models achieve accuracy of 44.10%, 44.48% and 42.80% only with cluster size of 100, 200 and 400, respectively.

Base line model uses CENTRIST features to encode scene level information and achieves 52.93% and 53.62% accuracy on validation and test data, respectively. Scene-model, as we have fine tuned only half of the model layers, is able to use learnt scene description to encode task specific feature information. It is able to achieve 66.40% and 63.43% accuracy on validation data and test data, respectively. Complete Face-Scene-model, which combines face features extracted by Face-model with the Scene-model features, gives 68.53% and 66.34% accuracy on validation and test data, respectively. Clearly, including local facial features in the final model improves the performance, resulting in 2.9% higher accuracy in Face-Scene-model than scene based model alone.

Table 4.6: Comparison of proposed model accuracy with state of the art models.

S.No.	Model Name	BOW clusters	Validation accuracy (%)	Test accuracy (%)
1.	ORB-100	100	41.31	42.24
2.	ORB-200	200	41.94	38.88
3.	ORB-400	400	39.85	36.55
4.	SURF-100	100	51.19	49.11
5.	SURF-200	200	49.20	47.43
6.	SURF-400	400	47.94	46.26
7.	SIFT-100	100	46.78	44.10
8.	SIFT-200	200	47.55	44.48
9.	SIFT-400	400	47.41	42.80
10	CENTRIST (base line)	-	52.97	53.62
11	Scene model	-	66.40	63.43
12	Face-Scene model	-	68.53	66.34

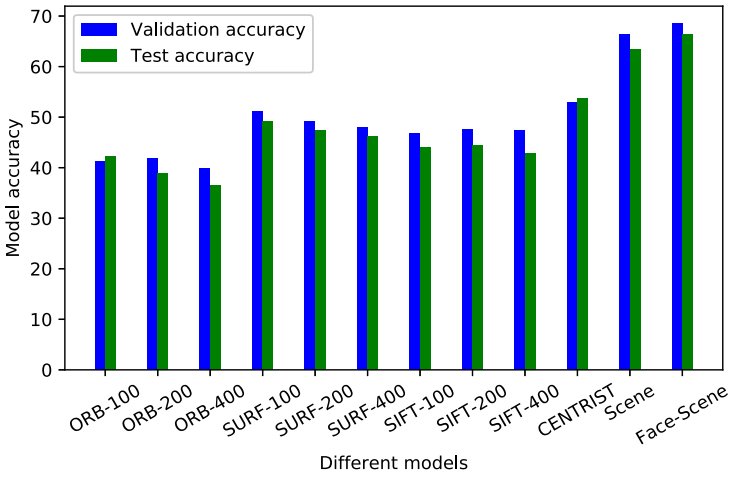


Figure 4.9: Bar chart of respective accuracies of Scene-model and Face-Scene-model against other state of the art models.

4.4 Conclusions

A novel framework is proposed in this work to analyze emotional sentiment in an image of a group of people. The framework is developed and tested on Group Effect Database 2.0. The proposed system contains two sub-modules, one of which focuses on scene level global features while the other targets facial expression based local features. Two different convolutional neural networks are designed and trained to learn scene features and facial expressions in an image, respectively. Features extracted by the scene feature extraction network and the face feature extraction network are given to LSTM network to learn joint distribution of features. Finally, these features are classified into positive, negative and neutral emotion by using Softmax regression. The solo scene based classifier is able to achieve a test accuracy of 63.43%, while by combining local face features with it, the test accuracy of the model becomes 66.34%.

5 Multimedia likability prediction system

5.1 Introduction

Facial expressions are the most powerful non-verbal tools used by humans to share different type information between themselves. Facial expressions have played significant role in the evolution of our species [130]. The expression convey true human response to any applied stimuli, irrespective of the person's intent to conceal his reaction. Study of facial expression finds application in wide range of area; such as, mental health diagnosis [131], gaming experience [132], customer services [133], automotive safety [134] and human machine interaction [135]. In last decade, facial expression analysis and emotion recognition has attracted substantial amount of interest from computer vision research community [56, 136, 40, 45, 60].

Involvement of different facial muscles makes facial expression analysis a very complex task [137]. The movement of facial muscles can also vary for each individual. There are two different type of approaches used to analyze facial expressions: appearance based methods [138, 139] and geometry based methods [140, 141]. Appearance based methods rely on various feature extraction techniques to capture the variation in facial features. Geometry based methods use fiducial facial point to model the movement of different facial muscles.

Most common application for study of facial expression is in human emotion recognition. According to earlier work of Paul Ekman [142], emotions can be divided into six basic categories: happy, sad, surprise, disgust, fear and anger. Although, more categories of emotions have been included by later works; such as, boredom, contempt and engagement etc. [29, 30]. These emotions are used to understand subject response and infer decisions. But, humans exhibit very complex behavior and sometimes response of subject to the provided stimuli can be a hybrid emotion; i.e.

generated emotion can be a combination of multiple emotions. Hence, approaches based on expression to emotion mapping can fail in complex scenarios.

Earlier emotion recognition and expression analysis methods used to rely on static images [143, 59, 56]. These methods use local spatial structures to identify the existing emotion. In other scenarios, where facial expression changes with time, these static methods remain inadequate. In such cases, dynamic (video) methods are preferred, which can extract spatial as well as temporal information from video of face [65, 62]. Sometimes multiple modalities are also used to understand the true emotion of subject [144, 145].

This work focuses on the commercial aspect of the facial expression analysis. In this chapter, we present a novel framework for automatic evaluation of likability of multimedia content by observing the facial expression of viewer. The system produces output in three classes: like, neutral and dislike. The following reasons make this task very intricate in nature:

1. The expression effectuated on face of subject are stimulated by the content of observed multimedia. Hence, expressions elicited are natural instead of acted ones. Natural expressions create very insignificant muscle movement in comparison to the acted ones. Therefore, they are hard to recognize correctly.
2. The multimedia can have a wide variety of content leading to wide range of expressions. For example, two different videos liked by a subject can have very different content and cause completely unique set of expressions. In other words, collected dataset will have very high intra-class variance.
3. Even in a single observed video, the subject can produce wide range of expressions depending on the flow of content.

The proposed system is a multi-modal system, which captures different types of features from subject face. The ensemble based architecture is designed to learn spatio-temporal features from input video sequence. The system is trained from end to end in single training cycle.

5.2 Motivation

With the advent of internet technology, online multimedia content has been continuously increasing. It has shown an exponentially increasing trend in last decade.

Popularity of social media sites like Youtube, Facebook, Twitter etc. are also one of the root factors. Facebook has around 2.23 billion active users every month [146]. Similarly, Youtube hosts more than 1.5 billion users every month [147]. Approximately, 30 million user use Youtube on daily basis out of its total user bank [148]. Over 300,000 user are paying Youtube for its services and 50 million users have uploaded a total of more than 5 billion videos.

This vast outreach of social media platforms has made them biggest commercial platform of this era. A large number of people are using these platforms as their main source of income. The corporations use these platforms to promote their products through advertisements. Additionally, online streaming services like Netflix, Amazon prime and Hulu etc. have also become mainstream sources of entertainment.

All these platforms are great way to get real-time feedback from the user and improve the quality of product. But current feedback systems ask user to manually provide feedback in the form of comments or by pressing like/dislike button. It has been observed that even a well performing content gets only around 4% view to feedback ratio. Whereas, maximum of the content gets less than 1% feedback. These statistics are too low and susceptible to deception. The feedback received by these methods can vary from true response, and is very small to be of any significance at early stage.

An automated feedback system based on facial expression of the viewer is a much better and faster way to collect the feedback of your target audience with 100% feedback rate. As the expression of the user are immune to deception, it is a good way to collect true response.

5.3 Data acquisition

5.3.1 Experimental setup

Data collection is very critical step for the development of a reliable system in any type of application. Any kind of biases and constraints in collected data can limit the scope of developed system. Hence, we tried to make the data collection process as realistic as possible.

Data collection process requires collection of two types of data. First data is the video samples, which are used to stimulate the natural response on face of subjects.

Figure 5.1: Some random samples from the videos used to stimulate the subject response.

Second data is the recorded response of volunteers against the shown video. For response stimulation, we selected more than 150 videos from a wide variety genres, such as, movie, music, sports, fights, war, speech, advertisements etc. Some samples from the used videos are shown in fig. 5.1 and number of samples in each genre are presented in table 5.1. Duration of these video clips varies between 2 mins and 5 mins. Length of recorded response is very important in this case. We observed that response of a subject varies a lot throughout the recording session. Hence, it is not feasible to identify the correct viewer sentiment based on a short length clip (e.g. 5-10 sec).

Table 5.1: Table showing the number of videos belonging to each genre.

Genre	Number of sample
Movie	25
Music	16
Speech	13
Advertisement	50
War/fight	18
Horror	11
Comedy	21

5.3.2 Sample recording

All recorded samples have a frame rate of 15 fps and frame size 640×480. In total, 73 volunteers were recruited for the task of response recording. All the recordings are

conducted in uncontrolled environment. Different locations and lighting conditions can be observed in all the samples shown in fig 5.2.

Figure 5.2: Samples from the recorded videos of subjects while watching the multimedia content.

Subjects are selected to maximize the diversity in gender and facial characteristics (shown in fig. 5.2). Subjects have wide range of facial features, such as, beard, mustache, turban, glasses etc. All the subject have age between 17 and 28 years. More details on subjects are given in table 5.2. Each subject is shown a video to stimulate the facial expressions; and these expressions are recorded using webcam. Only 3 to 6 samples are recorded from single subject in one sitting to avoid effect of physical fatigue. Each subject is urged to watch video from different genres to get a uniform number of samples for all categories of videos. Subjects are asked to rate the video in three classes: “Like”, “Neutral”, “Dislike”.

Table 5.2: Details of the volunteers recruited as subjects for data acquisition.

	Male	Female	Total
Video count	254	143	397
Subject count	29	44	73

Minimum age	18
Maximum age	28
Median of age	20
Mean of age	21.4

5.4 Proposed Method

5.4.1 Preprocessing of data

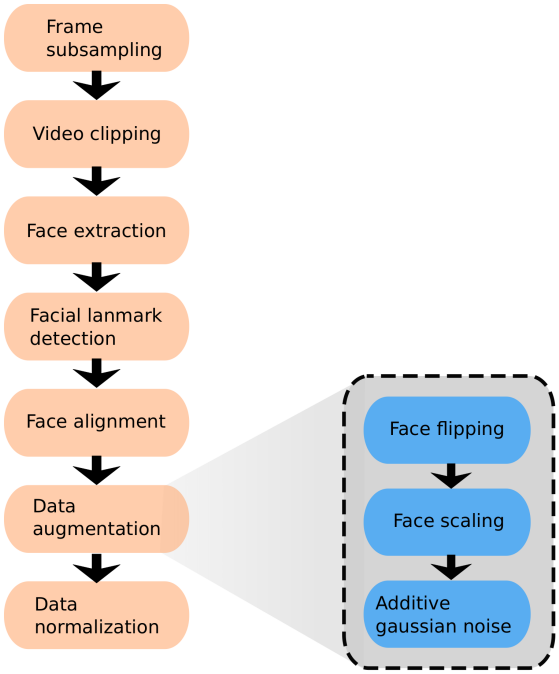


Figure 5.3: Block diagram of video preprocessing pipeline for the proposed system.

Deep neural networks are prone to overfitting and require large amount of data to achieve a better generalization on test data. Hence, multiple preprocessing methods are used on original videos to increase sample count for the training process. Block diagram of the data preprocessing pipeline is shown in fig. 5.3.

All the original videos of subjects are recorded at 15 fps. Using high frame rate can be computationally very expensive and do not hold any substantial advantage. Hence, first step of preprocessing is frame subsampling. We reduce the frame rate to 1 fps which is neither too low to loose significant information nor too high to increase computational requirements of framework. In next step, recorded videos are split into 1 min samples to increase the sample count for training data. One minute duration is long enough to provide enough arousal activity from subject's faces to correctly identify the response. This process enables us to increase sample count from 397 to 1372. Process of frame rate reduction and video splitting is also

shown in fig. 5.4. Total number of samples for each category after this procedure are shown in table 5.3.

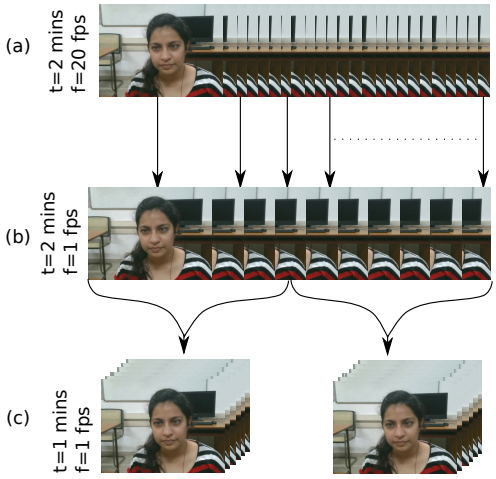


Figure 5.4: The frame rate reduction and video splitting procedure (a) Original video sample (b) Video sample of same duration but with reduced frame rate (c) Video split into multiple sample of 1 min duration.

Table 5.3: Count of samples belonging to each of the three categories.

Categories	Number of samples
Like	773
Neutral	317
Dislike	282
Total sample	1372

Next stage in preprocessing is segmentation of faces and extraction of facial landmarks in each frame. Dlib library [149] is used to extract the faces as well as facial landmarks from the input frames. The faces are extracted with frame size of (100×100) in RGB color space. Then, all 68 facial landmarks are extracted from each of the frame (shown in fig 5.5c). Each landmark has x & y coordinate, hence the output array of landmarks for each frame has (2×68) size. Results of face segmentation and landmark detection are shown in fig. 5.5.

Extracted facial landmarks are used for face alignment. Affine transformation [150] is used to align faces based on the location of selected landmarks. Only aligned face videos are used for model training. Data augmentation is also incorporated in the data preprocessing pipeline to further increase the variability in the data. We can not use all of the commonly used augmentation techniques on facial images due to

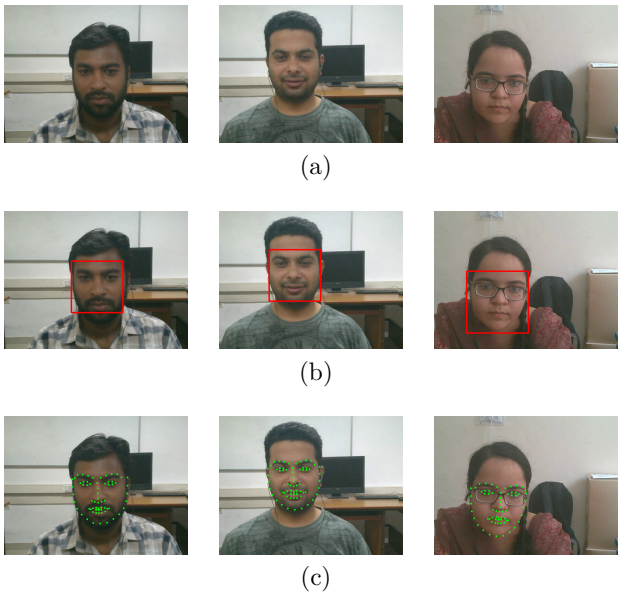


Figure 5.5: Results of segment faces and detected facial landmarks in video sequence (a) Frame from actual recorded video (b) Extracted faces from one frame of video sequence (c) Extracted facial landmarks from one frame of of video sequence

special geometry of faces. Three augmentation methods are used in this work: Face flipping, face scaling, and additive Gaussian noise. For face scaling, a scaling factor of 1.1 is used. Higher scaling factor can eliminate the important facial information. Additive Gaussian noise is also added to the frames of input video to achieve better generalization behavior from model. The Gaussian noise is generated with the help of following equation:

$$N_g(z) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(z-\mu)^2}{2\sigma^2}} \tag{5.1}$$

here, $N_g(z)$ represents the probability density function for Gaussian noise for given pixel value (z). μ and σ are mean and standard deviation for the noise and can be adjusted to change the amount of noise added into input sample.

