Fall 2017

Time-Efficient Hybrid Approach for Facial Expression Recognition

Roshni Velluva Puthanidam
San Jose State University

Time-Efficient Hybrid Approach for Facial Expression Recognition

A Writing Project

Presented to

The Faculty of the Department of Computer Science

San José State University

In Partial Fulfilment

of the Requirements for the Degree

Master of Computer Science

By

Roshni Velluva Puthanidam

Fall 2017

SAN JOSÉ STATE UNIVERSITY

The Undersigned Thesis Committee Approves the Thesis Titled

Time-Efficient Hybrid Approach for Facial Expression Recognition

By
Roshni Velluva Puthanidam

APPROVED FOR THE DEPARTMENT OF COMPUTER SCIENCE

_____
Dr. Teng Moh, Department of Computer Science 11/20/2017

_____
Dr. Sami Khuri, Department of Computer Science 11/20/2017

_____
Dr. Suneuy Kim, Department of Computer Science 11/20/2017

ABSTRACT

Facial expression recognition is an emerging research area for improving human and computer interaction. This research plays a significant role in the field of social communication, commercial enterprise, law enforcement, and other computer interactions. In this paper, we propose a time-efficient hybrid design for facial expression recognition, combining image pre-processing steps and different Convolutional Neural Network (CNN) structures providing better accuracy and greatly improved training time. We are predicting seven basic emotions of human faces: sadness, happiness, disgust, anger, fear, surprise and neutral. The model performs well regarding challenging facial expression recognition where the emotion expressed could be one of several due to their quite similar facial characteristics such as anger, disgust, and sadness. The experiment to test the model was conducted across multiple databases and different facial orientations, and to the best of our knowledge, the model provided an accuracy of about 89.58% for KDEF dataset, 100% accuracy for JAFFE dataset and 71.975% accuracy for combined (KDEF + JAFFE + SFEW) dataset across these different scenarios. Performance evaluation was done by cross-validation techniques to avoid bias towards a specific set of images from a database.

ACKNOWLEDGEMENT

I Would like to express my honest gratitude to my project advisor Dr. Teng Moh for giving his continuous guidance, precious time and assistance during this project. I am grateful, and I thank my committee members Dr. Sami Khuri and Dr. Suneuy Kim for their thoughts, time, and suggestions.

Special thanks to The Karolinska Institute Emotion Lab, Acted Facial Expression in the Wild Database team and Jaffe Database team for giving us access to use the dataset they had developed.

We would also like to thank our computer science department for giving us with the required hardware for our project. We would also like to express our thanks to the library for giving required books and materials to learn different aspects and concepts of our project.

Last but not the least, I would also like to heartfully thank my family and friends for their moral support, encouragement, and blessings without which I could not able to finish my project successfully.

Table of Contents

Table of Figures

Table of Tables

1. Introduction

Facial expression recognition is an emerging research area which improves human and computer interaction. Facial expression can be defined as the facial changes in response to a person's internal emotional state, intentions, and/or social communications [1]. Recognition of emotions from facial expression is becoming an interesting and challenging research area because facial expressions convey abundant information on human mental states, behavior, intentions [2] and plays an important role in social communication and interactions [2]. Moreover, facial recognition plays a significant role in the fields of commercial enterprise and law enforcement due to increasing demands for security. Traditional ways of ensuring security such as ID cards, passwords, etc. are not reliable because they are easily hackable and inconvenient [3]. Furthermore, it is difficult to carry IDs or remember passwords [4].

Facial expression recognition impacts key applications in computer vision fields and many other fields including human-computer interaction, data-driven animation, interactive games, sociable robotics and neuromarketing [5]. There have been several expression recognition approaches implemented in the last few years, and a lot of advanced techniques have been developed in this area [6] [7] [8]. However, despite these immense efforts, it remains a challenging area of research because there is a large scope for improvement both in terms of accuracy of identifying emotions that have similar facial characteristics and training time of the network.

The traditional approach for facial expression recognition consists of two parts: feature extraction and classification. Features extracted from the training data always play an important role in the recognition problem because the classifier makes its decision based on the combination of extracted features. Characteristic features can be created by applying different methods including log Gabor filter, Gabor filters, higher-order local autocorrelation (HLAC), local binary pattern (LBP) operator, and a newly introduced approach called HLAC-like features (HLACLF). The most educational features are chosen based on both wrapper and filter feature selection method [9]. Neural networks are widely used for emotion recognition through continuous learning processes. They can learn from experience to improve their performance and to adapt themselves to changes in the environment. With recent advancements in parallel computing and deep learning, it has been possible to apply CNN-based deep neural networks

into tasks such as object classification to the problem of facial expression recognition, which has been impressive in terms of success rate [10] [11] [12] [13] [14] [15] [16].

In this paper, we present a hybrid approach for facial expression recognition, combining image pre-processing steps and different CNN structures providing better accuracy and lesser training time. Here we are using a CNN for feature extraction and classification because CNNs are fast to train compared to traditional neural networks. The traditional neural network does not consider the spatial structure of faces and fails to distinguish multiple emotions that have similar facial characteristics such as anger, disgust, and sadness (sample set of composite face images is shown in Figure 1.). Our approach to improving accuracy is using the special 3-D architecture of CNN and incorporating appropriate image pre-processing steps such as image cropping, downsampling, and synthetic sample generation. Training time of neural network is greatly improved by performing batch processing. Training time is about 2 to 4 hours, depending on the size of training data. Four types of CNN structures were tested over different datasets, and they achieved very high accuracy rates. We evaluated performance with cross-validation techniques to avoid bias towards a specific set of images from a database.

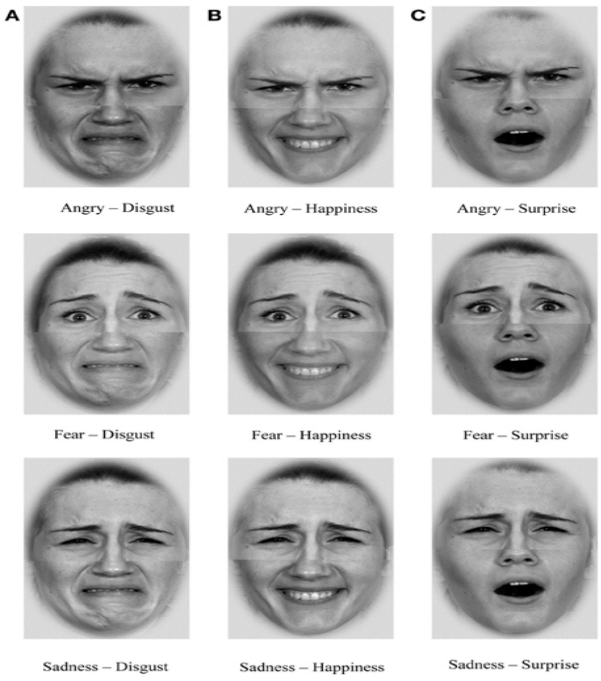


Figure 1. Facial Expression with similar facial characteristics

This paper is organized as follows: Section 2 presents technologies used and then the most recent related works that are followed by detailed description of the data and data preparation methods in Section 4. In Section 5, the CNN structure is described, experiments are explained, and the results are presented and compared with the state-of-the-art approach. Finally, Section 7 presents the conclusion.

2. Technologies Used

This project is about improving the accuracy of facial expression recognition rate under various scenarios like different facial orientation, luminance, etc. across multiple datasets. The main technologies/libraries we have used to implement this project are:

- TensorFlow
- OpenCV (Open Source Computer Vision)
- SciKit Learn
- NumPy

All the required software libraries are open source and will be downloaded freely.

2.1 Tensor Flow

TensorFlow is an open source deep learning software library for numerical computation using data flow graphs. TensorFlow was basically developed by researchers and engineers working on the Google Brain team within Google's Machine Intelligence Research organization for the purposes of conducting machine learning and deep neural networks research. The system is comprehensive enough to be fit in a broad variety of other fields including computer vision, speech recognition, natural language processing, robotics, geographic information extraction, information retrieval, and computational drug discovery.

Each node in dataflow graph represents the numerical operation, while the edges of the graph refer to multi-dimensional data arrays (tensors) that flow between them. All numerical operations occur within a session. Graphs and sessions (Tensor Flow's mechanism for running dataflow graphs across one or more local or remote devices) are created individually. A graph is like a blueprint (design plan), and a session is like a construction site. This architecture gives flexibility to the application developers: TensorFlow empowers developers to try different things with novel optimization and training algorithms [17].

TensorFlow owns a computational graph that helps them to perform with greater efficiency compared with implementing directly in Python for the same set of computations.

Since TensorFlow knows the entire computation must be done in the graph earlier, whereas NumPy only knows the single mathematical operation calculation at a time. So, TensorFlow can be more efficient than NumPy. Another important factor is that TensorFlow can also calculate the gradients needed to optimize the variables of the graph automatically and make the model perform better. The graph is a combination of simple mathematical formulations so the gradient of the whole graph can be estimated by applying the chain-rule for derivatives. Another benefit that TensorFlow can take is of multi-core CPUs as well as GPUs - and for TensorFlow Google built specialized chips which are called TPUs (Tensor Processing Units). TPUs are even faster than GPUs.

