# A Personal Facial Expression Monitoring System Using Deep Learning

A Thesis

by

JIAQI HU

BS, Missouri Valley College, 2015

Submitted in Partial Fulfillment of the Requirements for the Degree of

MASTER OF SCIENCE

in

COMPUTER SCIENCE

Texas A&M University-Corpus Christi
Corpus Christi, Texas

August 2017

# A Personal Facial Expression Monitoring System Using Deep Learning

A Thesis

by

JIAQI HU

This thesis meets the standards for scope and quality of
Texas A&M University-Corpus Christi and is hereby approved.

August 2017

## ABSTRACT

Facial expression recognition has been a challenge for many years. With the recent growth in machine learning, a real-time facial expression recognition system using deep learning technology can be useful for an emotion monitoring system for Human-computer interaction(HCI). We proposed a Personal Facial Expression Monitoring System (PFEMS).We designed a custom Convolutional Neural Network model and used it to train and test different facial expression images with the TensorFlow machine learning library. PFEMS has two parts, a recognizer for validation and a data training model for data training. The recognizer contains a facial detector and a facial expression recognizer. The facial detector extracts facial images from video frames and the facial expression recognizer distinguishes the extracted images. The data training model uses the Convolutional Neural Network to train data and the recognizer also uses Convolutional Neural Network to monitor the emotional state of a user through their facial expressions. The system recognizes the six universal emotions, angry, disgust, happy, surprise, sad and fear, along with neutral.

## DEDICATION

This thesis is lovingly dedicated to family, especially my grandmother, Bichun You. Their support and constant love have sustained me throughout my life.

## ACKNOWLEDGMENTS

This research is a part of talking head project for Pixel Island Lab. It is working at Pixel Island Lab at Texas A & M University-Corpus Chisti. I would like to thank Dr. Scott A. King, Dr. Alaa Sheta and Dr. Maryam Rahnemoonfar for their unceasing guidance and encouragement. I would express my sincere thanks to all the members from Pixel Island and Bina lab for their sincere support during this work.

# TABLE OF CONTENTS

# LIST OF TABLES

## LIST OF FIGURES

# CHAPTER 1

# INTRODUCTION

Deep learning is one of the fastest-growing and most exciting areas in machine learning. With recent advancements in graphics processing unit, it is possible to use Deep Learning for real-time applications. Emotions are an incredibly important aspect of human life, and play an important role in human interaction. Facial expressions represent the emotion of a person and it can give an indication of the emotional response of a person to the interaction with a computer. So detecting facial expressions can help create a better (i.e. less frustrating) user experience.

## 1.1 Motivation

In the 21st century, HCI products, such as Siri from Apple, Echo from Amazon and Cortana from Windows, became more and more popular in the world. The recent successes of AlphaGo brought machine learning to the world. AlphaGo uses a Monte Carlo tree search algorithm to find its moves based on the knowledge gathering from a pre-train data, which trained by artificial neural network (ANN) [32].

The successful use of machine learning in Go (game) encourages us to design a facial expression recognition system that can be used for HCI and solve facial expression recognition problem with machine learning.

## 1.2 Objective

In this work, the main goad is to design a python-based Personal Facial Expression Monitoring System (PFEMS). PFEMS uses a custom Convolutional Neural Network (CNN) model which is used to train facial expression images with the TensorFlow machine learning library. PFEMS can be used to detect the change of human facial expression during the interaction to allow changing how a system responds to the user. There are two parts in PFEMS. One is a training program. Since Convolutional Neural Network technique requires at least hours for training, training phase needs to be separate from the front end monitoring system. We design a proper Convolutional Neural Network for PFEMS. The other one is facial detection and recognition program. This program accesses an HD/FHD camera for input video and applies facial detection on video frames. Next, it uses the facial images from facial detection and the output from the training program for the facial expression recognition. The output of the PFEMS is single expression labels for the corresponding facial image. We also need to determinate the proper size and type for the input images used in the training program.