The last stage of pipeline is data normalization. Data normalization is used to limit the range of data values so that all the videos have similar sample structure. The

formula used for data normalization is as follows:

$$X_{norm}(i,j) = \frac{x(i,j) - X_{min}}{X_{max} - X_{min}} \tag{5.2}$$

here, X_{norm} is the output pixel value for input pixel value x at coordinates (i, j). X_{max} and X_{min} are maximum and minimum value present in matrix for a given batch, respectively.

5.4.2 Proposed architecture

A novel Deep Composite Neural Architecture (DCNA) is proposed to learn the arousal activity on the faces of subject. DCNA is an ensemble of three different models, which learn different modalities from the input video sequence (shown in fig. 5.6). The models in the ensemble of DCNA are named as: TCNN-LSTM, 3D-CNN and Dual-LSTM.

The first model (TCNN-LSTM) is the combination of time distributed CNN (TCNN) and LSTM (represented as LSTM-1 in fig. 5.6). The second model is a 3D convolutional neural network (3D-CNN), which has three dimensional kernels to extract spatio-temporal features from the frames of video sequence. The third model does not take video sequence as input but uses coordinates of all 68 facial landmarks to map the arousal patterns of different landmarks in the given time sequence. This model contains two LSTMs (LSTM-2 and LSTM-3) to extract temporal patterns in both x and y coordinates independently. The architectural details, like kernel size and kernel count for TCNN and 3D-CNN, are given in table 5.4 and 5.5, respectively.

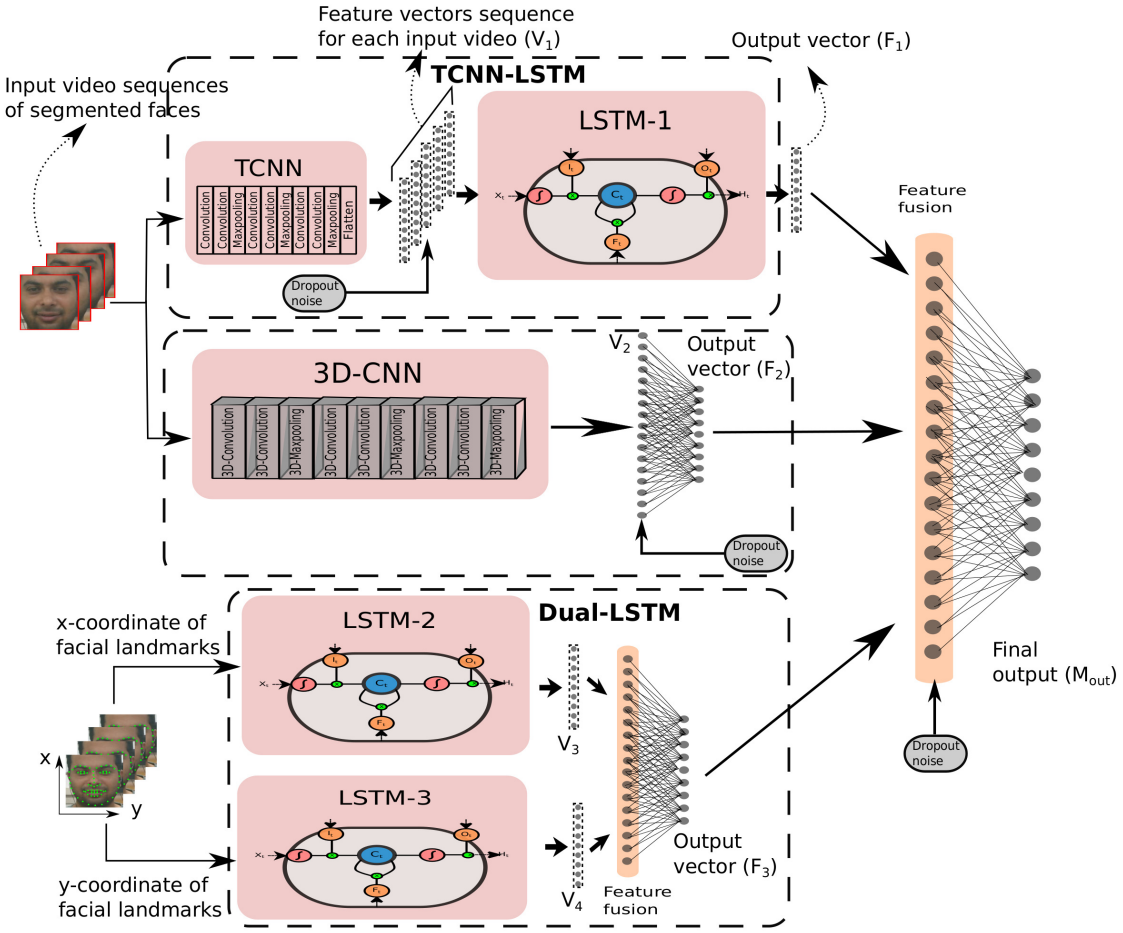


Figure 5.6: Proposed Deep Composite Neural Architecture (DCNA) with ensemble of three submodules to learn different modalities from input video sequence. The model is a single cycle end to end trainable system.

Table 5.4: Layer wise kernel size and count for TCNN architecture.

Layer	Kernel size	Kernel count
Conv2D	3×3	12
Conv2D	3×3	12
Maxpool2D	2×2	12
Conv2D	3×3	16
Conv2D	3×3	16
Maxpool2D	2×2	16
Conv2D	3×3	20
Conv2D	3×3	20
Maxpool2D	2×2	20

Table 5.5: Layer wise kernel size and count for 3D-CNN architecture.

Layer	Kernel size	Kernel count
Conv3D	$3 \times 3 \times 3$	12
Conv3D	$3 \times 3 \times 3$	12
Maxpool3D	$2 \times 2 \times 2$	12
Conv3D	$3 \times 3 \times 3$	16
Conv3D	$3 \times 3 \times 3$	16
Maxpool3D	$2 \times 2 \times 2$	16
Conv3D	$3 \times 3 \times 3$	20
Conv3D	$3 \times 3 \times 3$	20
Maxpool3D	$1 \times 2 \times 2$	20

The TCNN model is a 6 convolutional layer based CNN model (table 5.4). It fetches one frame at a time and produces a reduced feature vector per frame. Convolutional neural network extracts feature from the each frame and produces a 2304 dimensional feature vector for each image. Features extracted at each layer can be represented by eq. 5.3.

$$F_l(i,j) = \sum_m \sum_n \sum_d F_{l-1}(i+m, j+n, d) \times K_l(m,n,d) \tag{5.3}$$

here, F_l is the output feature map for convolutional layer (l) and F_{l-1} is the output feature map of previous layer. K_l is the convolutional kernel for current layer with size ($m \times n$). d representer the depth (number of kernels in previous layer). The final output of a time distributed CNN (TCNN) can be represented as:

$$V_{1i} = f_\beta(w_\beta, b_\beta, I_i) \; \forall \; i = (1, 2, 3 \ldots 60) \tag{5.4}$$

V_{1i} is output feature vector for input image frame (I_i) of a video, i varies from 1 to 60 as each video has 60 frames. f_β is the function representing CNN model with weights w_β and biases b_β.

Proposed model uses multi level noise induction at different stage of feature learning. This noise propels model to learn true data distribution and helps to avoid overfitting. Dropout is used as noise with random dropout value of 0.2. Output

after multiplying with binary noise vector becomes:

$$X_{1i} = V_{1i} \times \tilde{N}_{dropout} \tag{5.5}$$

here, $\tilde{N}_{dropout}$ represents the random binary vector for dropout of feature vector with a probability of 0.2. The output feature sequence (X_{1i}) is passed through the LSTM-1 to learn the temporal dependencies between features of different time frames of input video.

Hidden state of any memory cell (H_s) of LSTM can be defined as a function of previous hidden state (H_{s-1}) and input at current time step (X_s) (refer eq. 5.6).

$$H_s = f_\alpha(H_{s-1}, X_s) \;\; \forall \;\; X_s \in \mathbb{R}^{L \times T} \tag{5.6}$$

here, X_s is a real valued vector of dimensions ($L \times T$), L represents the length of a single vector and T represents the number of time steps in a given sequence. H_s can also be represented mathematically in term of parameters ($w_\alpha, u_\alpha, b_\alpha$) and input data ($X_s$) as shown in eq. 5.7.

$$H_s = H_f(w_\alpha X_s + u_\alpha H_{s-1} + b_\alpha) \; \forall \; s \in [1, T] \tag{5.7}$$

and input to LSTM is the output of TCNN, hence:

$$X_s = X_{1i} \tag{5.8}$$

Output feature vector (F_1) generated by LSTM-1 represents the probability of occurrence of feature vector (X_{t+1}) given the feature vector for previous frames ($X_t, X_{t-1}, \ldots\ldots$), which can be mathematically represented as:

$$F_1 = P(X_{t+1}|X_t, X_{t-1}, \ldots\ldots X_2, X_1) \tag{5.9}$$

By chain rule of probability, we can decompose equation for F_1 as:

$$F_1 = P(X_{t+1}|X_t) \times P(X_t|X_{t-1}) \times P(X_{t-1}|X_{t-2}) \ldots\ldots \tag{5.10}$$

The above equation can also be rewritten as:

$$F_1 = \prod_{t=1}^{T} P(X_{t+1}|X_t) \tag{5.11}$$

Similar to the first model of ensemble, second model (3D-CNN) also uses convolution based operation to extract spatial as well as temporal features from input video sequence. The operation can be mathematically represented as:

$$F_l(i,j,k) = \sum_m \sum_n \sum_o \sum_d F_{l-1}(i+m, j+n, k+o, d) \times K_l(m,n,o,d) \tag{5.12}$$

here, F_l represents the output feature map after convolution with 3 dimensional kernel (K_l) of size ($m \times n \times o$). d representer the depth (number of kernels in previous layer). It can be observed from eq. 5.12 and eq. 5.3 that both, output features and convolutional kernel, have an extra dimension in case of 3D-CNN. The final output of the 3D-CNN model can be represented as:

$$V_{2i} = f_\gamma(w_\gamma, b_\gamma, I_{video}) \tag{5.13}$$

here, V_{2i} is output feature vector for complete input video sequence (I_{video}). f_γ is the function representing 3D-CNN model with weights w_γ and biases b_γ. Architectural details of 3D-CNN model are given in table 5.5.

Dropout is also used at the output of 3D-CNN model to learn the true data distribution for a particular category with dropout probability of 0.2 (shown in eq.

5.14).

$$X_{2i} = V_{2i} \times \tilde{N}_{dropout} \tag{5.14}$$

The noisy output (X_{2i}) is then further passed through a fully connected layer to compress the feature vector. Final output (F_2) can be represented as a function of weight (w_θ), biases (b_θ) and input (X_{2i}) of fully connected layer (eq. 5.15).

$$F_2 = f_\theta(w_\theta, b_\theta, X_{2i}) \tag{5.15}$$

The third model in ensemble is used to model effect of arousal activity on the facial landmarks. Sequence of x coordinate of landmarks is given to LSTM-2, and sequence of y coordinates is given to LSTM-3. Output features of LSTM-2 and LSTM-3 are concatenated and passed through a fully connected layer for feature compression.

Coordinate of all 68 landmarks for 60 frames can be represented as $\mathscr{L}(x, y, n, f)$, where x and y represent the coordinate value in x and y direction. n represents the landmark number (varying from 1 to 68), and f is the frame number for the given set of coordinates. LSTM can not model two dimensional temporal sequence, therefore we separately model the activity in x and y direction for each of the landmark. LSTM-2 models the sequence of x coordinates of landmarks ($\mathscr{L}_t^x$) which can be mathematically represented as (used directly from previously derived eq. 5.11):

$$V_3 = \prod_{t=1}^{T} P(\mathscr{L}_{t+1}^x | \mathscr{L}_t^x) \tag{5.16}$$

here, V_3 is conditionally dependent feature vector of size 128 for x coordinates.

LSTM-3 models the sequence of y coordinates of landmarks ($\mathscr{L}_t^y$) which can be mathematically represented as:

$$V_4 = \prod_{t=1}^{T} P(\mathscr{L}_{t+1}^y | \mathscr{L}_t^y) \tag{5.17}$$

here, V_4 is conditionally dependent feature vector of size 128 for y coordinates.

Final output after feature compression through a fully connected layer can be represented as a function of weight (w_δ), biases (b_δ) and inputs (V_3 and V_4) of fully connected layer (eq. 5.18).

$$F_3 = f_\delta(w_\delta, b_\delta, V_3, V_4) \tag{5.18}$$

The final three outputs of three models are fused and passed trough a fully connected model to get a combined class score. The final output (M_{out}) of the composite architecture can be represented as:

$$M_{out} = f_\eta(w_\eta, b_\eta, F_1, F_2, F_3) \tag{5.19}$$

5.4.3 Model optimization

Categorical cross-entropy (shown in eq. 6.14) is used as loss function to train the end to end system.

$$L(y', y) = -\frac{1}{N}\sum_{i=0}^{N}[y_i log(y'_i) + (1 - y_i) log(1 - y'_i)] \tag{5.20}$$

here, $L(y', y)$ is the loss value, y'_i and y_i denote predicted and target labels, respectively and N represents the sample count in single mini-batch. In proposed work, mini-batch size of 10 is used to train the model.

We use RMSprop [151] optimizer to update model parameters according to value of loss function. If L_t is the loss value for a given minibatch and w_t is the model parameters; then, gradient term (g_t) for current minibatch is given by:

$$g_t = (1 - \gamma)\left[L'_t\right]^2 + \gamma g_{t-1} \tag{5.21}$$

here, γ represents the momentum term, which can be used to control the contribution of present and past gradients in the current updation of parameter. The parameter updation value (Δw_t) is given by following equation:

$$\Delta w_t = -\frac{\eta}{\sqrt{g_t + \epsilon}} \times L'_t \tag{5.22}$$

here, η is the learning rate of the algorithm and ϵ is the mathematical constant used to avoid non-integer value for parameter update. The final value of model parameter is sum of model parameter value and updation value (refer in eq. 5.23).

$$w_{t+1} = w_t + \Delta w_t \tag{5.23}$$

Out of total 1372 video samples generated after preprocessing of data, we use 1200 samples for training and 172 samples for testing. All the models are trained for 150 epochs with minibatch learning. Batch normalization is used on output of all the activation layers of all the models. Batch normalization helps in avoiding exploding and vanishing gradient problems.