A sample workflow of TensorFlow is shown below:

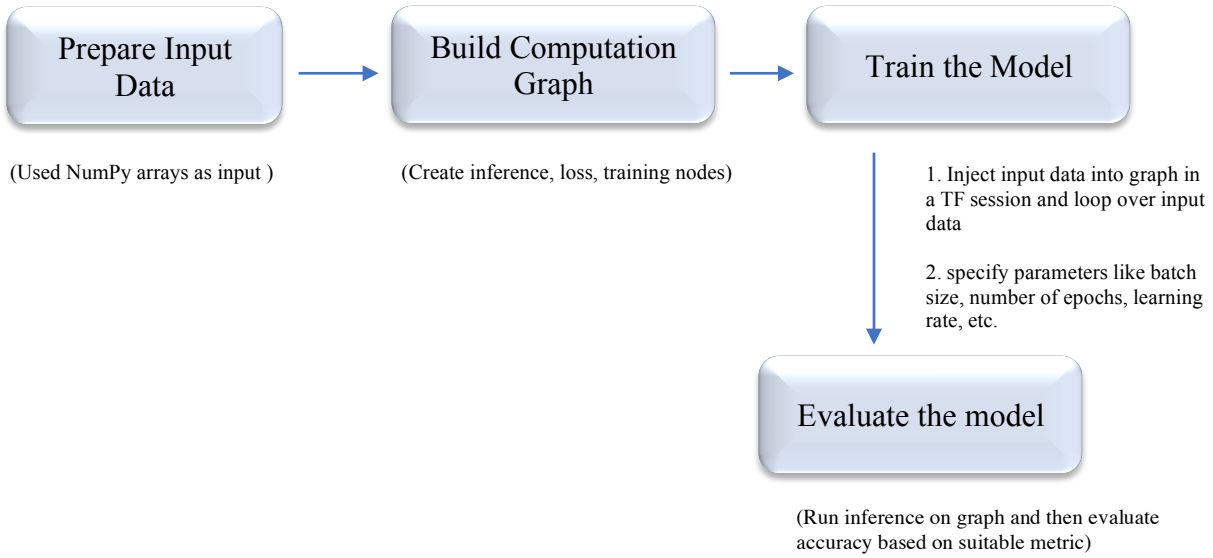


Figure 2. TensorFlow workflow

The TensorFlow graph we have used in our implementation consists of the following parts which will be detailed below:

- Placeholder variables are used to provide input to the graph.
- The variables that get optimized during cost optimization step makes convolutional network performance better.
- Optimization of the variables is guided using a cost measure.
- Variables are updated using an Optimization method.

- The graph in TensorFlow has various debugging statements.

2.2 OpenCV

OpenCV (Open Source Computer Vision Library) is an open source library that comprises of several functions/operations for computer vision application. It is currently managed and maintained by Willow Garage but initially developed by Intel. Functions in this library are cross-platform and execute on Mac OS, Windows, FreeBSD, Android, Maemo, OpenBSD, iOS, and Linux. Since it is released under a BSD license, it is free for both commercial and academic use. OpenCV was designed with a significant focus on real-time applications and higher computational efficiency. Facial recognition system, 2D and 3D feature toolkits, Gesture recognition, Human-computer interaction (HCI), Ego-motion estimation, etc. are some of the application fields in OpenCV's. OpenCV includes a statistical machine learning library that contains: Boosting, Decision tree learning, Artificial neural networks, Deep neural networks (DNN), etc. to back some of these areas. [18].

We have used OpenCV for data preparation including color to grayscale conversion, synthetic image generation, face recognition and cropping, downsampling, etc. The purposes of the developing libraries of OpenCV are different. We have used these purposes for the benefit of our project:

- “Advance vision research by providing open source as well as optimized code for basic vision infrastructure.”
- “Distribute vision information by implementing a common infrastructure for all developers so that they can share their research and knowledge.”
- “Building portable, performance-optimized code available for free so that others can contribute computer vision based application.”

2.3 Scikit-learn

Scikit (SciPy Toolkit)-learn is an open source Python library that implements a range of machine learning, pre-processing, cross-validation and visualization algorithms using a unified interface. It comprises of simple and efficient tools for data mining and data analysis. It is accessible to everybody and reusable in various contexts. It is a Python module built on top of SciPy and distributed under the 3-Clause BSD license. We have used new easy to use sklearn interface of TensorFlow for performance evaluation (cross-validation, confusion matrix). There are some functionalities that are similar in Scikit-learn and TensorFlow which allow scikit-learn users to seamlessly use TensorFlow.

While scikit-learn has highly-optimised algorithms in its armory, it lacks the capability to scale-up when encountered with many data points. However, TensorFlow gives quite a numerous advantage over sci-kit-learn. The following are some of the reason for us to choose TensorFlow over scikit-learn:

- High-performance ML modules
- Customisability
- Scaling-up to cater to large data points
- Ability to utilize GPUs and train across geographically distributed GPU devices
- Leverage Google Cloud to make inferences on a trained ML model
- Highly flexible Apache 2.0 license while scikit-learn is on BSD license (although both are commercially usable, Apache 2.0 is less prone to patent litigations). [19]

2.4 NumPy

The NumPy library is the central library for scientific computing in Python. It gives a high-performance multidimensional array object and tools for operating with these arrays. We have used NumPy to provide input data to TensorFlow. The following are some of the features of NumPy we have used:

- Object used is a powerful N-dimensional array
- Sophisticated (broadcasting) functions

- All Tools are for integrating C/C++ and Fortran code
- Useful in linear algebra, Fourier transform, and random number capabilities.

Besides its clear scientific applications, NumPy can also be applied as an efficient multi-dimensional package of generic data. Optional data-types can be defined. This supports NumPy to seamlessly and quickly integrates with a broad variety of databases. NumPy is authorized under the BSD license, allowing reuse by some restrictions [20].

3. Related Works

Facial expression recognition is an example of a problem that humans are good at solving, but computers are not [1]. A lot of work has been invested in trying to have computers reach the same accuracy as humans, and some examples of these attempts are highlighted here. Several emotion recognition methods have been developed in the last few years, and a lot of advancement has been achieved in this area [21]. This section focuses on some closely related approaches to our work.

Ma and Khorasani proposed two facial expression recognition techniques [22]. They used Lower-frequency 2-D DCT coefficients of binarized edge images as features for recognition. A constructive one-hidden-layer (OHL) feedforward neural network (OHL-NN) was used for the first approach, and K-means algorithm as classifiers was used in the second approach. Their approach significantly reduced the size of the neural network by pruning technique while improving the generalization capability and the recognition rate. The technique they proposed was tested to a database with images of 60 men, all having five facial expressions (neutral, smile, anger, sadness, and surprise). They used 40 images for network training, and the remaining 20 for generalization and testing. Confusion matrices were used for performance evaluation. They achieved high recognition rates, 100% for the training, and 93.75% for generalizing images. Even though they achieved high accuracy, they only had five distinct emotions.

Facial emotion recognition is usually accomplished sequentially in three individual stages: feature (shape of the eyes, nose, cheekbones, and jaw) extraction and learning, feature selection, and classifier modelling [23]. "Extensive empirical researches are required to search for an optimal combination of feature representation (feature extraction), feature set (feature selection), and classifier to achieve good recognition performance" [19]. P. Liu, et al. [19] presented a new Boosted Deep Belief Network (BDBN) for implementing the three training stages iteratively in a unified loopy framework: "BDBN framework consists of two interconnected learning processes: a bottom-up unsupervised feature learning (BU-UFL) method that learns hierarchical feature representations given input data and a boosted top-down supervised feature strengthen (BTD- SFS) process that refines the features jointly in a supervised manner." This method used input image patches instead of the entire image for facial

emotion recognition. A collection of features, which is useful to distinguish expression-related facial appearance/shape changes, could be detected and selected to develop a boosted powerful classifier in a statistical process through this BDBN framework. According to the authors, through this framework, strong classifiers improved iteratively as learning continued and discriminative capabilities of selected features were strengthened according to their comparative importance to the strong classifier via a joint fine-tune process [19]. Their experiments were conducted using two public databases of static images, Extended Cohn-Kanade (CK+) database [24] and JAFFE [25] database, and achieved an accuracy of 96.7% and 68.0% respectively. They focused on six basic expressions and took about eight days to complete the overall training.

C. Shan et al. used Local Binary Patterns (LBP), which is not using a neural network, as the feature extractor [26]. In this paper, they merged and compared different machine learning techniques such as Support Vector Machine, Linear Discriminant Analysis, etc. for facial expressions. They empirically assessed facial representation based on LBP and based on statistically local features for person-independent facial expression recognition. Through their experiments, they observed that LBP features operated stably and robustly across a useful range of low resolutions of face images and produced promising performance in compressed low-resolution video sequences captured in real-world circumstances. Using SVM and LBP, they achieved 95.1% accuracy in the Cohn-Kanade [24] database and used a 10-fold cross-validation scheme for performance evaluation.