## 1.3 Challenges

Facial expression recognition problem in a real-time environment has two important aspects: accuracy and efficiency. Efficiency is measured in terms of time complexity, computational complexity and space complexity. Current methods that had high accuracy also had very high computational complexity or space complexity. Besides the methodology, there are a few other factors that will effect the accuracy, such as subjectivity, occlusion, pose, low resolution, scale, variations in illumination level and identification of baseline frame.

In machine learning, how to prepare a proper dataset is a big challenge. A proper dataset needs to have quantity and quality. According to the complexity of the problem that needs to be solved by deep learning,affect the amount of data varied. However, 100,000 instances are considered an initial amount of data for solving problems using deep learning. Even though the facial expression recognition problem has been studied over 20 years, there are not many open databases available. Most of the open databases are gray-scale images and different databases use different methods to label the images. The largest open database for facial expressions contains less than 30,000 images for 7 classes and it is small based on the complexity of the facial expression recognition problem. How to prepare a proper dataset is one of the main challenges. Another challenge in machine learning is how to create a proper Convolutional Neural Network for facial expression recognition purpose.

The facial detector needs to detect and extract the facial images from frames in the video. When the target, human face, is moving, the camera will be refocused. So the extracted facial images may not be the same quality and size. We need to insure those images have similar quality as much as possible.

# CHAPTER 2

# BACKGROUND AND PRIOR WORK

Facial expression recognition is the process of identifying human facial emotion, researchers started doing research in this area in the 1970s and they had tremendous advances since then.

## 2.1 Background

Psychological research has classified six facial expressions, which correspond to distinct universal emotions: anger, disgust, happiness, surprise, sadness and fear [4]. There are many techniques often used for facial expression recognition, such as deep learning [35][14][7], Facial Action Unit Coding System(FACS) [13][20], linear discriminant analysis [24], Local Parametric model [4], Gaussian Process Classification [10] etc. FACS [12], which was developed by Ekman and Friesen, has been widely used to describe facial behaviors and help clinical psychologists to categorize the physical expression of emotions. It has also been widely used for facial expression recognition or related fields. Deep Learning is a subfield of machine learning concerned with algorithms that are inspired by an ANN [30]. It has been often used for object classification research and applications.

### 2.1.1 Convolutional Neural Networks

CNN is a type of feed-forward ANN in which the connectivity pattern between its neurons is inspired by the organization of the animal visual cortex. It is attractive for many deep learning tasks like image classification, scene recognition, and natural language processing [35] [14] [7] [31]. In 1962, Hubel and Wiesel [15] showed some individual neuronal cells in the brain responded only in the presence of edges of certain orientation in an experiment. They found that a columnar architecture was organizing those neurons and those neurons were able to produce visual perception together. CNN accepts input data, such as audio, video and image, with an optional dimension as the input layer and generates an output layer that has a vector of highly distinguishable features related to the pre-classified category. As a type of supervised learning, CNN used labeled training data which consist a set of training examples [27]. Like other supervised learning algorithms, CNN analyzes the training data and inferred a method that can be used for mapping new examples. CNN normally requires a large amount of training data and the inference accuracy can be improved by adopting more data. The inference accuracy in CNN also can be improved by adopting deeper and wider network. However, larger training or/and more complicated models will result in longer training time. There are a few distinct types of layers used commonly in CNNs [22], such as convolutional layer, pooling layer, ReLu (Rectified Linear Units) layer, fully connected layer and loss layer.

Convolutional layer is the core building block of a CNN based algorithm and it can be applied a varying number of times based on the methodology. The primary purpose of a convolution layer is to extract features from the input image.

Pooling layer is a form of non-linear down-sampling. There is two common type of pooling layers: max-pooling layer and average-pooling layer. It partitions the input images into a set of non-overlapping N by N rectangles, N can be any divisor of the image size and outputs the maximum/average value from each sub-region. Max-pooling works better on highlighting images.