5.5 Results and discussion

Each model in the ensemble of deep composite neural architecture (DCNA) is of paramount importance. We tested each modality independently to study their performance for the proposed problem. Table 5.6 shows test and training accuracies achieved by all of three basic models individually. Each of these models are trained using same training data pipeline described in Section 5.4.1. 3D-CNN model is able to achieve the highest accuracy out of three with test accuracy at 73.83.

Both the combinations of vision based models with landmark based model are able to significantly improve the performance (shown in table 5.7). Combination of TCNN-LSTM and Dual-LSTM models achieves same test accuracy as combination of 3D-CNN and Dual-LSTM models. Although, combination of TCNN-LSTM and Dual-LSTM achieve lower accuracy on 'Neutral' class and higher accuracy on both 'Like' and 'Dislike' categories. Both of these models achieve significantly lower

Table 5.6: Accuracy achieved by proposed composite neural architecture on test data.

Model	Parameters	Test Accuracy	Train accuracy
TCNN-LSTM	1,553,223	55.23	56.50
3D-CNN	5,198,479	73.83	99.25
Dual-LSTM	218,371	53.23	56.50

performance on neutral class because lack of arousal activity on face make it difficult to correctly identify the class.

Full DCNA has significantly higher performance when multi-level dropout is used in the architecture (shown in table 5.7). Elimination of dropout noise from model reduces test accuracy by 15.12%. Plot of training and test accuracy of CDNA with and without dropout are shown in fig. 5.8. Plot reveals that model starts to overfit training data in the absence of dropout at early stage of training. It is important to notice that data augmentation is used in both of these models.

Confusion matrices for all the four models (refer table 5.7) are shown in fig. 5.7. Although, both the model combinations (TCNN + Dual-LSTM and 3D-CNN + Dual-LSTM) are able to achieve equal test accuracy but confusion matrices of both the models provide in depth information about quality of results achieved by both the models. TCNN + Dual-LSTM (refer fig. 5.7a) has very biased behavior toward 'Like' category, and labels most of the neutral class samples in positive category along with actual ones. Whereas, 3D-CNN + LSTM model has significantly lower recall rate for neutral category (refer fig. 5.7b). Hence, even when test accuracy is same, 3D-CNN based model is able to learn much better feature representation than TCNN based model.

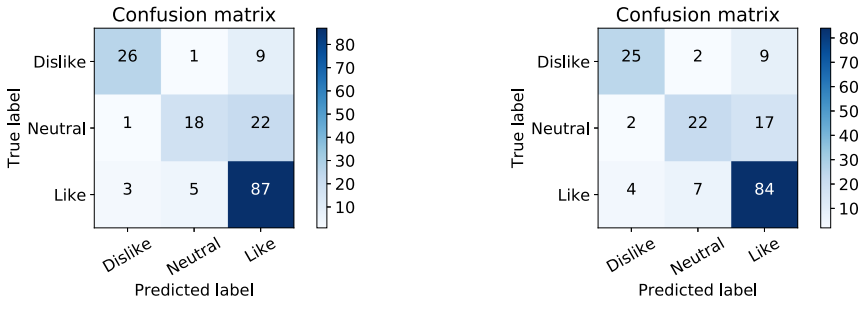
Similarly, confusion matrices for DCNA model with and without multi-level dropout are shown in fig. 5.7c and fig. 5.7d, respectively. Without induction of dropout in feature vectors, model is not able to learn the correct feature representations for 'Neutral' and 'Dislike' classes. The higher accuracy on positive class is because model classifies any arousal activity on face into positive class, and hence recall rate is very high for neutral and dislike classes. Dropout at multiple stage of feature extraction process removes some percentage of compressed feature representation, forcing model to learn more robust features.

Table 5.7: Analysis of performance of different combination basic models on final test performance against final DCNA.

Model	Parameters	Class accuracy			Average accuracy	Train accuracy
		Like	Neutral	Dislike		
TCNN-LSTM+ Dual-LSTM	1,796,103	91.57	43.90	72.22	76.16	92.16
3D-CNN+ Dual-LSTM	5,457,359	88.42	53.66	69.44	76.16	99.41
CDNA (without dropout)	7,028,371	86.31	39.02	41.67	65.69	99.33
CDNA (with multi-level dropout)	7,028,371	85.26	73.17	77.77	80.81	96.91

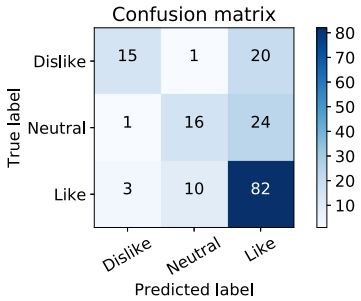
Table 5.8: Study of effect of different architectural variation on performance of DCNA.

Model	parameters	positive	neutral	negative	average accuracy	train accuracy
DCNA (low capacity)	7,028,371	85.26	73.17	77.77	80.81	96.91
DCNA (medium capacity)	13,443,599	85.26	56.10	75.00	76.16	94.74
DCNA (high capacity)	16,098,383	89.47	56.10	77.78	78.06	96.75
DCNA with ADAM optimizer	7,028,371	83.16	58.54	75.00	75.58	81.58
DCNA with SGD optimizer	7,028,371	84.21	46.34	72.22	72.67	99.08

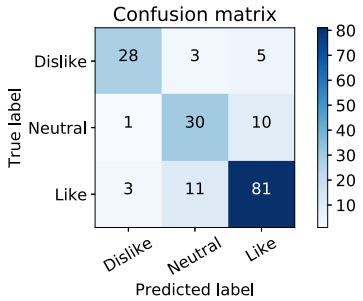


(a)

(b)



(c)



(d)

Figure 5.7: Confusion matrices for different models shown in table 5.7 (a) Confusion matrix for TCNN-LSTM + Dual-LSTM (b) Confusion matrix for 3D-CNN + Dual-LSTM (c) Confusion matrix for DCNA (without dropout) (d) Confusion matrix for DCNA (with multi-level dropout).

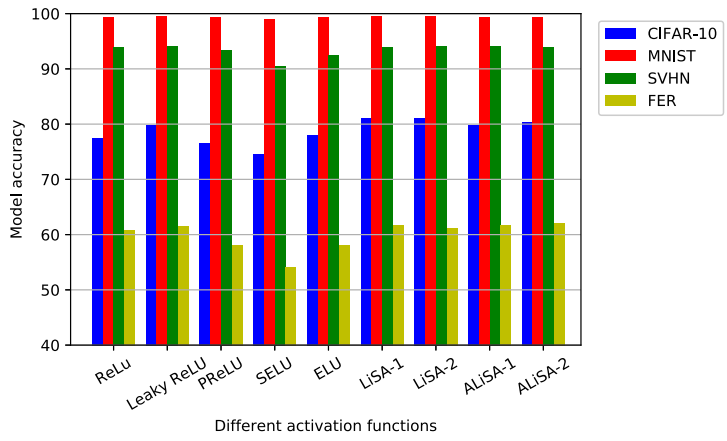


Figure 5.10: Bar chart showing the comparative analysis of test accuracies achieved by different models.

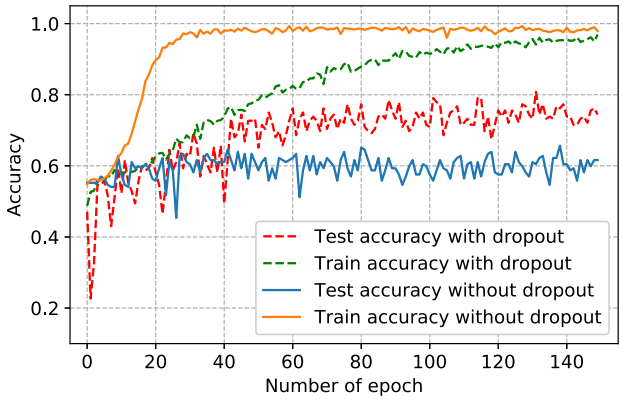


Figure 5.8: Plot of training and validation accuracy for DCNA with and without dropout.

Proposed model is tested with different kernel counts and optimizers (refer table 5.8). The DCNA (low capacity), in table 5.8, represents the model with kernel count shown in table 5.4 and table 5.5 for model TCNN and 3D-CNN, respectively. The model DCNA (medium capacity) and DCNA (high capacity) have 8 and 16 higher kernels in each layer than DCNA (low capacity), respectively. We can observe that increase in capacity of model actually reduces the test accuracy of the model. Intuitive explanation for this behavior is that higher dimensionality of parameter space makes it hard for gradient descent based optimizers to learn the correct parameter configuration. The proposed model is also tested with different optimizers. RMSprop [151], ADAM [152] and SGD [151] optimizer are able to achieve test accuracy of 80.81%, 75.58% and 72.67%, respectively (refer table 5.8).

Plot of validation accuracy of different variants of DCNA without the progress of training is shown in fig. 5.9. Although all the models show similar pattern in validation accuracy, the empirical selection of different parameters helps in improving the model performance. Barchart of the test accuracies achieved by all the test models is shown in fig. 5.10.

We also analyzed the internal layers of the proposed model to understand the learning characteristics of the model. Activation behavior of different layers of the appearance modeling network (TCNN) and dynamics modeling architecture are shown in fig. 5.11. Outputs of the 4th layer and 6th layer of 3D-CNN and TCNN models can be observed in the figure. Both the models show higher activation values in key facial areas like eyes, nose and lips. However, models were not provided with

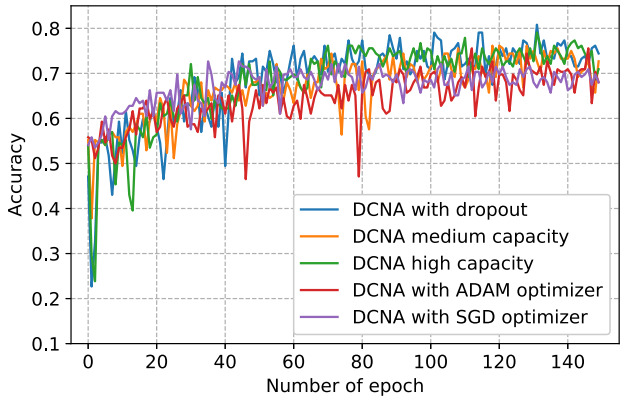


Figure 5.9: Plot of validation accuracy with progress of training for different variants of DCNA.

any specific information to lean the focus on these areas. These results employ that proposed architecture is able to correctly identify the key facial areas and extract the facial dynamics corresponding to each category.

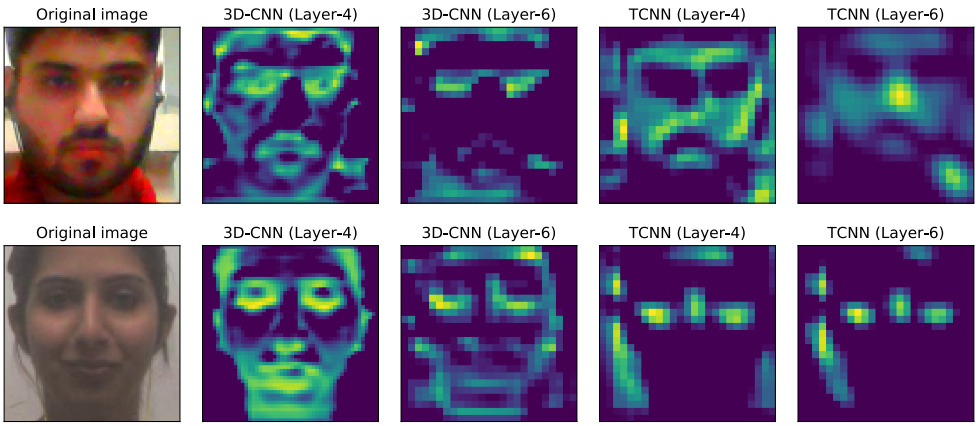


Figure 5.11: Analysis of activity of internal layers of TCNN and 3D-CNN models.

5.6 Conclusion

This work presents a novel framework to automatically generate users' response to the viewed commercial multimedia based on their facial expressions. For this experiment, volunteers were recruited to observe the videos of different genre. Facial

expressions of these observers are recorded against different genre of videos to cover different muscular activity. Subjects are also asked to provide their response against the watched multimedia, which is ~~divided~~ into three categories: Like, Neutral and Dislike.

A novel deep composite neural architecture is designed to learn the feature representation from the subject's face for this classification task. The proposed model uses ensemble of three different neural architectures to learn different modalities from the input data. The 3D convolutional neural network is used to extract spatio-temporal features from the input video. Similarly, long short term memory network is used to model the temporal dependencies between the features of different frames extracted by time distributed convolutional neural network. The third model (Dual-LSTM) is used to learn the sequential arousal activity observed in facial landmarks against each category.

Different combinations of all three basic models are studied to understand the learning behavior of each model. Final deep composite neural architecture is able to achieve highest accuracy of 80.81% with a multi-level dropout procedure. We presented the activity of internal layers of different models to provide an insight into learning behavior of these models.

6 Activation function for non-linear spatial dependencies

6.1 Introduction

In recent years, remarkable growth has been seen in deep neural networks (deep learning) after a long period of inactivity. This resurgence of neural networks has been largely due to increased computation power and emergence of large image datasets, like ImageNet, CIFAR-10, CIFAR-100, SVHN etc. Induction of Graphical Processing Units (GPU) in deep learning has also dramatically decreased the training time [153, 3]. Deep learning algorithms are now producing significantly higher performance than conventional algorithms in computer vision applications, such as image classification [90, 6, 3], image segmentation [91], object detection [85, 86, 87], image enhancement [88, 92], image retrieval [89, 93] and tracking [95].

Recent attention to deep learning has brought us better training strategies for bigger and deeper networks. These deep networks have substantially higher number of parameters, which dispenses much more learning power to models for learning better feature representations. Representation learning capability of Convolutional Neural Networks (CNN) is proportional to the depth of the model [4]. However, when depth of the model is increased, gradient flow in distant layer starts to vanish [154, 155]. Consequently, only deeper layers of the model learn representations and remaining layers remain useless. Models with higher number of parameters are also prone to over fitting. With emergence of better activation functions [7], regularization techniques [101], data augmentation [3, 5], pre-training [156] etc., difficulty of training deep neural networks has virtually resolved.

There has been considerable amount of work on activation functions. Among all non-saturating activation functions, Rectified Linear Unit (ReLU) [7] is most significant development contributing to success of deep convolutional neural network. Earlier,

Artificial Neural Networks (ANN) used saturating functions (like hyperbolic tangent and sigmoid) as activation of neurons [157, 158, 159, 160]. This makes it very difficult to back propagate the error if any unit is initialized in saturating range of the activation function. Both hyperbolic tangent and sigmoid activations have higher order non-linear behavior making them prone to over fitting [101, 161]. Hence, it is impossible to train neural network with multiple hidden layers. With the advent of ReLU, problems of over fitting and vanishing gradient got scraped. ReLU (refer fig. 3.8) is a linear function which clips the negative part of the input and forwards positive part unaffected. Therefore, gradient flow remains unhindered even with increased depth of the model.

There are two categories of activation functions. First type of approaches consider all the neurons and layers as identical and treat them the similar manner. These techniques make a sensible analysis of the task in hand and then a single fixed activation function is selected accordingly. Second category uses parametric strategies; where, parameters are learned either independently for layers or individual neurons. This class of functions provides different activation response either for each layer or for each neuron. Although these types of activation functions have performed better than standard activation functions in some applications, but their parametric nature increases number of trainable parameters. Increased number of parameters require more storage space and computational power.