S. Borah and S. Konwar [3] presented an artificial neural network (ANN)-based human facial expression recognition method. The authors detected 22 facial feature points through an automatic technique and generated feature vector by calculating the Euclidian distance between certain points. These vectors were given as input to a multi-layer perceptron (MLP) neural network. They used connected component analysis for face detection and 2D Color Space Skin Clustering method for skin color detection. In this paper, the facial expressions were classified into seven basic expressions (anger, disgust, sad, surprise, fear, happy, and neutral) using a feed forward back propagation neural network. The best result achieved in their work was an accuracy rate of 85% in the color FERET database. Even though the method proposed in this paper could automatically locate facial feature points at high accuracy for most front faced images, it was limited when angle rotated images were involved. They did not mention anything

about training time.

An approach using CNN for feature and classification has also been used before [5]. The authors proposed a simple model for facial emotion recognition which uses a combination of CNN and specific image pre-processing steps that achieve comparatively higher accuracy. CNN and most of the machine learning methods helped produce high accuracy depending on a given feature set. The CNN models used raw images as input rather than hand-coded features. A typical architecture of a CNN had sequenced layers including an input layer, convolutional layers (pooling layer + rectified linear units), dense layers (multilayer fully-connected networks), and an output layer. The CNNs achieved better results than traditional neural networks. A. T. Lopes et al. [5] used a single CNN for performing three learning stages such as feature learning, feature selection, and classification of their work. The CNN was trained using many input images and their emotions. The weights of neural networks in each layer were adjusted while training. For recognizing an unknown image, the system applied a sequence of image pre-processing steps including spatial normalization, image cropping, downsampling, and intensity normalization on test images. They performed their experiments using Extended Cohn-Kanade (CK+) database [20] and achieved 97.81% accuracy. We are using a few of their image pre-processing steps in our implementation.

M. Shin et al. recommended a baseline CNN structure and image pre-processing methodology to improve facial expression recognition [27]. They experimented with four different CNN structures and five types of pre-processed images (raw, histogram equalization, isotropic smoothing, diffusion-based normalization, and difference of Gaussian) and suggested an efficient baseline CNN structure and pre-processing for facial expression recognition. The authors have used five different datasets: JAFFE, FER-2013, SFEW2.0, CK+ (extended Cohn-Kanade), and KDEF (Karolinska Directed Emotional Faces) [28] for performance evaluation. All datasets used in their work included seven types of the same emotional expression and were fully labeled. Their experiment result showed that three-layer network with a simple convolutional and a max pooling layer with histogram equalization images performed best among the four networks they used. Their highest accuracy rates were 50.61% with JAFFE and 59.15% with KDEF. They did not mention anything about the time taken to train their networks. We are also using all four of these networks in our experiment and this paper as our baseline.

So far, a lot of works for facial expression recognition using CNN-based deep neural networks were introduced, and their CNN structures and image pre-processing methods were all different. The main reason for the differences is that the selection of pre-processing methods and CNN structures were not main focuses in their research. However, proper selection of CNN structure and appropriate image pre-processing of input data are very important for improving results. In this paper, we are using four different network structures and several image pre-processing steps on different public databases to compare performance.

4. Data Preparation

In this section, we first describe the type of datasets used for our experiment and then operations/modifications applied to images (image pre-processing steps) in detail.

4.1 Dataset

To train and test our deep CNN, we are using different public datasets. The first dataset is the "Japanese Female Facial Expression (JAFFE) Database" [25], which comprises 213 images of seven facial expressions [six basic facial expressions (sad, happy, disgust, fear, surprise and angry) and one neutral]. The images are modeled by ten Japanese female models, and the head pose is always frontal. This database was planned and developed by Michael Lyons, Miyuki Kamachi, and Jiro Gyoba. JAFFE is one of the few datasets containing facial images of Asians and is a good dataset for our performance evaluation. A sample set of images from JAFFE is shown in Figure 3.

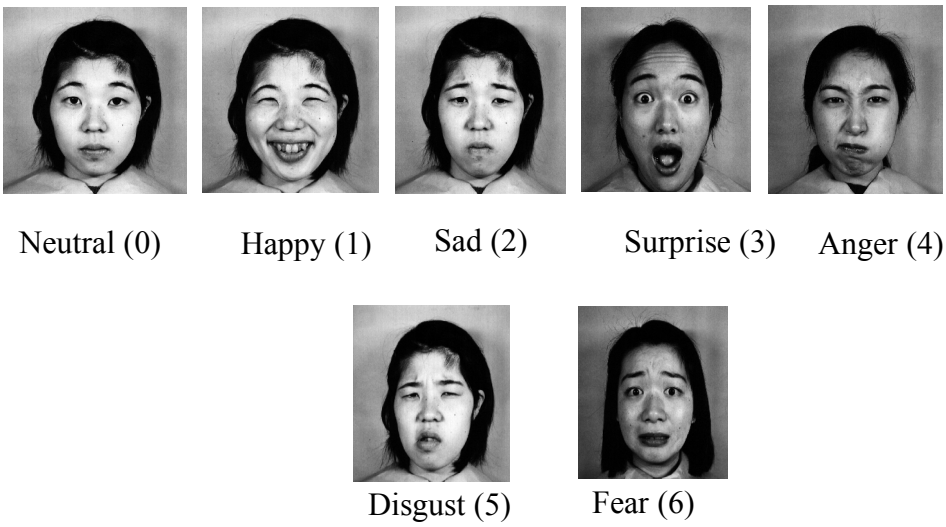


Figure 3. Sample images from JAFFE

Secondly, we are using the "Karolinska Directed Emotional Faces (KDEF) Database" [26] which contains 4900 images (980 frontal faced, 980 halves left rotated, 980 halves right rotated, 980 full left rotated and 980 full right rotated) of human facial expressions. This dataset was formed by Daniel Lundqvist, Anders Flykt, and Professor Arne Öhman at Karolinska Institutet, Department of Clinical Neuroscience, Section of Psychology, Stockholm, Sweden in 1998. This dataset was originally developed for psychological and medical research purposes, and is now particularly suitable for emotion experiments. It includes 70 people (35 female and

35 male), each displaying seven separate emotional expressions, with each expression available in five various angles. Since the dataset contains differently angled images, it is one of the best datasets we have for the performance evaluation. All datasets we are using in our work include seven types of the same emotion expression, and every image is fully labelled. A sample set of images from KDEF dataset is shown Figure 4.

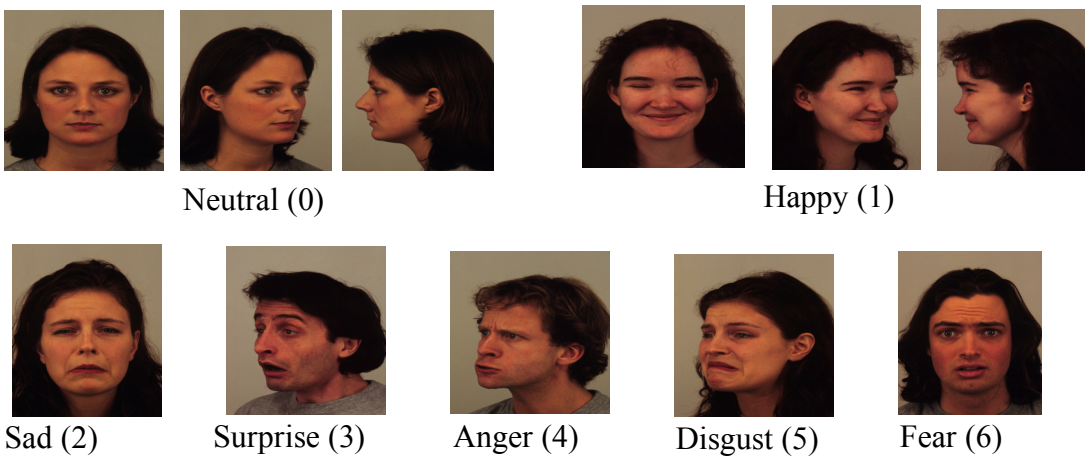


Figure 4. Sample images from KDEF (first is straight face, second is half rotated, and third is fully rotated face)

Finally, we are using the "Static Facial Expressions In The Wild (SFEW) Database" [29], selected from frames of a temporal facial expressions database, Acted Facial Expressions In the Wild (AFEW). This database includes unconstrained facial emotions, different head poses, wide age scale, occlusions, various focus and near to real-world illumination. AFEW was derived from movies clips. While movies are usually shot in slightly controlled circumstances, they provide close to real-life environments that are more realistic than other current datasets that were recorded in lab environments. SFEW has both frontal and non-frontal faces and different illumination conditions (Figure 5. shows some sample images from the database) which are very like real-world scenarios. SFEW images are labeled into basic expression angry, disgust, fear, happy, sad, surprise and the neutral class.