ReLU is one of the most popular types of nonlinearity to use in neural networks that is applied after the convolutional layer and before max pooling. It replaces all negative pixel values in the feature map by zero. It normally used after convolutional layer.

Fully connected layers, which is only applied at the end of the network, take an input volume which is output from the previous process and outputs an N dimensional vector where N is the number of classes that the program had to choose from. It is one of the cheapest ways of learning non-linear combinations of these features.

A system takes a one channel image as input and goes through all the steps shown in Figure 1 and generates a vector of highly distinguishable features related to object classes as an output layer. For example, there are two classes of objects need to be identified and the output will be a vector [ x, y ] where x and y are the weights for two classes and the system will identify the image based on the highest weights.

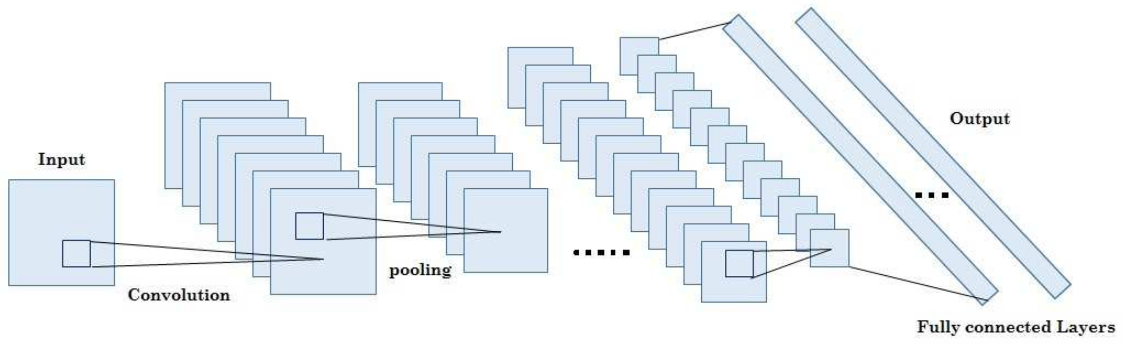

**Figure 1.** A CNN Model Example.

### 2.1.2 OpenCV

OpenCV is an open source computer vision library that is designed for computational efficiency and has a strong focus on real-time applications [5]. OpenCV has been used in a lot of computer vision application, such as monitoring system [26], object tracking [11], security detection system [3], medical image noise reduction [6] and manufacturing inspection system [6]. It was also used to design a facial detector. OpenCV contains a cascade classifier function. A cascade classifier is an object detection framework to provide competitive object detection rates in real-time, it was proposed in 2001 by Paul Viola and Michael Jones [34] and improved by Rainer Lienhart [21]. It can be used for different types of object classification, but it was motivated primarily by the problem of face detection. Cascade classifier has a very high detection rate (true-positive rate) and a very low false-positive rate. In our system, we needed facial detection and image cropping process on frame every 0.5 seconds. OpenCV cascade classifier can process fast enough to meet our need.

### 2.1.3 TensorFlow

TensorFlow [2] is an open source machine learning library that allows user to deploy computation to signed processing units with a single API. The TensorFlow provides a high-level API for different types of layers in neural network, such as convolutional layer, pooling layer and fully connected layer. It also provides methods that adding activation function and applying dropout regularization.

## 2.2 Prior Work

### 2.2.1 Local Parametric Model

Black and Yacoob designed a system for tracking and recognizing rigid and non-rigid facial motions using local parametric models of image motion in 1995 [4]. Their experiment had 92% accuracy rate on forty subjects from 128 expressions and 78.5% accuracy rate on 36 video-clips from talk shows, news, and movies.