In this chapter, we propose a novel Linearized Sigmoidal Activation (LiSA) function, which holds the advantages of both saturating and non-saturating activations. It shows segment wise linear behavior similar to non-saturating activation functions. But instead of modeling activity of neurons with a completely linear function, LiSA makes smaller non-linear range segments; which provides LiSA capability to learn higher order non-linear transformations similar to saturating activation functions. This non-linearity is of paramount importance while modeling higher complexity data. Neural network acquires much higher potential to learn non-linear representation, when all data range segments are provided with distinctive activation coefficient. LiSA (fig 6.1) exhibits linear behavior for middle range segment of input (linear activity region), hence data passes unaffected (similar to ReLU); whereas input values with high variance (value outside linear activity region) are suppressed. In spite of suppressing characteristics, units do not saturate even when data shows high variance, and the flow of gradient remains unhindered.

All the activation functions presented in the section 3.4 focus on the negative part of the signal. However, we propose a novel function which not only takes negative part in account but also encapsulates non-linear nature of the mutual relation of neurons.

6.2 Motivation

Before introducing our novel activation function, we discuss the basic flaw and limitations of the most commonly used activation functions (non-linearities) in deep neural networks. All recent activation functions are derived from ReLU and inherit all its basic behavioral characteristics (like non-saturating nature). Theoretically, ReLU is not non-linear function despite being actively used in most of applications, as it models only the linear relationship between data. To elaborate this, we consider an example of simple perceptron with weight matrix (W) and bias (B). For input (X), output is given by:

$$Y = WX + B \tag{6.1}$$

Output (Z) after applying non-linearity is given by

$$Z = f(Y) = f(WX + B) \tag{6.2}$$

We can rewrite this as

$$Z = Y \circ I = WX \circ I + B, \text{ where } \quad I^{m \times n} \in \{0, 1\} \tag{6.3}$$

here, m and n are dimensions of the output matrix (Y). I is a $m \times n$ binary matrix and its values are given by following equation

$$I = \begin{cases} 1 & \text{if } y > 0 \\ 0 & \text{if } y \leq 0 \end{cases} \tag{6.4}$$

here y represents the elements in the output matrix (Y).

Above discussion proves that ReLU is a linear function and clips certain part of the signal either randomly or based on gradient (fig 6.7e). The random clipping happens due to weight initialization. Output of some neurons fall in negative region if weights of neuron are negative. Hence, that neuron never receives any gradient and stays in same state. Second reason for clipping is gradient flow. If final score is invariant to activity of a neuron then gradient tries to suppress the activity by moving output of neuron into negative region.

A simple experiment is used to visualize the above judgment. In this experiment, we replaced ReLU with linear activation in the model and observe its effect on model performance. The results (table 6.1) show very minor decrease in test accuracy. On CIFAR-10 and MNIST datasets, ReLU improves model accuracy only by 1.69% and 0.61%, respectively. Hence, the only major advantage that ReLU provides is sparse activation. To gain non-linear modeling capacity, activation functions, like PReLU, use channel wise non-linear behavior. But still, modeling capacity of the activation function remains same in all the individual channels. In other words, although different activation slopes are used for the negative part, but structure of activation function remains the same. We rectify above discussed limitation of activation functions by proposing a range-wise non-linear activation function, which is yet immune to problems of saturating activation.

Table 6.1: Effect of changing ReLU activation with linear activation on model accuracy with CIFAR-10 and MNIST datasets.

Activation	Accuracy on CIFAR-10 (%)	Accuracy on MNIST (%)
ReLU	72.82	99.41
Linear activation	71.13	98.80
Difference (Δ)	1.69	0.61

6.3 Proposed method

6.3.1 Linearized sigmoidal activation

We present a novel activation function that provides non-linear behavior according to the segments of data range. Instead of considering data into only two segments

(positive or negative), proposed linearized sigmoidal activation (LiSA) function divides data range into multiple segments. The data within a single range segment hold linear relation, whereas the data points from different range segments hold a non-linear relationship. In other words, all the data points falling in the single range segment (e.g. within linear, positive or negative activity region) will hold a linear association. Hence, model can exploit wide range of linear and non-linear structural associations in data. Above mentioned behavior brings qualities of saturating and non-saturating activation functions into LiSA. Similar to non-saturating activation function, LiSA provides a smooth gradient flow even in models with much higher depth and hence does not suffer from vanishing gradient problem. On the other hand, LiSA is able to model higher order non-linear data associations because all the segments exhibit different activation behavior.

Proposed linearized sigmoidal activation function is presented in eq. 6.5 with corresponding visual representation in fig. 6.1. Where, α_1 and α_2 are empirically identified slope coefficients, and y and $f(y)$ are input and output of the layer, respectively.

$$f(y) = \begin{cases} \alpha_1 y - \alpha_1 + 1, & \text{if } 1 < y < \infty \\ y, & \text{if } 0 \leq y \leq 1 \\ \alpha_2 y, & \text{if } -\infty < y < 0 \end{cases} \tag{6.5}$$

The function considers input data range in three activity regions:

- Positive activity region ($1 < y < \infty$),
- Linear activity region ($0 \leq y \leq 1$), and
- Negative activity region ($-\infty < y < 0$)

Output in negative activity region is controlled by negative slope coefficient (α_2) and output for positive activity region is controlled by positive slope coefficient (α_1). The linear activity region has direct input to output mapping.

The proposed activation function shows interesting behavior traits as it can take shape of most of the common activation functions. For example, if the coefficients in eq. 6.5 are set as follows: $\alpha_1 = 1$ and $\alpha_2 = 0$, the proposed function transforms into ReLU activation function (eq. 6.6).

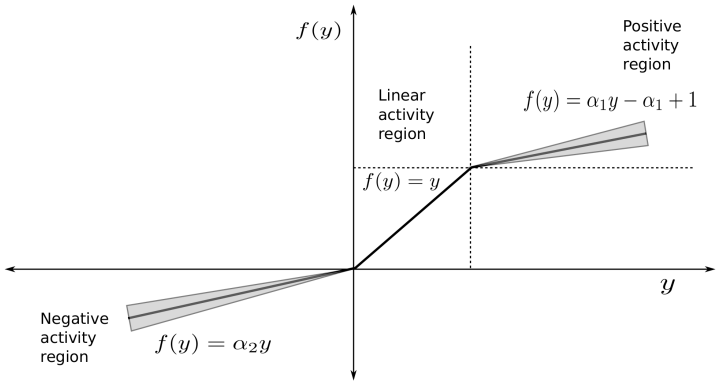


Figure 6.1: Proposed linearized sigmoidal activation function.

$$f(y) = \begin{cases} y, & \text{if } 1 < y < \infty \\ y, & \text{if } 0 \leq y \leq 1 \\ 0, & \text{if } -\infty < y < 0 \end{cases}$$

$$= \begin{cases} y, & \text{if } 0 \leq y \leq \infty \\ 0, & \text{if } -\infty < y < 0 \end{cases} \tag{6.6}$$

Similarly, if slope coefficients of proposed activation function are set as: $\alpha_1 = 1$ and $\alpha_2 = a$. Then, the propose activation function takes the shape of Leaky ReLU activation function (refer eq. 6.7).

$$f(y) = \begin{cases} y, & \text{if } 1 < y < \infty \\ y, & \text{if } 0 \leq y \leq 1 \\ ay, & \text{if } -\infty < y < 0 \end{cases}$$

$$= \begin{cases} y, & \text{if } 0 \leq y \leq \infty \\ ay, & \text{if } -\infty < y < 0 \end{cases} \tag{6.7}$$

6.3.2 Adaptive linearized sigmoidal activation

This section presents Adaptive Linearized Sigmoidal Activation (ALiSA) function. ALiSA uses trainable slope parameters which can adjust themselves to required value according to task in hand. These trainable slope parameters are trained alongside the model parameters during training process.

The basic structure of ALiSA remains same as LiSA. Hence, same figure can be used to represent the function (refer fig. 6.1). However, mathematical formulation is slightly different and is shown in eq. 6.8.

$$f(y) = \begin{cases} \alpha_1^k y^k - \alpha_1^k + 1, & \text{if } 1 < y < \infty \\ y^k, & \text{if } 0 \leq y \leq 1 \\ \alpha_2^k y^k, & \text{if } -\infty < y < 0 \end{cases} \tag{6.8}$$

here, α_1^k represents the channel-wise positive slope coefficient and α_2^k represents the channel-wise negative slope coefficient. y^k is the input to the activation function and k is number of channels. $f(y)$ is output of the activation function.

Similar to LiSA, adaptive linearized sigmoidal activation function can also manifest a wide range of functional structures. For example, if coefficient in eq. 6.8 are set as: $\alpha_1^k = 1$ and $\alpha_2^k = a^k$. Then, proposed activation function transforms itself into parametric rectified linear unit (refer eq. 6.9).

$$\begin{aligned} f(y) &= \begin{cases} y^k, & \text{if } 1 < y < \infty \\ y^k, & \text{if } 0 \leq y \leq 1 \\ a^k y^k, & \text{if } -\infty < y < 0 \end{cases} \\ &= \begin{cases} y^k, & \text{if } 0 \leq y \leq \infty \\ a^k y^k, & \text{if } -\infty < y < 0 \end{cases} \end{aligned} \tag{6.9}$$

During backpropagation, gradients through proposed activation function can also be calculated segment-wise. LiSA has fixed slope coefficients, hence gradient through

LiSA with respect to its input (y) can be given by eq. 6.10. Gradient for both negative and positive segments will remain different if the condition ($\alpha_1 \neq \alpha_2$) is satisfied.

$$\frac{df(y)}{dy} = \begin{cases} \alpha_1, & \text{if } 1 < y < \infty \\ 1, & \text{if } 0 < y < 1 \\ \alpha_2, & \text{if } -\infty < y < 0 \end{cases} \tag{6.10}$$

In case of ALiSA, both of the model weights as well as the layer parameters are trainable, hence we calculate the gradients with respect to input as well as activation slope coefficients. Gradient with respect to the input is given by eq. 6.11. The gradients with respect to positive and negative slope coefficients are given by eqs. 6.12 and 6.13.

$$\frac{\partial f(y)}{\partial y^k} = \begin{cases} \alpha_1^k, & \text{if } 1 < y < \infty \\ 1, & \text{if } 0 < y < 1 \\ \alpha_2^k, & \text{if } -\infty < y < 0 \end{cases} \tag{6.11}$$

$$\frac{\partial f(y)}{\partial \alpha_1^k} = \begin{cases} y^k - 1, & \text{if } 1 < y < \infty \\ 0, & \text{if } 0 < y < 1 \\ 0, & \text{if } -\infty < y < 0 \end{cases} \tag{6.12}$$

$$\frac{\partial f(y)}{\partial \alpha_2^k} = \begin{cases} 0, & \text{if } 1 < y < \infty \\ 0, & \text{if } 0 < y < 1 \\ y^k, & \text{if } -\infty < y < 0 \end{cases} \tag{6.13}$$

6.4 Experiment

6.4.1 Experimental setup

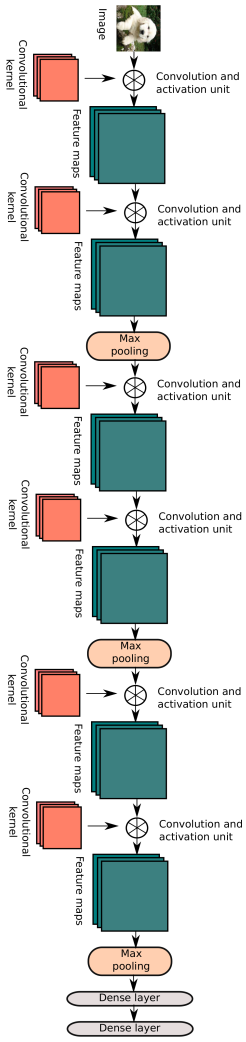


Figure 6.2: CNN architecture used for testing the performance of proposed activation functions against the state of the art activation functions.

We use VGGNet [4] based CNN architecture to test the proposed activation functions (shown in fig. 6.2). Eight layer VGG architecture is used with six convolutional layers and two dense layers. First and second convolutional layers contain 32 kernels. Kernel count for third and fourth layer is 48 and last two convolutional layers have 64 kernels. Kernel size of all convolutional layers is (3×3). Model architecture

and training condition are kept same in all the experiments. Weight initializations for the models are also kept same for comparative performance analysis.

Performance of proposed activation functions is evaluated on following benchmark datasets: CIFAR-10 [162], MNIST [163], Street View House Number (SVHN) [164], Facial Expression Recognition (FER-2013) [117]. Comparative analysis is conducted against state of the art activation functions: Rectified Linear Unit (ReLU), leaky Rectified Linear Unit (Leaky ReLU), Parametric Rectified Linear Unit (PReLU), Exponential Linear Unit (ELU) and Scaled Exponential Linear Unit (SELU).

For uniform test scenario, categorical cross-entropy (eq. 6.14) function is used as loss function for all the models, given by following equation.

$$L(y', y) = -\frac{1}{N}\sum_{i=0}^{N} [y_i log(y'_i) + (1 - y_i) log(1 - y'_i)] \tag{6.14}$$

where, $L(y', y)$ is the loss function, y_i and y'_i denote actual and predicted labels, respectively. N is total number of samples per mini-batch.

RMSprop [165] algorithm is used in all the experiments for gradient backpropagation. If L_t is the loss value for a particular example and θ_t is the parameters' value at time t. Then, mean square term (r_t) is given by eq. 6.15. Where, γ is the momentum term and is used to control the contribution of past gradients values, and L'_t denotes derivative of loss with respect to parameters (θ_t).

$$r_t = (1 - \gamma) \left[L'_t(\theta_t)\right]^2 + \gamma r_{t-1} \tag{6.15}$$

Actual update gradient (v_{t+1}) is calculated using eq. 6.16. Scaling of current gradient $(L'_t(\theta_t))$ with the mean square term (r_t) provides smoother and uniform gradients.

$$v_{t+1} = \frac{\alpha}{\sqrt{r_t}} L'_t(\theta_t) \tag{6.16}$$

here, α is the learning rate of optimization function.

For set of parameters defined by θ_t at time t, the next set of values (θ_{t+1}) are given by:

$$\theta_{t+1} = \theta_t - v_{t+1} \tag{6.17}$$

6.4.2 Analysis of linearized sigmoidal activation

Proposed activation function (LiSA) has two hyper-parameters: α_1 and α_2 (refer eq. 6.5). Choice of α_1 and α_2 is based on empirical evaluation. In experiments, it has been observed that best value of LiSA parameters lies between 0.15 and 0.25, generally. The exact value for parameters can only be identified using heuristic search method for the selected application and model architecture.