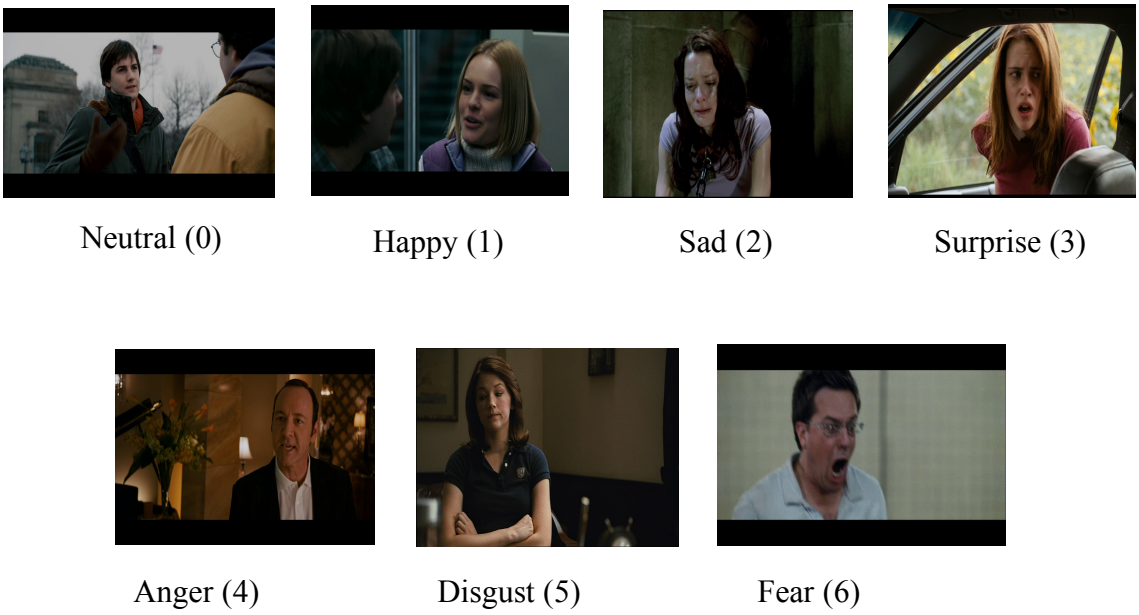


Figure 5. Sample images from SFEW

4.2 Image Pre-processing

Our proposed method of facial recognition consists of two steps: image pre-processing and performance evaluation. Performance evaluation comprises of two phases: training and testing. Quality and performance of our neural network model are constrained by the quality of data we consider for testing and training. All existing deep neural networks are susceptible to the quality of images such as distortions, particularly blurriness, rotations, brightness, and image size. Another problem of deep neural networks is that they usually require a lot of data in the training phase to achieve a good accuracy rate. In this section, we are describing some of the pre-processing methods applied to each dataset to improve quality of images.

1) Converting Color to Grayscale: Processing color images are complex and time-consuming, and grayscale provides an easy way out. For example, when we use color images, all operations should perform on all three-image planes (R, G, and B), while grayscale has only one image plane. As shown in Figure 6, the original image is the color image, and the color information doesn't benefit us to recognize important edges or other characteristics that are important for expression recognition.

To reduce the complexity of the code, we have converted color images to grayscale. Another important advantage of converting color to gray is that we can reduce the processing

time. While grayscale conversion can cause loss of color information that is required in many image processing applications, color info does not add any value in our experiment.

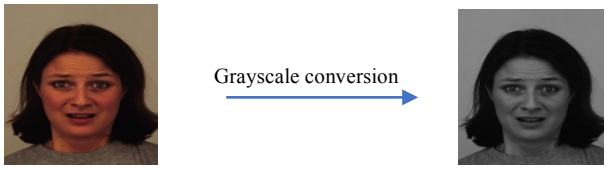


Figure 6. Converting color image to grayscale

2) Synthetic Image Generation: One of the best practices to get good results using the CNN is to use as large a training set as possible. P. Simard et al. proposed a method called synthetic images (artificial rotation on real images) [30]. According to this method, if the data is scarce, we can generate additional data by applying small transformations like rotation or translation to improve the performance of our network. So, we have expanded the smaller database to larger by adding a new form of noise (distortion) to the data. Generated synthetic images are very similar to the original images, resulting from applying small angle rotations on original images. Otherwise, noise in the image will affect the learning process and decrease the accuracy rate of the neural network if too many images with huge rotation angles (e.g. 45, 90, 180 degrees, etc.) are used. Illustration of synthetic sample generation is shown in Figure 7.

3) Image Cropping: Most of the papers on expression recognition used the face cropping technique to achieve high accuracy. As shown in Fig. 6, the image from our dataset has a lot of background information that does not give any important information about the expression to the expression recognition procedure. Moreover, this extra background information could decrease the accuracy rate of the recognition of expression because the neural network model has an extra problem to solve, differentiating between foreground and background information. We have used the Viola-Jones object detection framework for detecting faces and cropped face parts from the image. In this method, the object is detected using Haar feature-based cascade classifiers, and this method was proposed by Paul Viola and Michael Jones [230]. After the cropping process, all parts of the image that do not add any value to the expression classification procedure are removed. Thus, the resulted image does not contain facial parts that do not contribute to the expression (e.g., hair, ears, etc.). This step helped us to extract meaningful features from the image for classification. Image cropping example is shown in Figure 8.

4) Down Sampling: Downsampling, downscaling images with geometric transformations without losing the quality of the image helps us to reduce the amount of data we have. As the size of the image data increases, a lot of memory is required for the neural network to perform their operations. Nowadays machines we can use for image processing applications have limited memory. We applied downsampling to the images without losing the important information to avoid the problem of memory consumption. The final images from all datasets are 256 * 256 pixels. An example of downsampling is shown Figure 9.

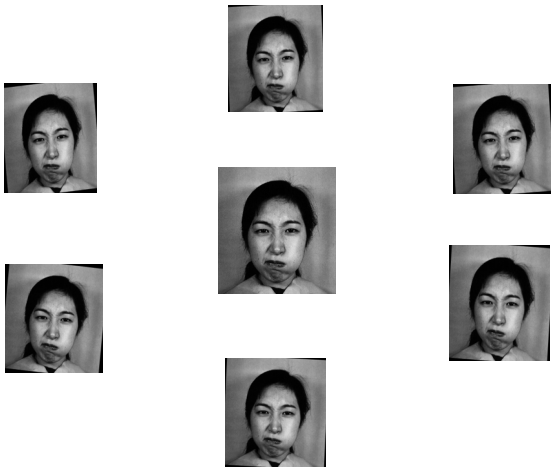


Figure 7. Illustration of synthetic sample generation. The Central image is an original image, and others are synthetic samples of the original image by applying small left and right angle rotation.

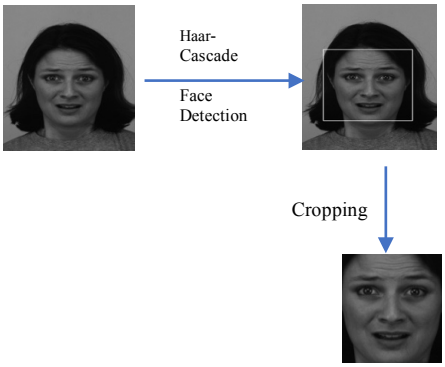


Figure 8. Image Cropping Example

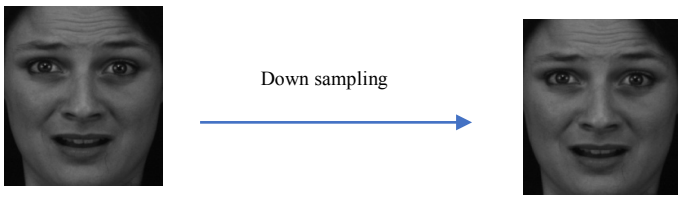


Figure 9. Downsampling Example. Image on the left is 562 * 762 pixels, and image on the right is 256 * 256 pixel

5. Design and CNN Structure Description

My design approach for recognizing emotions from an image is shown in Figure 10. Initially, I have sets of images that are derived from multiple sources. This image data is prepared using specific image processing steps such as grayscale conversion, synthetic image generation, cropping, and downsampling. After data preparation, resulting images will be of good quality and quantity. These prepared images are used for feature extraction and classification. These two processes are done using CNN. This is the critical step in facial expression recognition. Once classification is done, performance is evaluated using 5-fold cross-validation.

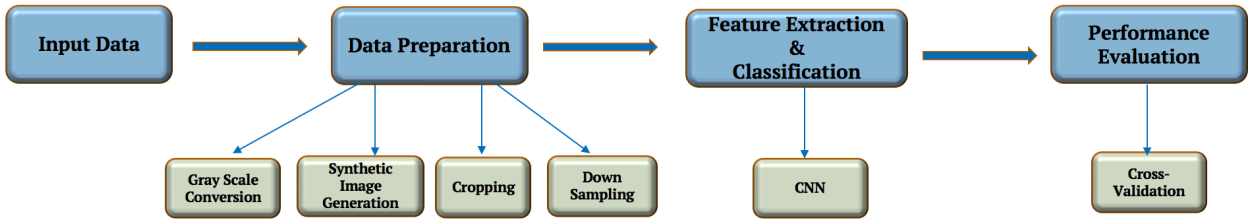


Figure 10. Design Approach

CNNs are known to mimic how the human brain works when analyzing visuals. CNNs compare images piece by piece, which are called features. Each feature is a mini-image, the size of which we can customize. When CNNs are presented with a new image, each feature will pass over the entire image and generate new feature maps (the output received on convolving an image with a feature is called a feature map; also called activation maps).