### 2.2.2 Facial Action Coding System

Mihai Gavrilescu developed a fully integrated neural-network based facial expression recognition system based on FACS [13]. He used 27 out of 44 action units (AUs) from FACS to detect changes in localized facial features. His method separated face feature into 5 subfeatures: eye features, brow features, cheek features, lip features and wrinkle features. And then he used a classifier to classified the AU for each component. The output of the classifiers will feed to a 4-layer neural network which will take the decision of the final AU map and the corresponding recognized emotion. In terms of AU classification, the Classification Rate is 0.981 with MMI database [25] and 0.978 with Cohn-KanadeFacial Expression (CK) database [18]. Their experimental results have shown that this type of approach offers significantly better results than any other facial expression recognition systems in terms of AU classification.

James Lien, et al. developed a computer vision system that automatically recognizes individual action units or action unit combinations in the upper face using Hidden Markov Models (HMMs) in 1998 [20]. Their approach to facial expression recognition is based on FACS. They separate expressions into upper and lower face action and use the upper face for the recognition. They use three approaches to extract facial expression information on the upper face: (1) facial feature point tracking, (2) dense flow tracking with principal component analysis (PCA), and (3) high gradient component detection. The recognition results of the upper face expressions using feature point tracking, dense flow tracking, and high gradient component detection are 85%, 93%, and 85%, respectively.

### 2.2.3 Deep Learning

Tong Zhang, et al. proposed a novel deep neural network (DNN) driven feature learning method for multi-view facial expression recognition (FER) in 2016 [35]. DNN is an ANN with multiple hidden layers of units between the input and output layers. In their method, scale invariant feature transform (SIFT) features corresponding to a set of landmark points are first extracted from each facial image. The feature matrix that is consisting of the extracted SIFT feature vectors is used as input data input for the DNN model to learning optimal discriminative features for expression classification. Their DNN model employs several layers to characterize the corresponding relationship between the SIFT feature vectors and their corresponding high-level semantic information. They used two non-frontal facial expression databases, namely BU-3DFE [19] and Multi-PIE [28] to evaluate the effectiveness of their method. The experimental results show that their algorithm outperforms the state-of-the-art methods. Their method achieves 80.1%.

Yanan Guo, et al. proposed a scheme termed Deep Neural Networks with Relativity Learning (DNNRL) for Facial Expression Recognition in 2016 [14]. According to sample importance, DNNRL treats samples differently. DNNRL consists of three convolutional layers, four pooling layers, three Inception layers, and one fully connected layer. By using the exponential triplet loss, DNNRL can directly learn a mapping from original images to a Euclidean space, where relative distances correspond to a measure of facial expression similarity. In their experiment, they used the FER2013 [16] dataset and the SFEW2.0 [1] dataset to demonstrate the effectiveness of the their proposed method. They had overall 69.85% accuracy on FER2013 test set while only trained on FER2013 training set and 73.71% accuracy on FER2013 test set while trained on both FER2013 training set and SFEW2.0.

Peter Burkert, et al. proposed a convolutional neural network (CNN) architecture for facial expression recognition in 2015 [7]. [7].The proposed architecture is independent of any hand-crafted feature extraction and performs better than the earlier proposed convolutional neural network based approaches. Visualizing the automatically extracted features which have been learned by the network to provide a better understanding of those features. They used Extended CK and MMI Facial Expression Database for the quantitative evaluation. On the CK set, the current state of the art approach, using CNN achieves an accuracy of 99.2%. For the MMI dataset, currently the best accuracy for emotion recognition is 93.33%. The proposed architecture achieves 99.6% for CK and 98.63% for MMI.

Minchul Shin, et al. proposed a baseline CNN structure and image preprocessing methodology to improve facial expression recognition algorithm using CNN in 2016 [31]. They investigated four network structures that are known to show good performance in facial expression recognition. They also investigated the effect of input image preprocessing methods. They used five types of data as input: raw, histogram equalization, isotropic smoothing, diffusion-based normalization, difference of Gaussian. They trained 20 different CNN models, 5 data input types for 4 networks, and verified the performance of each network with test images from five different databases, FER2013, SFEW2.0 [1], CK+(extended Cohn-Kanade)/citelucey2010extended, KDEF (Karolinska Directed Emotional Faces) [23], and Jaffe [17]. The experiment result showed that a three-layer structure consisting of a simple convolutional and a max pooling layer with histogram equalization image input was the most efficient.