Activation behavior of LiSA has been tested for two activation scenarios. In first case, both positive and negative slope coefficients are assigned same value (eq. 6.18). It transforms eq. 6.5 into eq. 6.19.

$$\alpha_1 = \alpha_2 = \alpha \tag{6.18}$$

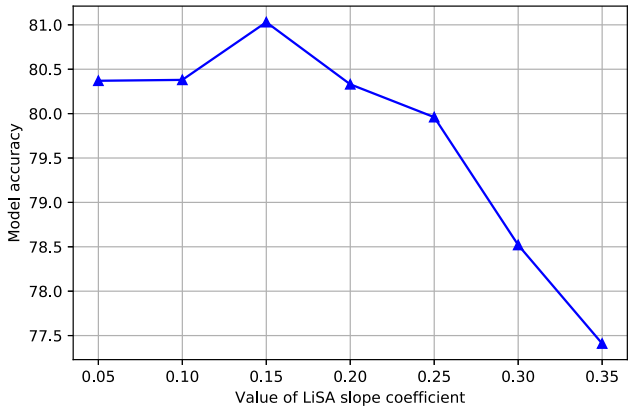
$$f(y) = \begin{cases} \alpha y - \alpha + 1, & \text{if } 1 < y < \infty \\ y, & \text{if } 0 \leq y \leq 1 \\ \alpha y, & \text{if } -\infty < y < 0 \end{cases} \tag{6.19}$$

In eq. 6.19, optimum value of α can be found using heuristic search through single parameter space. To analyze the effect of slope coefficient of activation layer on the performance of convolutional neural network model, we conduct trials for slope value (α) from 0.05 to 0.35 range while keeping all the other training parameters constant. The model performance is evaluated at a step size of 0.05 for slope coefficient. The performance statistics (validation loss, training loss and validation accuracy) of a CNN model (shown in fig 6.2) with different values of α are presented in table 6.2.

The relationship between the slope coefficients and model accuracy is plotted in fig. 6.3. From fig. 6.3 and table 6.2, we can observe that model accuracy initially

Table 6.2: Analysis of effect of slope coefficient's values in single slope LiSA based model on validation loss, training loss and validation accuracy.

LiSA parameter (α)	Validation loss	Training loss	Validation accuracy (%)
0.05	1.062	1.513	80.37
0.10	1.065	1.492	80.38
0.15	1.039	1.444	81.03
0.20	1.011	1.453	80.33
0.25	0.992	1.434	79.96
0.30	0.999	1.412	78.52
0.35	0.991	1.408	77.41



S

Figure 6.3: Plot showing the relation between the slope coefficient (α) in single slope LiSA and validation accuracy achieved by the model.

increases with increase in the slope coefficient value, but starts to decrease after certain value. It is observed in experiments that best value of parameter in single slope LiSA lies around 0.15.

In second case, LiSA is tested with standard structure where both positive (α_1) and negative (α_2) slope coefficients are considered to be different. Effect of both the slope coefficients on model accuracy is shown in table 6.3 and table 6.4. Heuristic search approach is used on both the parameters in succession. Table 6.3 shows the performance of model when positive slope coefficient is swept, while keeping negative slope coefficient constant at 0.2. Similarly, effects of negative slope coefficient on model performance with constant positive slope coefficient are presented in 6.4. The best parameter value from table 6.3 is selected as positive slope coefficient for table 6.4 tests.

Plot of model accuracy against both the slope coefficients is shown in fig 6.4. Both the parameters show behavior similar to single slope LiSA. Initially accuracy increases with increase in parameter value, but after a threshold the continuous fall is observed in accuracy. Both the parameters have different optimum values. The model achieves highest accuracy with positive slope coefficient at 0.25 and negative slope coefficient at 0.15. Highest accuracy achieved by the dual slope LiSA is 81.12%. If we compare the performance of both single and dual slope LiSA functions, the dual slope LiSA attains 0.08% higher accuracy than single slope version. In other words, providing different activation behavior to positive and negative activity regions improves model performance.

Table 6.3: Analysis of effect of variation in positive slope coefficient (α_1) on model performance with negative slope coefficient (α_2) at a constant value of 0.2.

LiSA parameter		Validation loss	Training loss	Validation accuracy (%)
α_1	α_2			
0.05	0.2	1.007	1.458	79.80
0.10	0.2	1.026	1.453	79.72
0.15	0.2	1.017	1.459	80.15
0.20	0.2	1.011	1.453	80.33
0.25	0.2	1.020	1.449	80.98
0.30	0.2	0.992	1.449	80.24
0.35	0.2	0.988	1.452	79.98

Table 6.4: Analysis of effect of variation in negative slope coefficient (α_2) on model performance with positive slope coefficient (α_1) at a constant value of 0.25.

LiSA parameter		Validation loss	Training loss	Validation accuracy (%)
α_1	α_2			
0.25	0.05	1.088	1.525	80.30
0.25	0.10	1.061	1.476	80.23
0.25	0.15	1.013	1.445	81.12
0.25	0.20	1.020	1.450	80.98
0.25	0.25	0.992	1.434	79.96
0.25	0.30	0.988	1.414	78.95
0.25	0.35	0.987	1.409	77.99

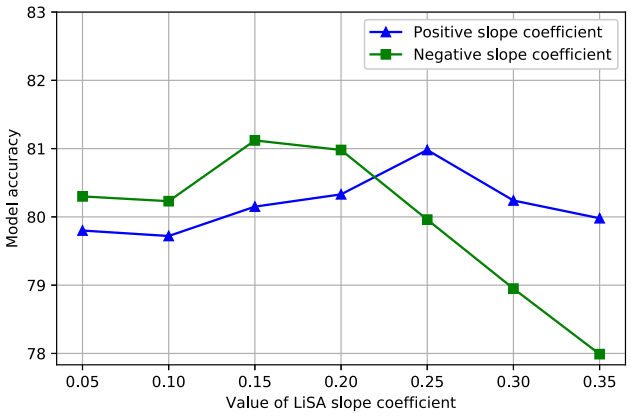


Figure 6.4: Plot showing the relation between the value of slope coefficients (α_1 and α_2) and validation accuracy achieved by the model.

6.4.3 Analysis of adaptive linearized sigmoidal activation

The proposed ALiSA function is also tested under two scenarios (similar to LiSA). In first case, trainable slope coefficients of positive and negative activity regions are considered equal. Therefore, same parameter is trained according to activity in positive as well as negative activity regions. This can be formalized as below:

$$\alpha_1^k = \alpha_2^k = \alpha^k \tag{6.20}$$

Substitute eq. 6.20 in eq. 6.8, the equation for single slope ALiSA becomes:

$$f(y) = \begin{cases} \alpha^k y^k - \alpha^k + 1, & \text{if } 1 < y < \infty \\ y^k, & \text{if } 0 \leq y \leq 1 \\ \alpha^k y^k, & \text{if } -\infty < y < 0 \end{cases} \tag{6.21}$$

In second case, standard ALiSA function is used with dual slope coefficients (refer eq. 6.8). Both the slope coefficients are trained independently during model training. Gradients are given by eqs. 6.12 and 6.13 for dual slope ALiSA. Parameters of both single and dual slope ALiSA are initialized to 0.2 at the start of training.

The accuracy achieved by CNN models, trained using both versions of ALiSA, is

Table 6.5: Accuracy achieved by ALiSA with single and dual trainable slope coefficients. ALiSA-1 represents the CNN model with single parameter activation and ALiSA-2 represents model with dual parameter activation.

Activation	Validation loss	Training loss	Validation accuracy (%)
ALiSA-1	1.015	1.442	79.77
ALiSA-2	0.998	1.449	80.34

shown in table 6.5. Single slope ALiSA achieves accuracy of 79.77%. Dual slope ALiSA attains 80.34% providing improvement of 0.57% over single slope ALiSA. Binding of the gradients for positive and negative activity regions in single slope ALiSA is the reason for lower performance. Parameter update equation for the single slope ALiSA can be written as:

$$\alpha_{t+1} = \alpha_t - \frac{\partial f(y)}{\partial \alpha} \tag{6.22}$$

$$\text{where, } \frac{\partial f(y)}{\partial \alpha} = \begin{cases} y^k - 1, & \text{if } 1 < y < \infty \\ 0, & \text{if } 0 < y < 1 \\ y^k, & \text{if } -\infty < y < 0 \end{cases} \tag{6.23}$$

From eq. 6.23, it can be observed that due to presence of single trainable coefficient for both positive and negative part of input, non-linearity in positive activity region of ALiSA affects the activity characteristics of negative activity region and vice versa. Due to this mutual dependence of positive and negative activity regions, higher correlation in positive activity region affects the activation characteristics of the layer for negative data components as well, resulting in decrease of model accuracy. Although, single coefficient ALiSA has advantage of lower memory space requirement to store the coefficients, and lower computational complexity for gradient calculation during back-propagation.

Layer by layer learned coefficients for single and dual slope ALiSA are shown in table 6.6. ALiSA-1 and ALiSA-2 denote the single and dual slope ALiSA in table 6.6. CNN architecture used in the experiments has seven activation layers. All the layers in the model learn very different coefficients from each others depending on

data. When both models use same training data and training condition, learned coefficients in single parameter ALiSA are also very different from the parameters learned by dual parameter ALiSA even.

Table 6.6: Layer-wise analysis of the trainable parameter values of ALiSA with single and dual trainable parameter.

Activation layer	ALiSA-1	ALiSA-2	
	α^k	α_1^k	α_2^k
Activation-1	0.055	0.131	-0.136
Activation-2	0.088	0.095	0.422
Activation-3	0.019	0.025	0.563
Activation-4	0.043	0.055	0.546
Activation-5	0.010	0.003	1.023
Activation-6	0.137	0.151	0.714
Activation-7	0.018	0.005	0.086

Single parameter ALiSA shows very low slope coefficient value in all the layers except layer-6. From this we can infer that model is trying to clip off any sample values beyond the linear range of the model. In other words, the layer has started to behave more like ReLU with positive threshold clipping.

In dual parameter ALiSA, positive as well as negative slope coefficients show very high variation. The negative slope coefficients have higher value than positive slope coefficients of corresponding layers. Higher value of negative slope coefficients indicates higher contribution of negative data samples in the final score. In general, positive slope coefficients (α_1^k) of initial layers have higher values than the deep layers.

6.5 Results and discussion

6.5.1 CIFAR-10

CIFAR-10 dataset is a benchmark dataset most commonly used in testing deep learning algorithms. Dataset contains 32×32 sized color images, organized in ten classes. The dataset has 50,000 training images and 10,000 test images. Data is normalized using global contrast normalization to limit the input range. Performance of the proposed activation functions against the state of the art functions is presented in

table 6.7. Performance evaluation tests are conducted using the architecture shown in fig. 6.2. Leaky ReLU activation has the highest performance between all state of the art methods with 79.75% classification accuracy. All proposed activation functions are able to surpass this performance. Dual parameter LiSA (LiSA-2) achieves highest accuracy with 81.12% providing improvement of 1.37%.

We also test the proposed activation function against ReLU on different model architectures to test the scalability. Proposed function is tested on two CNN models with different depths (8 layer and 6 layer). Function is also tested with and without dropout. Results of the tests are shown in table 6.8. Single slope LiSA layer is used against ReLU in all of these tests. LiSA based model perform better than ReLU based model in all the test scenarios. In 6 layer and 8 layer CNN model, LiSA based model has 1.2% and 3.2% higher accuracy. Proposed activation function achieves 3.6% higher classification accuracy against ReLU when dropout of 0.4 is used. It reflects better modeling ability of proposed activation function under different architectural conditions.

Table 6.7: Classification accuracy of proposed activation functions against state of the art methods on CIFAR-10 dataset.

Activation function	Parameter value	Accuracy
ReLU [7]	Nil	77.43
Leaky ReLU [8]	0.2	79.75
PReLU [107]	Nil	76.43
SELU [166]	0.2	74.58
ELU [109]	Nil	77.89
LiSA-1	0.15	81.03
LiSA-2 *	0.25, 0.15	81.12
ALiSA-1	Nil	79.77
ALiSA-2	Nil	80.34

6.5.2 MNIST

MNIST is a binary image dataset of digits. It contains ten classes and have images of size 28×28 with 60,000 training and 10,000 test samples. The dataset is preprocessed with global contrast normalization to limit the input range. No data augmentation is used in the experiments. Due to low complexity of dataset, all models are able to achieve higher accuracy on this dataset.

Table 6.8: Different CNN architectures for testing scalability of the proposed activation function against structural variations.

Model	Activation	Dropout	α	Accuracy
CNN (6 layer)	ReLU	Nil	Nil	72.82
CNN (6 layer)	LiSA	Nil	0.2	74.02
CNN (8 layer)	ReLU	Nil	Nil	72.85
CNN (8 layer)	LiSA	Nil	0.2	76.05
CNN (8 layer)	ReLU	0.4	Nil	77.43
CNN (8 layer)	LiSA	0.4	0.2	81.03

Leaky ReLU again acquires the highest accuracy of 99.42% among the base activation functions (refer table 6.9). The proposed dual slope LiSA (LiSA-2) surpasses Leaky ReLU by gaining the accuracy of 99.53%. Single slope LiSA (LiSA-1) also achieves the approximately same accuracy as Leaky ReLU. Although, their adaptive versions (ALiSA-1 for single slope ALiSA and ALiSA-2 for dual slope ALiSA) fall little behind in performance. In our analysis, we found that trainable parameters of ALiSA take little more time to find optimum coefficient values for the best results. Since Adaptive versions (ALiSA) based models are also trained for equal number of epochs as other models in the tests, they too fall slightly short in performance.

Table 6.9: Classification accuracy of proposed activation functions against state of the art methods on MNIST dataset.

Activation function	Parameter value	Accuracy
ReLU [7]	Nil	99.39
Leaky ReLU [8]	0.2	99.42
PReLU [107]	Nil	99.40
SELU [166]	0.2	98.97
ELU [109]	Nil	99.29
LiSA-1	0.15	99.41
LiSA-2 *	0.175, 0.25	99.53
ALiSA-1	Nil	99.36
ALiSA-2	Nil	99.38

6.5.3 SVHN

Street View House Number (SVHN) is a dataset of digits similar to MNIST, but it contains real world images of digits from house numbers marked on the streets.

SVHN dataset contains 32×32 color images of digits with ten classes. It is divided into 73,527 training and 26,032 test samples. There are 531,131 additional images. We only used training and test data in our experiments. Image may contain multiple digits in single image. The target of task is to classify image based on digit in the center.

On SVHN, proposed dual slope LiSA activation achieves highest classification accuracy (94.07%) closely followed by single slope ALiSA (ALiSA-1) and Leaky ReLU with 94.02%. Table 6.10 shows the accuracy achieved my all the activation functions.

Table 6.10: Classification accuracy of proposed activation functions against state 0f the art methods on Street View House Number dataset.

Activation function	Parameter value	Accuracy
ReLU [7]	Nil	93.93
Leaky ReLU [8]	0.2	94.02
PReLU [107]	Nil	93.33
SELU [166]	0.2	90.37
ELU [109]	Nil	92.51
LiSA-1	0.15	93.86
LiSA-2 *	0.25, 0.15	94.07
ALiSA-1	Nil	94.02
ALiSA-2	Nil	93.83

6.5.4 FER-2013

Facial expression recognition dataset (FER-2013) contains images of human faces. Faces represents seven human emotion i.e. angry, disgust, fear, happy, sad, surprise and neutral. Dataset contains 48×48 size gray scale images. There are 28,709 images in training set and 3,589 images in both validation and test set. In our experiments, we use training and validation sets for training, and test set for testing.

The adaptive proposed activation (ALiSA) based models deliver the highest accuracy on this dataset (refer table 6.11). Highest accuracy (61.95%) is achieved by dual slope ALiSA (ALiSA-2) followed by single slope LISA and ALiSA at 61.63% and 61.61%, respectively. Leaky ReLU is able to provide the highest classification accuracy among the base models but lower than proposed models.