Another important building block of CNN is the pooling layer which usually succeeds a convolutional layer (convolution and max-pooling demonstration is shown in 11). Its primary advantage lies in limiting the spatial dimensions (Width x Height) of the input data for the following convolutional layer. It does not change the depth dimension of the data. In this layer, large images are downsampled. This layer reduces the number of parameters and computations in the network. This decrease of size leads to some loss of information as well. However, such a loss is useful for our network without compromising on accuracy for the following two reasons:

1. Reduction in size leads to fewer computational overhead for the upcoming layers of the network;

2. It acts against over-fitting.

Much like the convolution operation performed in pooling layer as well, the pooling layer takes a sliding window or a specific region that is passed in stride over the input converting the values into illustrative values. The transformation is either done by using the maximum value from the values visible in the window (called ‘max pooling’) or by taking the average of the values (called 'average pooling'). Usually, max pooling has been preferred over others due to its greater performance characteristics. In our architecture, we have used max pooling and average pooling.

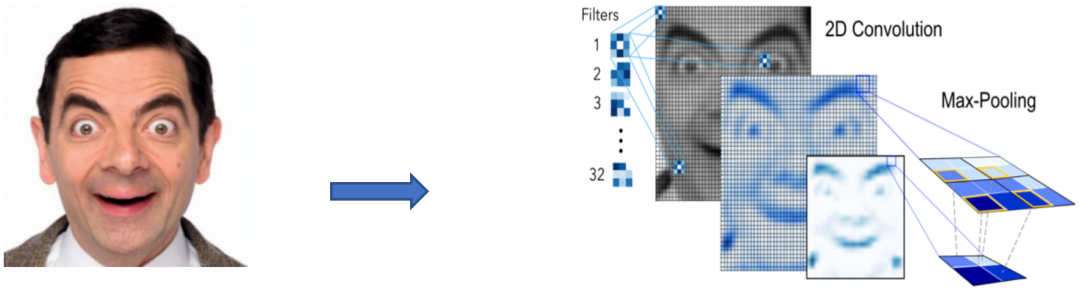


Figure 11. Convolution and max-pooling demonstration [32]

The next small but important layer is Rectified Linear Unit (ReLU). The math used in this layer is very simple: wherever a negative number occurs in the feature map, replace it with zero. It is the most generally deployed activation function for the outputs of the CNN neurons. Detailed working of CNNs can be see in[30]. This layer helps to reduce the likelihood of vanishing gradient. We can have any number of convolutional layers, pooling layers, etc. according to the type of structure we are using. We are using four different combination layers. Finally, a fully connected neural network is used as the classifier. Generally, a CNN is composed of two stages: feature extraction and classification. The feature extraction stage consists of flexible trainable fully convolutional (conv) layers which learn high-level features,

while the classification stage, which is the fully connected layer, deals with various combinations of those features to get the final decisions.

Workflow of general CNN classifier network is shown in Figure 12.

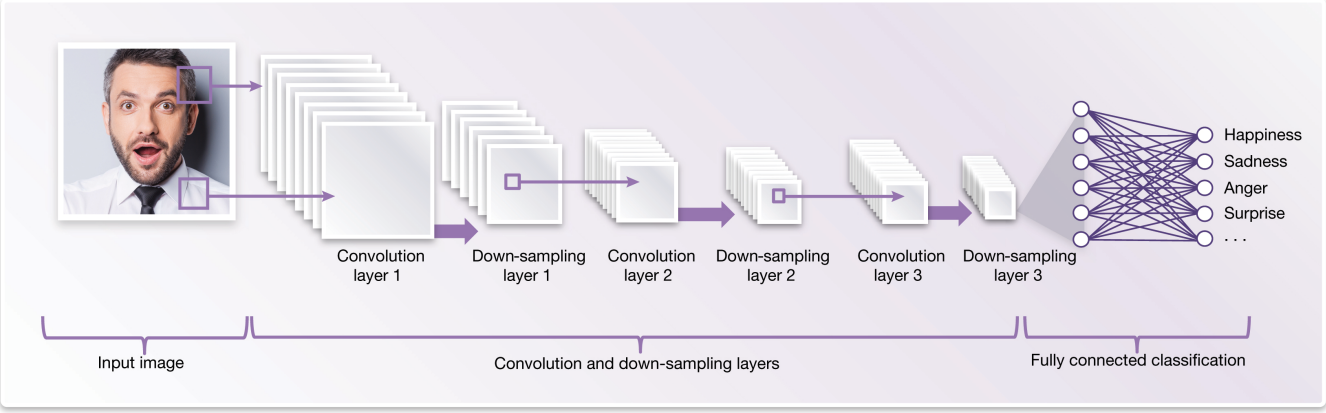


Figure 12. Workflow of CNN in facial emotion recognition [33]

CNNs can achieve better results than traditional neural networks. The three learning stages of feature learning, feature selection, and classification can be performed using a single CNN. They can be trained using many input images and their expression labels. Weights of neural networks in each layer can be adjusted while training by cost optimization step. We have used Adam Optimizer, which is an advanced form of Gradient Descent. The network receives as input a 256*256 grayscale pre-processed image and outputs the probability of each of the seven expressions (Neutral, Happy, Sad, Surprise, Anger, Disgust, and Fear). The expression with the highest value is used as the expression in the image. Once trained, the CNN gives a label number that expresses the basic emotion for the corresponding test image.

We are using four different CNN architectures called ImageNet, Kahou, Tang, and Yu for our work. All four structures are selected from four different research papers published in recent years. The first one is the ImageNet structure [34] which was designed for classifying 1000 classes from the ImageNet dataset. This structure has the most layers of the four structure we have selected; it contains five convolutional layers. Here we have modified it for seven classes. The second one is the structure named Kahou to refer one of its authors [15]. It consists

of three convolutional and pooling layers, with local response normalization (spatial batch normalization) and has both max and average pooling. The next structure is inspired from Tang's structure [11], consisting of three convolutional and pooling layer (max pooling). The main difference of Tang's structure from Kahou's is that Tang is not using local response normalization step in their structure. The last candidate is selected from Yu's structure [14]. They used five convolutional layers in their structure. The Detailed architecture description of structures is shown in Table 1.

Table 1. CNN Structure Description

Network	Category	Layer1	Layer2	Layer3	Layer4	Layer5	Layer6	Layer7	Layer8	Layer9	Layer10	Layer11	Layer12	Layer13
Image-Net	Layer	conv	maxp	lrn	conv	maxp	lrn	conv	conv	conv	maxp	flatten	fc	output
	Kernel	5 * 5 (1,2)	3 * 3 (2,1)	-	3 * 3 (1,1)	3 * 3 (2,1)		3 * 3 (1,1)	3 * 3 (1,1)	3 * 3 (1,1)	3 * 3 (2,1)	-	-	
Kahou	Layer	conv	maxp	lrn	conv	avgp	lrn	conv	avgp	flatten	fc	output		
	Kernel	5 * 5 (1,2)	3 * 3 (2,1)	-	3 * 3 (1,1)	3 * 3 (2,1)		3 * 3 (1,1)	3 * 3 (2,1)	-	-			
Tang	Layer	conv	maxp	conv	maxp	conv	maxp	flatten	fc	output				
	Kernel	5 * 5 (1,2)	3 * 3 (2,1)	4 * 4 (1,2)	3 * 3 (2,1)	5 * 5 (1,2)	3 * 3 (2,1)	-	-					
Yu	Layer	conv	maxp	conv	conv	maxp	conv	conv	maxp	flatten	fc	output		
	Kernel	5 * 5 (1,2)	3 * 3 (2,1)	3 * 3 (1,1)	3 * 3 (1,1)	3 * 3 (2,1)	3 * 3 (1,1)	3 * 3 (1,1)	3 * 3 (2,1)	-	-			

conv, fc, lrn,: convolutional, fully-connected, local response norm and maxp, avgp : max-pooling, average-pooling
kernel: [kernel size] ([stride], [zero-padding]) [27]

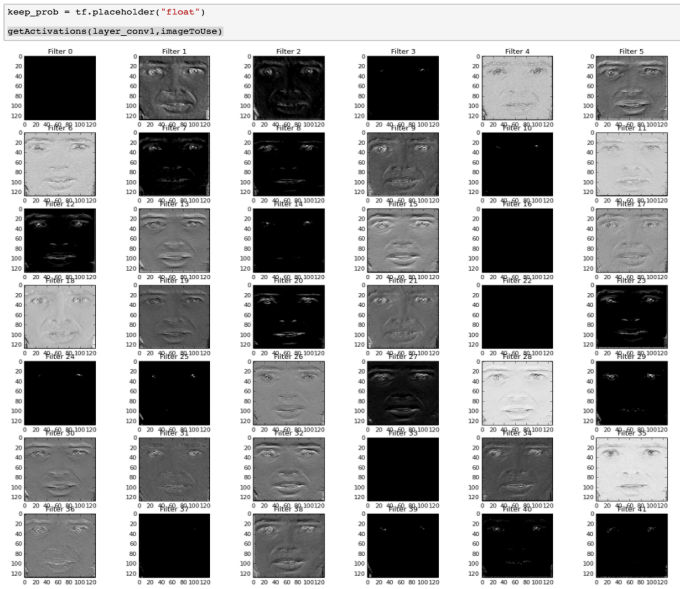
In the following section, we are showing how each image activates the neurons of each convolutional layers. Figure 12, Figure 13, Figure 14, and Figure 15, representing sample activation maps from the first level of convolutional layer, can, from a single image convolution layer, generate any number of activation maps, depending on the size of the kernel. Notice how each filter has learned to activate optimally for different features of the image. The networks we used in this project is a relatively simple network, but the visualization of feature/activation maps can be extended to provide insights into each of our convolutional network.

5.1. Visualization of Activation Maps:

1. ImageNet Network:

When an image is presented to a first convolutional layer of ImageNet, each feature from the input image will convolve with the entire image and generate feature maps. Depending on how many features we consider, there will be multiple numbers of activation maps. Activation maps generated after first convolutional layer is shown in Figure 13.

Image fed to First layer – Input



Activation Maps from layer1

Figure 13. Activation maps from ImageNet

2. Kahou Network

When an image is presented to a first convolutional layer of Kahou network, each feature from the input image will convolve with entire image and generate feature maps. Depending on how many features we consider, there will be multiple numbers of activation maps. Activation maps generated after first convolutional layer is shown in Figure 14.

Figure 14. Activation maps from Kahou

3. Tang Network

When an image is presented to a first convolutional layer of Tang network, each feature from the input image will convolve with the entire image and generate feature maps. Depending on how many features we consider, there will be multiple numbers of activation maps. Activation maps generated after first convolutional layer is shown in Figure 15.

Figure 15. Activation maps from Tang

4. Yu Network

When an image is presented to a first convolutional layer of Yu network, each feature from the input image will convolve with entire image and generates feature maps. Depending on how many features we consider, there will be multiple numbers of activation maps. Activation maps generated after first convolutional layer is shown in Figure 16.

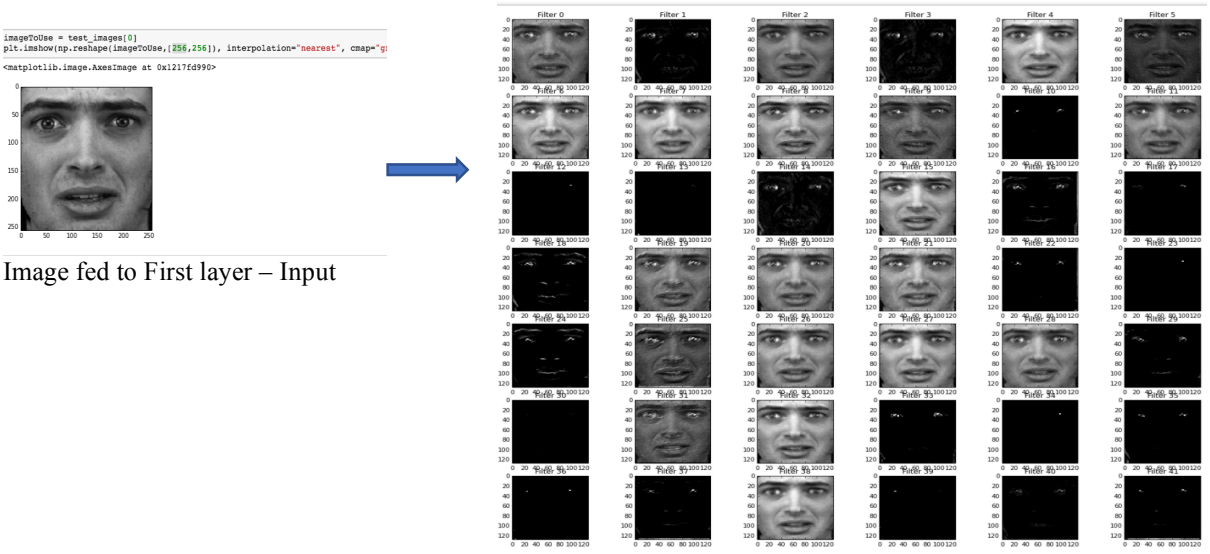


Image fed to First layer – Input

Activation Maps from layer1

Figure 16. Activation maps from Yu

6. Results and Discussion

The expression classification performance evaluation is performed over three datasets, JAFFE [25], KDEF database [26], and combined dataset (JAFFE+KDEF+SFEW) with four different net structures (ImageNet, Kahou, Tang, and Yu). Accuracy is calculated by considering one classifier to classify all learned expressions. Images are tested and trained in batches which helped us to greatly improve the training time.

Firstly, we conducted tests over JAFFE dataset which contains 213 images with seven expressions (neutral, happy, sad, surprise, fear, anger and disgust). Yu's structure shows the highest accuracy at about 61% which is more than 10% higher than the highest state-of-the-art accuracy, not as we expected. The main reason for this low performance is because the number of images in the dataset is fewer for a CNN to give a better result.

Next, we generated synthetic images of JAFFE dataset to increase the number of training images to improve network performance. Synthetic images were generated by applying small rotation on existing images. As a first step, we applied about a degree angle of rotation in a clockwise direction to generate synthetic images. The resulting JAFFE set contained 2130 images instead of 213. As we increased our training samples, accuracy improved from 61% to 100% (three networks gave us 100% accuracy). As we all know, performance of network decreases when we add random noise (artificial rotation) on input data. To examine the impact of rotation on accuracy, we applied a higher degree of rotation on the images of about 15 degrees (+7.5 and -7.5 degrees). As expected, accuracy decreased to 96.325% from 100%. In both cases, Yu's structure performed better over other networks. The performance results and comparison with the state-of-the-art approach is presented in Table 2. The graphical representation of the result is shown in Figure 17.

Table 2. Performance Result- Dataset JAFFE

N/W Structure	JAFFE - Accuracy			
	State-of-the-art Approach	Raw Cropped Image	Synthetic Image Generation 1	Synthetic Image Generation 2
ImageNet	47.78 %	47.58 %	99.38 %	89.23 %
Kahou	**50.28 %**	51.64 %	**100 %**	90.58 %
Tang	49.56 %	59.08 %	**100 %**	94.73 %
Yu	47.89 %	**61 %**	**100 %**	**96.33 %**

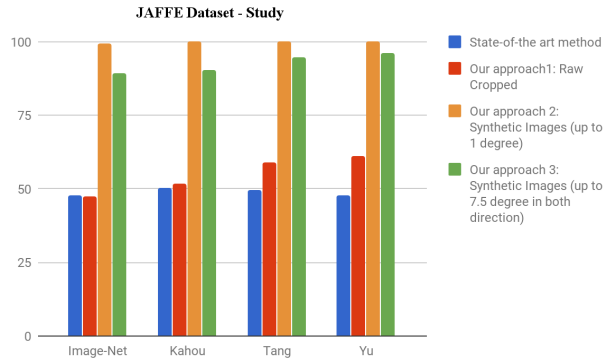


Figure 17. JAFFE Dataset Performance Comparison

The second dataset we have used for performance evaluation is KDEF database [26]. It consists of frontal faced images, half rotated images and fully rotated images with seven expressions. As a first step, we have taken only frontal face images (980 images) and achieved the highest accuracy of 89.58% using ImageNet structure. This result is more than 32% higher than the highest state-of-the art using action. Initially, we used manual cropping technique, then we have used haar-cascade object detection method to identify the face and cropped it accordingly. The result from manual and automatic cropping is different because facial features resulting from haar-cascade cropping method is different from manual.

Secondly, we used both frontal faced and half rotated images (2940 images) for performance evaluation. As expected, accuracy dropped from 89.58 to 80.86% for the dataset with rotated images. This result is obvious because any rotation or translation of images

introduces noises and negatively affect the classifier's performance. Next, we used all three types of images (frontal faced, half rotated, and fully rotated: 4900 images) for performance evaluation. As a result, accuracy further decreased to 77.66%. It proves the fact that as complexity of input data increases in terms of degree of orientation (i.e., dataset contains both angle rotated and frontal faced), network underperforms because network could not identify common feature points in frontal and rotated which contribute towards an expression. Yu's network performed better over all three other networks. The performance results and comparison with the state-of-the-art approach is presented in Table 3. The graphical representation of the result is shown in Figure 18.

Table 3. Performance Result- Dataset KDEF

N/W Structure	KDEF - Accuracy				
	State-of-the art approach	Frontal faced Images		Frontal faced + half rotated	Frontal faced + half rotated + fully rotated
		Manual Crop	Automatic Crop		
ImageNet	**56.86%**	83.16%	**89.58%**	73.86%	69.7%
Kahou	52.28%	81.92%	83.68%	74.26%	71.6%
Tang	54.18%	83 %	81.12%	79.73%	76.4%
Yu	53.21%	**85.38%**	83.06%	**80.86%**	**77.6%**

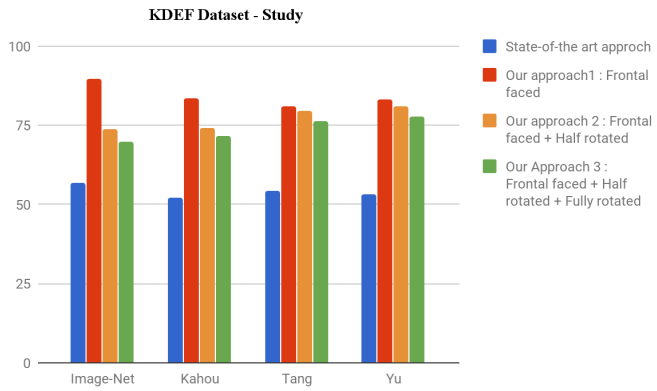


Figure 18. KDEF Dataset Performance Comparison

As a final step of performance evaluation, we combined three different databases (JAFFE, KDEF, and SFEW) to get a generalized result. Each database had images with different characteristics: JAFFE images were captured in grayscale, KDEF images were colored with different orientation, and SFEW images had diverse orientations and illumination as they were captured from movie clips. Usually, there would be a drift in data when it is read in a specific order. We combined disparate sources of data, so there was a greater probability for drift in data. Random shuffling of the data removed these possible drifts. Moreover, when we use a very large number of training data, it is extremely important to randomly shuffle the data so that we do not get entire mini-batches of highly correlated samples. It is highly likely to get biased result during cross-validation if we do not do random shuffling on test data. We applied random permutations on the ordering of images that were taken from different databases to avoid bias and over-fitting. Random permutation makes sure that images are randomly distributed across all the folds during cross-validation. And in cross-validation, each image will validate exactly once. Random permutation and then cross-validation makes sure that the ratio of three dataset size doesn't affect the performance result. The consolidated dataset of 5395 images, Tang achieved the highest accuracy of 71.975%. From the results of the consolidated dataset, Tang's network performed slightly better (0.6% higher) over Yu's network. As dataset complexity increases in terms of orientation degree along with the number of layers in a network, other factors like kernel size, the number of filters, etc. might make a difference in result. The performance results are presented in Table 4. The graphical representation of the result is shown in Figure 19.