Table 6.11: Classification accuracy of proposed activation functions against state of the art methods on FER-2013 dataset.

Activation function	Parameter value	Accuracy
ReLU [7]	Nil	60.85
Leaky ReLU [8]	0.2	61.48
PReLU [107]	Nil	58.08
SELU [166]	0.2	53.99
ELU [109]	Nil	58.01
LiSA-1	0.15	61.63
LiSA-2	0.15, 0.25	61.17
ALiSA-1	Nil	61.61
ALiSA-2 *	Nil	61.95

6.5.5 Convergence and activation characteristics

Time taken by a model to converge is one of the very important traits of deep learning models. Fast convergence leads to lesser training time and lower power computation. Analysis of the convergence characteristics is shown in fig. 6.6. Rate of decay of training loss is much higher in proposed models in comparison to the state of the art models (refer fig. 6.6a).

Non-adaptive activation (LiSA-1 and LiSA-2) based models are able to maintain lowest loss value till the end of training process. Whereas, adaptive activation function (ALiSA-1 and ALiSA-2) observe small rise in loss towards the end. However, the validation loss (fig. 6.6b) of adaptive activation functions is the lowest of all. Similarly, adaptive activation functions are able to maintain higher validation accuracy (fig. 6.6c) than ReLU [7] and Leaky ReLU [8] based models.

Output maps of the activation functions also convey a significant amount of information about model's learning ability and learned features [120, 167]. Analysis of the activity in feature maps of different layers is shown in fig. 6.7. Output of activation functions for multiple kernels of first, fourth and sixth layers are selected to provide the insight on learning behavior of models. Layers are chosen based on their association with initial, middle and end sections of the model. Activity of kernels for non-adaptive single and dual parameter model are shown in fig. 6.7a and 6.7b, respectively. Similarly, activity of kernels of adaptive single and dual parameter models is shown in fig. 6.7c and 6.7d, respectively. For comparative analysis, activation maps of ReLU [7] and Leaky ReLU [8] based models are shown in fig. 6.7e

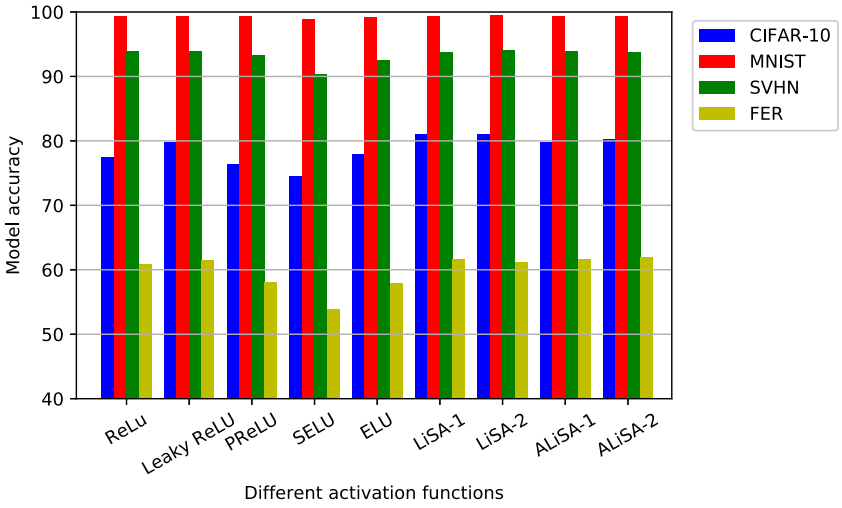


Figure 6.5: Collective bar chart showing test accuracy of proposed activation functions against state of the art on CIFAR-10, MNIST, SVHN and FER-2013 datasets, respectively.

and 6.7f, respectively.

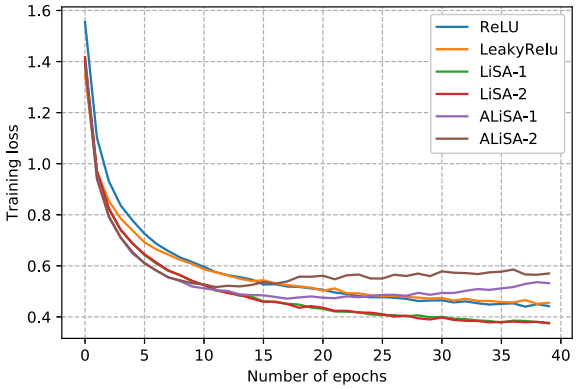
Activation maps of proposed activation functions cover wider spectrum and include outputs in all three regions of activity. After the analysis of all the activation maps for different images, we found that higher number of kernels of initial layers produce output in linear and negative regions of activity. But as we go deeper, more number of kernels show output activity in all three regions. This behavior can be used to infer that initial layer of the model learn linear patterns, deeper layers learn higher order non-linear patterns. On the other hand, output feature maps of ReLU activation function shows very sparse activity. Most of output maps for ReLU based model are zero and hence, these maps block the flow of gradient to all units. This can also be seen as a reason for observed dead kernels in ReLU based models (fig. 6.7e).

6.6 Conclusion

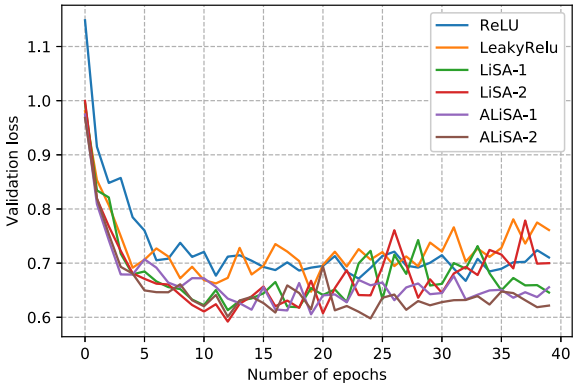
The proposed linearized sigmoidal activation function decomposes full range of data into small non-linear segments. Each segment holds distinct activation behavior. It helps to learn non-linear transformation for better feature representation. We investigated two different variants of proposed activation function. First activation function (LiSA) uses empirically evaluated hyper-parameters. Experimental eval-

uation shows that for best performance, positive and negative slope coefficients in dual slope LiSA should be set to 0.25 and 0.15, respectively. Whereas, single slope LiSA should preferably be used with coefficient value of 0.15. Second version (ALiSA) uses trainable slope coefficients, which are learned during the training process along side the model parameters. ALiSA is also tested in single slope and dual slope versions.

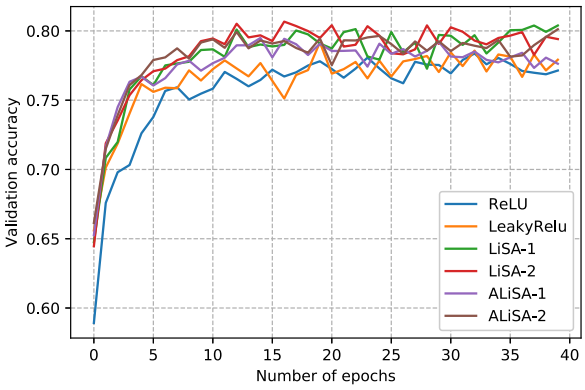
In experiments, initially we test the effect of different slope coefficients on the classification accuracy. Later, performances of proposed activation functions are tested on bench mark datasets: CIFAR-10, MNIST, SVHN and FER-2013. Dual slope LiSA is able to achieve highest accuracy on CIFAR-10, MNIST and SVHN datasets. On FER-2013, dual slope ALiSA provides the best results. Experiments demonstrate that proposed activation function performs better than other state of the art activation functions on all datasets.



(a) Plot of training loss.

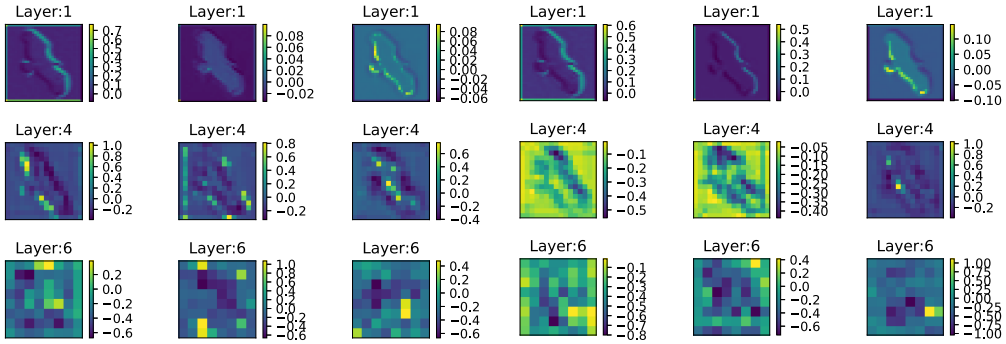


(b) Plot of validation loss.

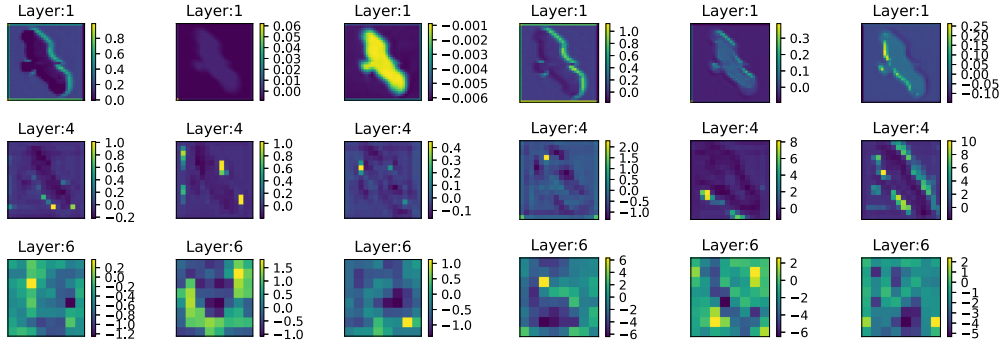


(c) Plot of validation accuracy.

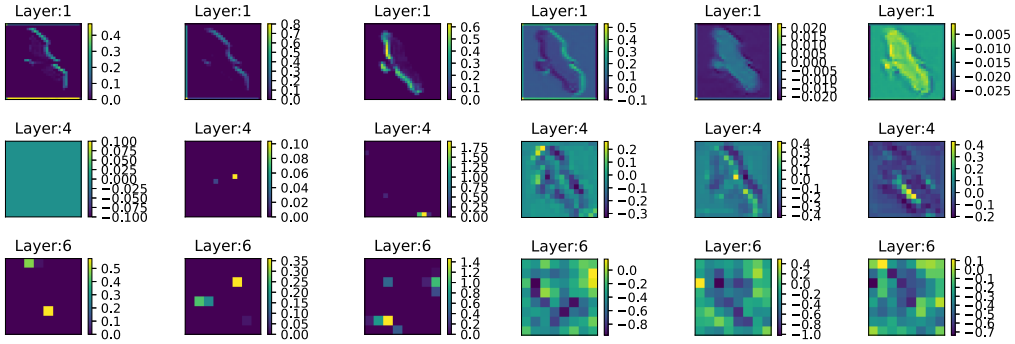
Figure 6.6: Plots showing rate of convergence of proposed activation based model against state of the art.



(a) Activity map for LiSA-1 activation function (b) Activity map for LiSA-2 activation function



(c) Activity map for ALiSA-1 activation function (d) Activity map for ALiSA-2 activation function



(e) Activity map for ReLU activation function (f) Activity map for Leaky ReLU activation function

Figure 6.7: Plots of the output of different activation functions based models for first, fourth and last convolutional layer of the model.

7 Conclusion and Future Scope

7.1 Conclusion

Facial expressions are very important component of our interaction with the outside world. Human expressions not only provide the internal emotional status of a person but also help in identifying the mental state and involuntary responses generated against any stimuli. Previous facial expression analysis methods involved hand crafted feature detectors like HOG, SOFT, LBP etc. These type of feature detector are inefficient at extracting robust features from non rigid object. It happens due to continuous variation in geometric structure of the facial muscles. Hence, in this work, we explored the representation learning based approaches for facial expression analysis. The main aim of this dissertation was to develop deep learning based novel solutions for facial expression analysis. In addition, this work also targets to develop novel and reliable commercial solutions for different application based on facial expression analysis.

This thesis has produced three novel solutions. Details of these proposed frameworks and deep learning solutions are as following:

7.1.1 Sentiment analysis from social media images

In recent years, online multimedia content has risen exponentially with the outburst of internet and social media. There are at present no known solutions deployed in these platforms to learn and analyze the sentiments represented by these images. We developed a system to learn the sentiment represented by an image based on its contents. The system contains two submodules: one for analyzing the facial expression of people in the image and other analyzed the general global feature of the image such as location and setup of image.

The first module segments faces from the image, and then deep convolutional neural network model is used to extract the low dimensional feature representation for these faces. Similarly, second module extracts the low level feature representation for the global features. A long short term memory network is used to learn the conditional probabilities of existence of these features. The network output is classified using a regression classifier. The system categorizes the predicted sentiment into three categories: positive, neutral and negative.

7.1.2 Multimedia likability prediction system

This dissertation also presents a novel system to predict the response of the viewer against shown multimedia by observing the facial expressions of the viewer. First we collected the dataset for this task. Volunteers were recruited to record their facial expression while they watched different type of videos. Then, subject was asked to manually provide the feedback against the watched video. Deep learning based ensemble architecture was used to model the relationship between viewer response and his/her recorded expressions.

Initially, frame rate of videos were lowered to reduce the computational requirements for the processing of media. The proposed architecture contains three deep learning based sub-modules to learn different modalities from the input data. Combination of time distributed convolutional neural network and long short term memory network is used to extract the spatio-temporal features from the input sequence of faces, extracted from the original recorded video. Another model called as 3D convolutional neural network is also used to extract the spatio-temopral features. It was observed that both of these models learned very different feature description for the input data. A third model was deployed to track and analyze the movement of facial landmarks. This module contained two LSTMs to learn the temporal pattern against each output category.

7.1.3 Activation function for non-linear spatial dependencies

Developing a model using deep learning architectures to model non-linear data is a very complex task. Higher order non linearities tend to induce problems like exploding and vanishing gradient, while linear model are incapable to model the higher

order relationships in the data. It is a well know fact that facial expression classification is a non-linear modeling problem due the very high intra-class variation of data. Hence, in this thesis we developed a novel activation function to model the nonlinear spatial dependencies in the image. The activation function has characteristics of both saturating and non saturating activation. The proposed activation function is divided into multiple segments, each have linear behavior inside its range, but non-linear relationship with other segments. This helps in increasing learning capacity of deep learning model. Multiple experiments were conducted to show that proposed function is able to achieve higher performance on tasks like facial emotion recognition.

7.2 Future scope

Facial expression analysis and emotion detection are very complex problems. Although, a lot of work is done in this field, but still there remain a lot of challenges that need to be addressed. This work tries to use deep learning based solution with two target in focus: (1) to generate better feature representation, (2) for exploring applications of facial expression analysis.