Table 4. Performance Result – Combined Data

N/W Structure	Combined (JAFFE +KDEF+ SFEW) Dataset - Accuracy
ImageNet	70.45 %
Kahou	65.725 %
Tang	**71.975 %**
Yu	**71.35 %**

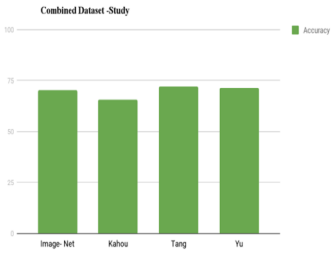


Figure 19. Combined Dataset Performance Comparison

One important observation from the combined dataset performance result is that Tang network performed better than large numbered layer network Yu or ImageNet. One thought about this result is that as dataset complexity increases in terms of orientation degree along with the number of layers in a network, other factors like kernel size, the number of filters, etc. might make a difference in result. To further investigate this thought, we did another performance evaluation where we have taken only rotated images. Rotated images are complex than frontal face because they are covering some of the features that are required for recognizing facial expression. The performance results are presented in Table 5. The graphical representation of the result is shown in Figure 20.

Table 5. Performance Result – Rotated Images

N/W Structure	Rotated Images
ImageNet	78.175 %
Kahou	81.025 %
Tang	83.87 %
Yu	83.15 %

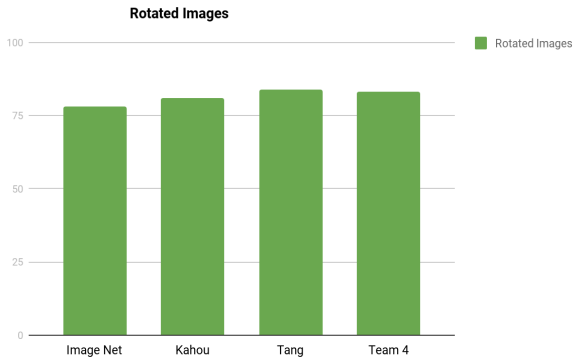


Figure 20. Rotated images Performance Comparison

From the above results, it is evident that if the images are complex in terms of orientation, Tang performs slightly better than Yu network. But the difference in accuracy between Yu and Tang is very small. Further study and experiment should be done to get a clear idea about how other network parameters affect network performance.

6.1. Result Observations

Here is a sample set of images that are successfully detected with their correct expressions in Figure 21. A sample set of images that are not detected is in Figure 22. In Figure 20 and 21, True refers actual emotion each image possesses, and Pred means what network predicted.

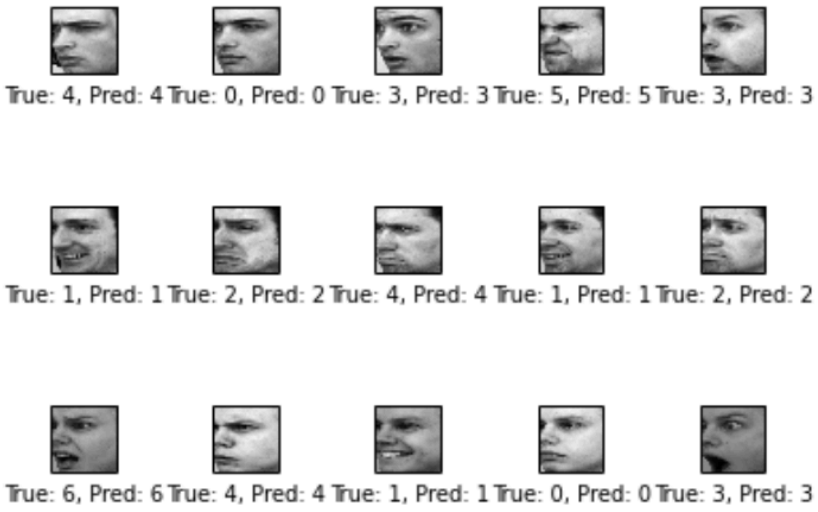


Figure 21. Sample set of images that are successfully detected

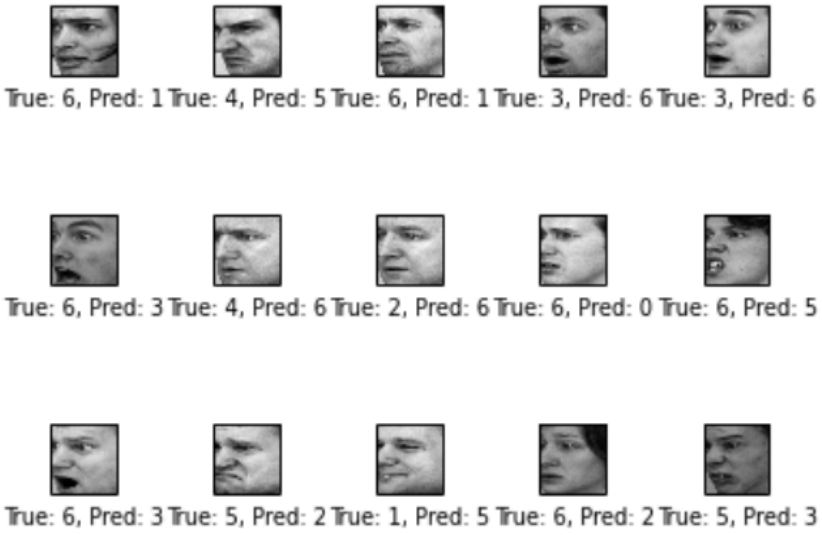


Figure 22. Sample set images that are not detected

Sample output from one of the networks we used is shown in Figure 23. We have used confusion matrix to get a clear idea about 'True Positive' (Number of expressions that are

predicted correctly) rate of each emotion. From the sample output in figure, second column of confusion matrix (i.e., Happy – red arrow) got higher True Positive rate. I.e., out of 35 happy faces, 32 are identified correctly. In confusion matrix, it is also shown that fear (7*7 cell) emotion have the lowest True positive rate (25/35), and it is mostly miss-classified in surprise. Another observation from the figure is about 4th emotion (Surprise – 4*4 cell - blue arrow). Out of 35 surprised faces, 28 are identified correctly, and the rest of the 7 are miss-classified as fear (4*7 cell). From Figure 1 and this miss-classification, it is evident that there is a higher similarity in facial characteristics of fear and surprise. On an average, out of 245 images, 200 are identified correctly. Another observation from figure 23 is that if there is hair fall into the face, our model failed to identify the correct emotion.