One of the major concerns for FER systems is the computational requirement. As these systems are generally deployed on mobile devices, which do not have enough computation power for deep learning algorithm. We need to develop more computationally efficient deep learning architectures. Additionally, large data requirement for the model training is also a major concern. Every new application requires a new and large dataset which can be time consuming and costly. Although new research is targeting the feature transferability in deep learning but there is still a long way to go.

Most of the work conducted in the emotion and facial expression analysis is under controlled environmental conditions. Most of the available datasets are developed in the labs and are not suitable for development of application for commercial use. We have already developed dataset for video response generation in uncontrolled setup. We also intend to develop more complex and realistic datasets for different FER based computer vision applications in future.

Bibliography

[1] P. Ekman, "An argument for basic emotions," *Cognition & emotion*, vol. 6, no. 3-4, pp. 169–200, 1992.

[2] S. Du, Y. Tao, and A. M. Martinez, "Compound facial expressions of emotion," p. 201322355, National Acad Sciences, 2014.

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," pp. 1097–1105, 2012.

[4] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[5] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, 2015.

[6] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[7] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," pp. 807–814, 2010.

[8] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," vol. 30, no. 1, p. 3, 2013.

[9] A. Dhall, J. Joshi, K. Sikka, R. Goecke, and N. Sebe, "The more the merrier: Analysing the affect of a group of people in images," in *Automatic Face and Gesture Recognition (FG), 2015 11th IEEE International Conference and Workshops on*, vol. 1, pp. 1–8, IEEE, 2015.

[10] R. A. Hinde, *Non-verbal communication*. Cambridge University Press, 1972.

[11] P. Ekman, *Telling lies: Clues to deceit in the marketplace, politics, and marriage (revised edition)*. WW Norton & Company, 2009.

[12] P. Bull, "State of the art: Nonverbal communication," *The Psychologist*, vol. 14, no. 12, pp. 644–647, 2001.

[13] P. Carrera-Levillain and J.-M. Fernandez-Dols, "Neutral faces in context: Their emotional meaning and their function," *Journal of Nonverbal Behavior*, vol. 18, no. 4, pp. 281–299, 1994.

[14] Z. Zeng, M. Pantic, G. I. Roisman, and T. S. Huang, "A survey of affect recognition methods: Audio, visual, and spontaneous expressions," *IEEE transactions on pattern analysis and machine intelligence*, vol. 31, no. 1, pp. 39–58, 2009.

[15] L. Maat and M. Pantic, "Gaze-x: Adaptive, affective, multimodal interface for single-user office scenarios," in *Artifical Intelligence for Human Computing*, pp. 251–271, Springer, 2007.

[16] A. Kapoor, W. Burleson, and R. W. Picard, "Automatic prediction of frustration," *International journal of human-computer studies*, vol. 65, no. 8, pp. 724–736, 2007.

[17] K. R. Scherer, T. Bänziger, and E. Roesch, *A Blueprint for Affective Computing: A sourcebook and manual*. Oxford University Press, 2010.

[18] P. Ekman, "Universals and cultural differences in facial expressions of emotion.," in *Nebraska symposium on motivation*, University of Nebraska Press, 1971.

[19] P. Ekman and E. L. Rosenberg, *What the face reveals: Basic and applied studies of spontaneous expression using the Facial Action Coding System (FACS)*. Oxford University Press, USA, 1997.

[20] M. A. Sayette, C. S. Martin, M. A. Perrott, J. M. Wertz, and M. R. Hufford, "A test of the appraisal-disruption model of alcohol and stress.," *Journal of Studies on Alcohol*, vol. 62, no. 2, pp. 247–256, 2001.

[21] A. C. d. C. Williams, "Facial expression of pain: an evolutionary account," *Behavioral and brain sciences*, vol. 25, no. 4, pp. 439–455, 2002.

[22] D. McDuff and R. el Kaliouby, "Applications of automated facial coding in media measurement," *IEEE Transactions on Affective Computing*, vol. 8, no. 2, pp. 148–160, 2017.

[23] M. Pantic, A. Pentland, A. Nijholt, and T. S. Huang, "Human computing and machine understanding of human behavior: a survey," in *Artifical Intelligence for Human Computing*, pp. 47–71, Springer, 2007.

[24] K. Anderson and P. W. McOwan, "A real-time automated system for the recognition of human facial expressions," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 36, no. 1, pp. 96–105, 2006.

[25] A. Moors, P. C. Ellsworth, K. R. Scherer, and N. H. Frijda, "Appraisal theories of emotion: State of the art and future development," *Emotion Review*, vol. 5, no. 2, pp. 119–124, 2013.

[26] R. D. Lane and L. Nadel, *Cognitive neuroscience of emotion*. Oxford University Press, 2002.

[27] R. R. Cornelius, *The science of emotion: Research and tradition in the psychology of emotions*. Prentice-Hall, Inc, 1996.

[28] W. G. Parrott, *Emotions in social psychology: Essential readings*. Psychology Press, 2001.

[29] S. Zhalehpour, O. Onder, Z. Akhtar, and C. E. Erdem, "Baum-1: A spontaneous audio-visual face database of affective and mental states," *IEEE Transactions on Affective Computing*, vol. 8, no. 3, pp. 300–313, 2017.

[30] H. Monkaresi, N. Bosch, R. A. Calvo, and S. K. D'Mello, "Automated detection of engagement using video-based estimation of facial expressions and heart rate," *IEEE Transactions on Affective Computing*, vol. 8, no. 1, pp. 15–28, 2017.

[31] M. Mehu and K. R. Scherer, "Emotion categories and dimensions in the facial communication of affect: An integrated approach.," *Emotion*, vol. 15, no. 6, p. 798, 2015.

[32] P. Ekman and W. V. Friesen, *Manual for the facial action coding system*. Consulting Psychologists Press, 1978.

[33] J. F. Cohn, "Foundations of human computing: facial expression and emotion," in *Proceedings of the 8th international conference on Multimodal interfaces*, pp. 233–238, ACM, 2006.

[34] R. Li, P. Liu, K. Jia, and Q. Wu, "Facial expression recognition under partial occlusion based on gabor filter and gray-level cooccurrence matrix," in

Computational Intelligence and Communication Networks (CICN), 2015 International Conference on, pp. 347–351, IEEE, 2015.

[35] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 1, pp. 886–893, IEEE, 2005.

[36] G. Zhao and M. Pietikainen, "Dynamic texture recognition using local binary patterns with an application to facial expressions," *IEEE transactions on pattern analysis and machine intelligence*, vol. 29, no. 6, pp. 915–928, 2007.

[37] H. Soyel and H. Demirel, "Facial expression recognition based on discriminative scale invariant feature transform," *Electronics letters*, vol. 46, no. 5, pp. 343–345, 2010.

[38] T. Baltrušaitis, P. Robinson, and L.-P. Morency, "Openface: an open source facial behavior analysis toolkit," in *Applications of Computer Vision (WACV), 2016 IEEE Winter Conference on*, pp. 1–10, IEEE, 2016.

[39] M. P. Segundo, L. Silva, O. R. P. Bellon, and C. C. Queirolo, "Automatic face segmentation and facial landmark detection in range images," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 40, no. 5, pp. 1319–1330, 2010.

[40] P. Lucey, J. F. Cohn, T. Kanade, J. Saragih, Z. Ambadar, and I. Matthews, "The extended cohn-kanade dataset (ck+): A complete dataset for action unit and emotion-specified expression," in *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on*, pp. 94–101, IEEE, 2010.

[41] M. Lyons, M. Kamachi, and J. Gyoba, "Japanese female facial expressions (jaffe). database of digital images (1997)," *Google Scholar*, 1997.

[42] A. Mollahosseini, B. Hasani, and M. H. Mahoor, "Affectnet: A database for facial expression, valence, and arousal computing in the wild," *arXiv preprint arXiv:1708.03985*, 2017.

[43] I. Sneddon, M. McRorie, G. McKeown, and J. Hanratty, "The belfast induced natural emotion database," *IEEE Transactions on Affective Computing*, vol. 3, no. 1, pp. 32–41, 2012.

[44] N. Aifanti, C. Papachristou, and A. Delopoulos, "The mug facial expression

database," in *Image analysis for multimedia interactive services (WIAMIS), 2010 11th international workshop on*, pp. 1–4, IEEE, 2010.

[45] A. Dhall, R. Goecke, S. Lucey, T. Gedeon, *et al.*, "Collecting large, richly annotated facial-expression databases from movies," *IEEE multimedia*, vol. 19, no. 3, pp. 34–41, 2012.

[46] A. Dhall, R. Goecke, S. Ghosh, J. Joshi, J. Hoey, and T. Gedeon, "From individual to group-level emotion recognition: Emotiw 5.0," in *Proceedings of the 19th ACM International Conference on Multimodal Interaction*, pp. 524–528, ACM, 2017.

[47] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[48] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[49] A. Graves, M. Liwicki, S. Fernández, R. Bertolami, H. Bunke, and J. Schmidhuber, "A novel connectionist system for unconstrained handwriting recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 31, no. 5, pp. 855–868, 2009.

[50] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Acoustics, speech and signal processing (icassp), 2013 ieee international conference on*, pp. 6645–6649, IEEE, 2013.

[51] S. Alhazmi, M. K. Saini, and A. El Saddik, "Multimedia fatigue detection for adaptive infotainment user interface," in *Proceedings of the 2nd Workshop on Computational Models of Social Interactions: Human-Computer-Media Communication*, pp. 15–24, ACM, 2015.

[52] Y. Tong, Y. Wang, Z. Zhu, and Q. Ji, "Robust facial feature tracking under varying face pose and facial expression," *Pattern Recognition*, vol. 40, no. 11, pp. 3195–3208, 2007.

[53] J. Hosek, M. Ries, P. Vajsar, L. Nagy, S. Andreev, O. Galinina, Y. Koucheryavy, Z. Sulc, P. Hais, and R. Penizek, "User's happiness in numbers: Understanding mobile youtube quality expectations," in *2015 38th International Conference on Telecommunications and Signal Processing (TSP)*, pp. 607–611, IEEE, 2015.

[54] D. McDuff, R. El Kaliouby, D. Demirdjian, and R. Picard, "Predicting online media effectiveness based on smile responses gathered over the internet," in

Automatic Face and Gesture Recognition (FG), 2013 10th IEEE International Conference and Workshops on, pp. 1–7, IEEE, 2013.

[55] D. McDuff, R. El Kaliouby, J. F. Cohn, and R. W. Picard, "Predicting ad liking and purchase intent: Large-scale analysis of facial responses to ads," *IEEE Transactions on Affective Computing*, vol. 6, no. 3, pp. 223–235, 2015.

[56] S. Taheri, P. Turaga, and R. Chellappa, "Towards view-invariant expression analysis using analytic shape manifolds," in *Automatic Face & Gesture Recognition and Workshops (FG 2011), 2011 IEEE International Conference on*, pp. 306–313, IEEE, 2011.

[57] K. Sikka, T. Wu, J. Susskind, and M. Bartlett, "Exploring bag of words architectures in the facial expression domain," in *European Conference on Computer Vision*, pp. 250–259, Springer, 2012.

[58] Y. Rahulamathavan, R. C.-W. Phan, J. A. Chambers, and D. J. Parish, "Facial expression recognition in the encrypted domain based on local fisher discriminant analysis," *cognition*, vol. 4, p. 6, 2013.

[59] S. Happy and A. Routray, "Automatic facial expression recognition using features of salient facial patches," *IEEE transactions on Affective Computing*, vol. 6, no. 1, pp. 1–12, 2015.

[60] I. J. Goodfellow, D. Erhan, P. L. Carrier, A. Courville, M. Mirza, B. Hamner, W. Cukierski, Y. Tang, D. Thaler, D.-H. Lee, *et al.*, "Challenges in representation learning: A report on three machine learning contests," in *International Conference on Neural Information Processing*, pp. 117–124, Springer, 2013.

[61] P. Werner, A. Al-Hamadi, K. Limbrecht-Ecklundt, S. Walter, S. Gruss, and H. C. Traue, "Automatic pain assessment with facial activity descriptors," *IEEE Transactions on Affective Computing*, vol. 8, no. 3, pp. 286–299, 2017.

[62] Y. Li, S. Wang, Y. Zhao, and Q. Ji, "Simultaneous facial feature tracking and facial expression recognition," *IEEE Transactions on Image Processing*, vol. 22, no. 7, pp. 2559–2573, 2013.

[63] F. Long, T. Wu, J. R. Movellan, M. S. Bartlett, and G. Littlewort, "Learning spatiotemporal features by using independent component analysis with application to facial expression recognition," *Neurocomputing*, vol. 93, pp. 126–132, 2012.

[64] T. R. Almaev and M. F. Valstar, "Local gabor binary patterns from three orthogonal planes for automatic facial expression recognition," in *2013 Humaine Association Conference on Affective Computing and Intelligent Interaction*, pp. 356–361, IEEE, 2013.

[65] X. Huang, Q. He, X. Hong, G. Zhao, and M. Pietikainen, "Improved spatiotemporal local monogenic binary pattern for emotion recognition in the wild," in *Proceedings of the 16th International Conference on Multimodal Interaction*, pp. 514–520, ACM, 2014.

[66] Y. Fan, X. Lu, D. Li, and Y. Liu, "Video-based emotion recognition using cnn-rnn and c3d hybrid networks," in *Proceedings of the 18th ACM International Conference on Multimodal Interaction*, pp. 445–450, ACM, 2016.

[67] J. Han, Z. Zhang, N. Cummins, F. Ringeval, and B. Schuller, "Strength modelling for real-worldautomatic continuous affect recognition from audiovisual signals," *Image and Vision Computing*, vol. 65, pp. 76–86, 2017.

[68] J. Atkinson and D. Campos, "Improving bci-based emotion recognition by combining eeg feature selection and kernel classifiers," *Expert Systems with Applications*, vol. 47, pp. 35–41, 2016.

[69] S. Poria, I. Chaturvedi, E. Cambria, and A. Hussain, "Convolutional mkl based multimodal emotion recognition and sentiment analysis," in *Data Mining (ICDM), 2016 IEEE 16th International Conference on*, pp. 439–448, IEEE, 2016.

[70] F. Eyben, M. Wöllmer, and B. Schuller, "Opensmile: the munich versatile and fast open-source audio feature extractor," in *Proceedings of the 18th ACM international conference on Multimedia*, pp. 1459–1462, ACM, 2010.

[71] W.-L. Zheng and B.-L. Lu, "Investigating critical frequency bands and channels for eeg-based emotion recognition with deep neural networks," *IEEE Transactions on Autonomous Mental Development*, vol. 7, no. 3, pp. 162–175, 2015.

[72] S. E. Kahou, X. Bouthillier, P. Lamblin, C. Gulcehre, V. Michalski, K. Konda, S. Jean, P. Froumenty, Y. Dauphin, N. Boulanger-Lewandowski, *et al.*, "Emonets: Multimodal deep learning approaches for emotion recognition in video," *Journal on Multimodal User Interfaces*, vol. 10, no. 2, pp. 99–111, 2016.

[73] H. Kaya, F. Gürpınar, and A. A. Salah, "Video-based emotion recognition in the wild using deep transfer learning and score fusion," *Image and Vision Computing*, vol. 65, pp. 66–75, 2017.

[74] B. B. Turker, Y. Yemez, T. M. Sezgin, and E. Erzin, "Audio-facial laughter detection in naturalistic dyadic conversations," *IEEE Transactions on Affective Computing*, vol. 8, no. 4, pp. 534–545, 2017.