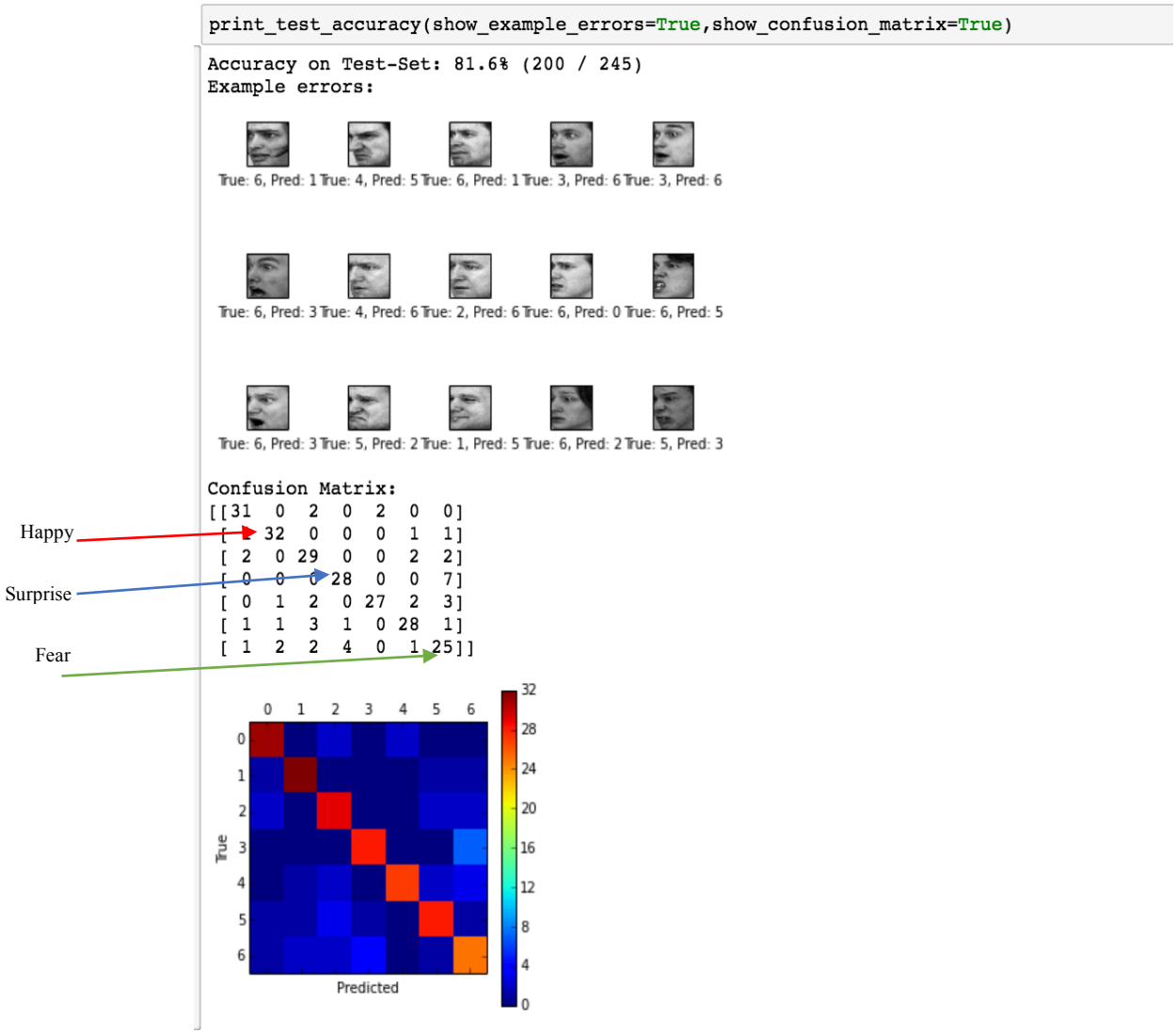


Figure 23. Sample output from a network

7. Conclusion

In this paper, we have implemented a hybrid approach for facial expression recognition where we used a combination of specific image pre-processing steps and convolutional neural networks. We have used four different CNN structures over three datasets; each structure behaved differently with each dataset. Our proposed approach is time efficient and provides better accuracy compared to state of the art approach. Different from previous researches, we combined multiple datasets to get a global picture. Network with large layers performed better over others in the first two experiments. For combined dataset, Tang network works slightly better over Yu. One inference from this result is that for rotated images, Tang network gave better performance. Image pre-processing steps like cropping and downsampling plays a significant role in improving the accuracy and training time of our model. As future work, we wanted to test our model in other databases and perform a cross-database validation. I.e., train the model using one database and test it using another. We also wanted to check the effect of network features like kernel size, number of filters, etc. on the performance.

8. REFERENCES

[1] Li, S. Z. and Jain, A. K., Handbook of Face Recognition. Springer Science & Business Media, 2011.

[2] Sert, M. and Aksoy, N., "Recognizing facial expressions of emotion using action unit specific decision thresholds," in Proc. ASSP4MI '16 Proceedings of the 2nd Workshop on Advancements in Social Signal Processing for Multimodal Interaction, 2016.

[3] Konwar, S. and Borah, S., "ANN based human facial expression recognition in color images," in Proc. International Conference on High Performance Computing and Applications (ICHPCA), 2014.

[4] Brunelli, R. and Paggio, T., "Face Recognition: Features Versus Templates", IEEE Trans On PAMI,1993,15(10), pp 1042- 1052

[5] Lopes, A. T., Aguiar, E., and. Santos, T.O., "A facial expression recognition system using convolutional networks," in Proc. 28th SIBGRAPI Conference on Graphics, Patterns and Images, 2015.

[6] Donato, G., Bartlett, M. S., Hager, J. C., Ekman, P., and Sejnowski, T. J. (1999), Classifying facial actions. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 21(10), 974-989.

[7] Pantic, M., and Rothkrantz, L. J. (2000). Automatic analysis of facial expressions: The state of the art. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 22(12), 1424-1445.

[8] Samal, A., and Iyengar, P. A. (1992). Automatic recognition and analysis of human faces and facial expressions: A survey. Pattern recognition, 25(1), 65-77.

[9] Lajevardi, S. M, Hussain, Z. M. , Automatic facial expression recognition: feature extraction and selection ,Signal, Image and Video Processing March 2012, Volume 6, pp 159–169

[10] Goodfellow, I. J., Erhan, D., Carrier, P. L., Courville, A., Mirza, M., Hamner, B., and Zhou, Y. (2013, November). Challenges in representation learning: A report on three machine learning contests. In Neural information processing (pp. 117-124). Springer

Berlin Heidelberg.

[11] Tang, Y. (2013). Deep learning using support vector machines. CoRR, abs/1306.0239.

[12] Liu, M., Li, S., Shan, S., Wang, R., & Chen, X. (2014). Deeply learning deformable facial action parts model for dynamic expression analysis. In Computer Vision–ACCV 2014 (pp. 143-157). Springer International Publishing.

[13] Fasel, B., Head-pose invariant facial expression recognition using convolutional neural networks, In Multimodal Interfaces 2002 Proceedings in Fourth IEEE International Conference, pp. 529-534

[14] Yu, Z., and Zhang, C. (2015, November). Image based static facial expression recognition with multiple deep network learning. In Proceedings of the 2015 ACM on International Conference on Multimodal Interaction (pp. 435-442). ACM.

[15] Kahou, S. E., Pal, C., Bouthillier, X., Froumenty, P., Glehre, Memisevic, R., and Mirza, M. (2013, December). Combining modality specific deep neural networks for emotion recognition in video. In Proceedings of the 15th ACM on International conference on multimodal interaction (pp. 543-550). ACM. ISO 690

[16] Kim, B. K., Lee, H., Roh, J., and Lee, S. Y. (2015, November). Hierarchical committee of deep CNNs with exponentially-weighted decision fusion for static facial expression recognition. In Proceedings of the 2015 ACM on International Conference on Multimodal Interaction (pp. 427-434).

[17] Abadi, M. , Barham, P. , Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M. , Kudlur, M., Levenberg, J., Monga, R.T, Moore, S., Murray, D.G., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., Wicke, M., Yu, Y., and Zheng, X., TensorFlow: A system for large-scale machine learning, in the Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI ’16), Savannah, GA, USA.

[18] [Online]. Available: http://en.wikipedia.org/wiki/OpenCV, http://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_tutorials.html, (last retrieved on: 11/1/2017).

[19] [Online]. Available: https://medium.com/towards-data-science/from-scikit-learn-to-tensorflow-part-1-9ee0b96d4c85, (last retrieved on: 11/1/2017).

[20] [Online]. Available: http://www.numpy.org/, (last retrieved on: 11/1/2017).

[21] Caleanu, C.D., "Face expression recognition: A brief overview of the last decade," in 2013 Proc. IEEE 8th International Symposium on Applied Computational Intelligence and Informatics (SACI), 2013, pp. 157–161.

[22] Ma, L., Khorasani, K., "Facial expression recognition using constructive feedforward neural networks," Systems, Man, and Cybernetics, Part B: Cybernetics.

[23] Liu, P., Han, S., Meng, Z., and Tong, Y., "Facial expression recognition via a boosted deep belief network," in Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014, pp. 1805–1812.

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

[25] Lyons, M., Akamatsu, S., Kamachi, M., and Gyoba, J., "Coding facial expressions with gabor wavelets," in Third IEEE International Conference on Automatic Face and Gesture Recognition, 1998. Proceedings, 1998, pp. 200–205

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

[27] Shin, M., Kim, M., and Kwon, D.S., "Baseline CNN structure analysis for facial expression recognition," in Proc 25th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN), 2016.

[28] Goeleven E., DeRaedt R., Leyman L., and Verschuere B, The Karolinska directed emotional faces: a validation study, Cognition and Emotion, 22(6), 1094-1118,2008.

[29] Dhal, A.l, Goecke, R., Lucey, S., and Gedeon, T., Static Facial Expressions In The Wild: Data and Experiment Protocol.

[30] Simard, P., Steinkraus, D., and Platt, J. C., "Best practices for convolutional neural networks applied to visual document analysis," in Seventh International Conference on Document Analysis and Recognition 2013 Proceedings, 2003, pp. 958–963.

[31] Viola, P., Jones, M., "Rapid Object Detection using a Boosted Cascade of Simple Features", in 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2001.

[32] [Online]. Available: https://github.com/JostineHo/mememoji, (last retrieved on: 11/1/2017).

[33] [Online]. Available: https://www.embedded-vision.com/sites/default/files/technical-articles/FacialAnalysis/Figure6.jpg, (last retrieved on: 11/1/2017).

[34] Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., and Darrell, T., "Caffe: Convolutional architecture for fast feature embedding". In Proceedings of the ACM International Conference on Multimedia, pp. 675-678,2014.