[75] Z. Guo, L. Zhang, and D. Zhang, "A completed modeling of local binary pattern operator for texture classification," *IEEE Transactions on Image Processing*, vol. 19, no. 6, pp. 1657–1663, 2010.

[76] D. G. Lowe, "Object recognition from local scale-invariant features," vol. 2, pp. 1150–1157, 1999.

[77] J. Wu and J. M. Rehg, "Centrist: A visual descriptor for scene categorization," *IEEE transactions on pattern analysis and machine intelligence*, vol. 33, no. 8, pp. 1489–1501, 2011.

[78] J. Li, S. Roy, J. Feng, and T. Sim, "Happiness level prediction with sequential inputs via multiple regressions," in *Proceedings of the 18th ACM International Conference on Multimodal Interaction*, pp. 487–493, ACM, 2016.

[79] V. Vonikakis, Y. Yazici, V. D. Nguyen, and S. Winkler, "Group happiness assessment using geometric features and dataset balancing," in *Proceedings of the 18th ACM International Conference on Multimodal Interaction*, pp. 479–486, ACM, 2016.

[80] B. Sun, Q. Wei, L. Li, Q. Xu, J. He, and L. Yu, "Lstm for dynamic emotion and group emotion recognition in the wild," in *Proceedings of the 18th ACM International Conference on Multimodal Interaction*, pp. 451–457, ACM, 2016.

[81] Q. You, J. Luo, H. Jin, and J. Yang, "Robust image sentiment analysis using progressively trained and domain transferred deep networks.," in *AAAI*, pp. 381–388, 2015.

[82] X. Wang, J. Jia, J. Tang, B. Wu, L. Cai, and L. Xie, "Modeling emotion influence in image social networks," *IEEE Transactions on Affective Computing*, vol. 6, no. 3, pp. 286–297, 2015.

[83] J. Yuan, S. Mcdonough, Q. You, and J. Luo, "Sentribute: image sentiment analysis from a mid-level perspective," p. 10, 2013.

[84] D. Borth, T. Chen, R. Ji, and S.-F. Chang, "Sentibank: large-scale ontology and classifiers for detecting sentiment and emotions in visual content," pp. 459–460, 2013.

[85] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," pp. 91–99, 2015.

[86] X. Jiang, Y. Pang, X. Li, and J. Pan, "Speed up deep neural network based pedestrian detection by sharing features across multi-scale models," *Neurocomputing*, vol. 185, pp. 163–170, 2016.

[87] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," pp. 580–587, 2014.

[88] G. Lin, Q. Wu, L. Qiu, and X. Huang, "Image super-resolution using a dilated convolutional neural network," *Neurocomputing*, vol. 275, pp. 1219–1230, 2018.

[89] J. Yu, Y. Rui, and D. Tao, "Click prediction for web image reranking using multimodal sparse coding," *IEEE Transactions on Image Processing*, vol. 23, no. 5, pp. 2019–2032, 2014.

[90] Y. Tan, P. Tang, Y. Zhou, W. Luo, Y. Kang, and G. Li, "Photograph aesthetical evaluation and classification with deep convolutional neural networks," *Neurocomputing*, vol. 228, pp. 165–175, 2017.

[91] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *arXiv preprint arXiv:1606.00915*, 2016.

[92] Z. Tang, L. Luo, H. Peng, and S. Li, "A joint residual network with paired relus activation for image super-resolution," *Neurocomputing*, vol. 273, pp. 37–46, 2018.

[93] J. Yu, X. Yang, F. Gao, and D. Tao, "Deep multimodal distance metric learning using click constraints for image ranking," *IEEE transactions on cybernetics*, 2017.

[94] R. Aly, D. Hiemstra, F. de Jong, and P. M. Apers, "Simulating the future of concept-based video retrieval under improved detector performance," *Multimedia Tools and Applications*, vol. 60, no. 1, pp. 203–231, 2012.

[95] G. Wu, W. Lu, G. Gao, C. Zhao, and J. Liu, "Regional deep learning model for visual tracking," *Neurocomputing*, vol. 175, pp. 310–323, 2016.

[96] K. Moustafa, S. Luz, and L. Longo, "Assessment of mental workload: a comparison of machine learning methods and subjective assessment techniques," in *International Symposium on Human Mental Workload: Models and Applications*, pp. 30–50, Springer, 2017.

[97] S. M. Khan, A. A. Khan, and O. Farooq, "A neural network classification of semg signals for estimation of force while gripping," in *Advances in Engineering Design*, pp. 585–593, Springer, 2019.

[98] M. Sood, V. Kumar, and S. V. Bhooshan, "Comparison of machine learning methods for prediction of epilepsy by neurophysiological eeg signals," *Int J Pharm Bio Sci*, vol. 5, no. 2, pp. 6–15, 2014.

[99] J. M. González-Calabozo, F. J. Valverde-Albacete, and C. Peláez-Moreno, "Interactive knowledge discovery and data mining on genomic expression data with numeric formal concept analysis," *BMC bioinformatics*, vol. 17, no. 1, p. 374, 2016.

[100] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," pp. 248–255, 2009.

[101] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting.," *Journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[102] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proceedings of COMPSTAT'2010*, pp. 177–186, Springer, 2010.

[103] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," pp. 2261–2269, 2017.

[104] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," *arXiv preprint arXiv:1610.02357*, 2016.

[105] M. Lin, Q. Chen, and S. Yan, "Network in network," *arXiv preprint arXiv:1312.4400*, 2013.

[106] I. J. Goodfellow, D. Warde-Farley, M. Mirza, A. Courville, and Y. Bengio, "Maxout networks," *arXiv preprint arXiv:1302.4389*, 2013.

[107] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE international conference on computer vision*, pp. 1026–1034, 2015.

[108] B. Xu, N. Wang, T. Chen, and M. Li, "Empirical evaluation of rectified activations in convolutional network," *arXiv preprint arXiv:1505.00853*, 2015.

[109] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (elus)," *arXiv preprint arXiv:1511.07289*, 2015.

[110] L. Trottier, P. Giguere, and B. Chaib-draa, "Parametric exponential linear unit for deep convolutional neural networks," *arXiv preprint arXiv:1605.09332*, 2016.

[111] F. Agostinelli, M. Hoffman, P. Sadowski, and P. Baldi, "Learning activation functions to improve deep neural networks," *arXiv preprint arXiv:1412.6830*, 2014.

[112] Ö. F. Ertuğrul, "A novel type of activation function in artificial neural networks: Trained activation function," *Neural Networks*, vol. 99, pp. 148–157, 2018.

[113] L. Xiao, "A nonlinearly activated neural dynamics and its finite-time solution to time-varying nonlinear equation," *Neurocomputing*, vol. 173, pp. 1983–1988, 2016.

[114] L. Xiao and R. Lu, "Finite-time solution to nonlinear equation using recurrent neural dynamics with a specially-constructed activation function," *Neurocomputing*, vol. 151, pp. 246–251, 2015.

[115] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *arXiv preprint arXiv:1603.04467*, 2016.

[116] G. Bradski and A. Kaehler, "Opencv," *Dr. Dobbs journal of software tools*, 2000.

[117] I. J. Goodfellow, D. Erhan, P. L. Carrier, A. Courville, M. Mirza, B. Hamner, W. Cukierski, Y. Tang, D. Thaler, D.-H. Lee, *et al.*, "Challenges in representation learning: A report on three machine learning contests," *Neural Networks*, vol. 64, pp. 59–63, 2015.

[118] A. Dhall, O. Ramana Murthy, R. Goecke, J. Joshi, and T. Gedeon, "Video and image based emotion recognition challenges in the wild: Emotiw 2015,"

in *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*, pp. 423–426, ACM, 2015.

[119] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2818–2826, 2016.

[120] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," pp. 818–833, 2014.

[121] P. Viola and M. J. Jones, "Robust real-time face detection," *International journal of computer vision*, vol. 57, no. 2, pp. 137–154, 2004.

[122] N. Kalchbrenner, I. Danihelka, and A. Graves, "Grid long short-term memory," *arXiv preprint arXiv:1507.01526*, 2015.

[123] J. Yang, Y.-G. Jiang, A. G. Hauptmann, and C.-W. Ngo, "Evaluating bag-of-visual-words representations in scene classification," pp. 197–206, 2007.

[124] P. Tirilly, V. Claveau, and P. Gros, "Language modeling for bag-of-visual words image categorization," pp. 249–258, 2008.

[125] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," vol. 2, pp. 2169–2178, 2006.

[126] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," pp. 404–417, 2006.

[127] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," pp. 2564–2571, 2011.

[128] E. Rosten, R. Porter, and T. Drummond, "Faster and better: A machine learning approach to corner detection," *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 1, pp. 105–119, 2010.

[129] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "Brief: Binary robust independent elementary features," pp. 778–792, 2010.

[130] P. Ekman, *Darwin and facial expression: A century of research in review.* Ishk, 2006.

[131] S. Tokuno, G. Tsumatori, S. Shono, E. Takei, G. Suzuki, T. Yamamoto, S. Mituyoshi, and M. Shimura, "Usage of emotion recognition in military health

care," in *Defense Science Research Conference and Expo (DSR), 2011*, pp. 1–5, IEEE, 2011.

[132] E. Hudlicka and J. Broekens, "Foundations for modelling emotions in game characters: Modelling emotion effects on cognition," in *Affective Computing and Intelligent Interaction and Workshops, 2009. ACII 2009. 3rd International Conference on*, pp. 1–6, IEEE, 2009.

[133] D. Morrison, R. Wang, L. C. De Silva, and W. Xu, "Real-time spoken affect classification and its application in call-centres," in *Information Technology and Applications, 2005. ICITA 2005. Third International Conference on*, vol. 1, pp. 483–487, IEEE, 2005.

[134] A. Tawari and M. M. Trivedi, "Audio visual cues in driver affect characterization: Issues and challenges in developing robust approaches," in *Neural Networks (IJCNN), The 2011 International Joint Conference on*, pp. 2997–3002, IEEE, 2011.

[135] M. Turk, "Multimodal interaction: A review," *Pattern Recognition Letters*, vol. 36, pp. 189–195, 2014.

[136] S. W. Chew, P. J. Lucey, S. Lucey, J. Saragih, J. Cohn, and S. Sridharan, "Person-independent facial expression detection using constrained local models," *Proceedings of FG 2011 Facial Expression Recognition and Analysis Challenge*, pp. 915–920, 2011.

[137] Y.-I. Tian, T. Kanade, and J. F. Cohn, "Recognizing action units for facial expression analysis," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 23, no. 2, pp. 97–115, 2001.

[138] M. S. Bartlett, G. Littlewort, M. G. Frank, C. Lainscsek, I. R. Fasel, J. R. Movellan, *et al.*, "Automatic recognition of facial actions in spontaneous expressions.," *Journal of multimedia*, vol. 1, no. 6, pp. 22–35, 2006.

[139] C. Shan, S. Gong, and P. W. McOwan, "Facial expression recognition based on local binary patterns: A comprehensive study," *Image and vision Computing*, vol. 27, no. 6, pp. 803–816, 2009.

[140] M. Pantic and I. Patras, "Dynamics of facial expression: recognition of facial actions and their temporal segments from face profile image sequences," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 36, no. 2, pp. 433–449, 2006.

[141] M. F. Valstar and M. Pantic, "Combined support vector machines and hidden markov models for modeling facial action temporal dynamics," in *International workshop on human-computer interaction*, pp. 118–127, Springer, 2007.

[142] P. Ekman, "Darwin, deception, and facial expression," *Annals of the New York Academy of Sciences*, vol. 1000, no. 1, pp. 205–221, 2003.

[143] A. Dhall, A. Asthana, R. Goecke, and T. Gedeon, "Emotion recognition using phog and lpq features," in *Automatic Face & Gesture Recognition and Workshops (FG 2011), 2011 IEEE International Conference on*, pp. 878–883, IEEE, 2011.

[144] M. Liu, R. Wang, S. Li, S. Shan, Z. Huang, and X. Chen, "Combining multiple kernel methods on riemannian manifold for emotion recognition in the wild," in *Proceedings of the 16th International Conference on Multimodal Interaction*, pp. 494–501, ACM, 2014.

[145] J. Chen, Z. Chen, Z. Chi, and H. Fu, "Emotion recognition in the wild with feature fusion and multiple kernel learning," in *Proceedings of the 16th International Conference on Multimodal Interaction*, pp. 508–513, ACM, 2014.

[146] statista, "https://www.statista.com/statistics/264810/number-of-monthly-active-facebook-users-worldwide/," (Date last accessed on January 08, 2018).

[147] statista, "https://www.statista.com/topics/2019/youtube/," (Date last accessed on May 16, 2018).

[148] omnicoreagency, "https://www.omnicoreagency.com/youtube-statistics/," (Date last accessed on April 19, 2018).

[149] D. E. King, "Dlib-ml: A machine learning toolkit," *Journal of Machine Learning Research*, vol. 10, no. Jul, pp. 1755–1758, 2009.

[150] R. C. Gonzalez, R. E. Woods, *et al.*, "Digital image processing [m]," *Publishing house of electronics industry*, vol. 141, no. 7, 2002.

[151] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747*, 2016.

[152] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[153] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio, "Theano: A cpu and gpu math compiler in python," pp. 1–7, 2010.

[154] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," pp. 249–256, 2010.

[155] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," pp. 1310–1318, 2013.

[156] G. E. Dahl, D. Yu, L. Deng, and A. Acero, "Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition," *IEEE Transactions on audio, speech, and language processing*, vol. 20, no. 1, pp. 30–42, 2012.

[157] A. R. Barron, "Universal approximation bounds for superpositions of a sigmoidal function," *IEEE Transactions on Information theory*, vol. 39, no. 3, pp. 930–945, 1993.

[158] G. Lewicki and G. Marino, "Approximation by superpositions of a sigmoidal function," *Zeitschrift fur Analysis und ihre Anwendungen*, vol. 22, no. 2, pp. 463–470, 2003.

[159] D. Costarelli and G. Vinti, "Pointwise and uniform approximation by multivariate neural network operators of the max-product type," *Neural Networks*, vol. 81, pp. 81–90, 2016.

[160] D. Costarelli and G. Vinti, "Convergence for a family of neural network operators in orlicz spaces," *Mathematische Nachrichten*, vol. 290, no. 2-3, pp. 226–235, 2017.

[161] W. Zaremba, I. Sutskever, and O. Vinyals, "Recurrent neural network regularization," *arXiv preprint arXiv:1409.2329*, 2014.

[162] A. Krizhevsky, V. Nair, and G. Hinton, "The cifar-10 dataset," (Date last accessed on November 19, 2017).

[163] Y. LeCun, "The mnist database of handwritten digits," (Date last accessed on July 19, 2017).

[164] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning," vol. 2011, no. 2, p. 5, 2011.

[165] T. Tieleman and G. Hinton, "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude," *COURSERA: Neural networks for machine learning*, vol. 4, no. 2, pp. 26–31, 2012.

[166] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter, "Self-normalizing neural networks," pp. 972–981, 2017.

[167] D. Bau, B. Zhou, A. Khosla, A. Oliva, and A. Torralba, "Network dissection: Quantifying interpretability of deep visual representations," *arXiv preprint arXiv:1704.05796*, 2017.