### 2.2.4 Combined models

Kwok-Ping Chan, et al. applied Correlated topic models (CTM) and Nayes Modeling to facial expression recognition in 2015 [9]. Correlated topic models is a type of topic models which is very close to the Action Unit. The base unit that they use for quantizing facial feature is words. Their preliminary results showed that by combining the shape transition words and facial appearance words with the CTM models obtained recognition rate of 84.1% by suitably choosing the SVM kernels and their parameters. They further applied a Bayes modeling to model the relationship between images in the expression sequence, and the result further improved to 91.3%.

Yu-Dong Zhang, et al. proposed a new facial expression recognition system based on Biorthogonal Wavelet Entropy, Fuzzy Support Vector Machine, and Stratified Cross Validation in 2016 [35]. Their three state-of-the-art method achieved an overall accuracy of 96.770.10% over 700 images. However, their dataset only contained male and female Asian and it is hard to tell the effectiveness of the system while applied on other races.

Adin Ramirez Rivera, et al. proposed a novel local feature descriptor,local directional number pattern (LDN), for face and expression recognition in 2013 [29]. LDN encodes the structure of a local neighborhood by analyzing its directional information. They computed the edge responses in the neighborhood, in eight different directions with a compass mask. Then, from all the directions, they chose the top positive and negative directions to produce a meaningful descriptor for different textures with similar structural patterns. Their descriptor uses the information of the entire neighborhood, instead of using sparse points for its computation like local binary patterns. They used CK, CK+, Jaffe, MMI and the CMU pose, illumination, and expression (PIE) database(CMU-PIE) [33] to test their descriptor for 6-class facial expression recognition and 7-class facial expression recognition. Their results show that under the same methodology, 6-class facial expression recognition has a higher accuracy rate than 7-class facial expression recognition for most of the expression and the overall. The best result they had is 99.50% overall accuracy rate by using CK and the worst they had is 63.30% overall accuracy rate by using CK+. Their method showed reliable and robust under time lapse and illumination variations.

Aruna Chakraborty, et al. presents a fuzzy relational approach to human emotion recognition from facial expressions and its control in 2009 [8]. The proposed scheme uses external stimulus to excite specific facial expression in human subjects whose facial expressions are analyzed by segmenting and localizing the individual frames into regions of interest. Features at the eye, mouth, and eyebrow are extracted from the localized regions, fuzzified, and mapped onto an emotion space by employing Mamdani-type relational models. They used India and America movies to create their data base. Their fuzzy relational approach to emotion detection from facial feature space to emotion space has a classification accuracy of around 90%. However, they used movie clips as the experiment database which consists of actor acting. Most of the facial expression is not genuine. They should test their algorithm on a better database.

# CHAPTER 3

# PROPOSED SYSTEM DESCRIPTION

## 3.1 System Requirement

PFEMS run on Ubuntu machine and it works for both ubuntu 14.04 and ubuntu 16.04. The system requires at least one Nvidia graphics processing unit card(GPU) card with CUDA Compute Capability 3.0 or higher, such as Nvidia GeForce GTX 900 series and Nvidia GeForce GTX 10 series. The system needs to install or upgrade to at least the following library versions to guarantee it will operate properly: TensorFlow 1.0, OpenCV 2.4 , CUDA Toolkit 7.0 and cuDNN v4.1. The best performance is obtained using the combination of CUDA Toolkit 8.0 and cuDNN v5.1.

## 3.2 System Structure

PFEMS contains two parts: a recognizer for validation and a pre-train program for data training. The recognizer contains a facial detector and a facial expression recognizer. The pre-train program uses a CNN deep learning training model. Figure 2 shows the system design.

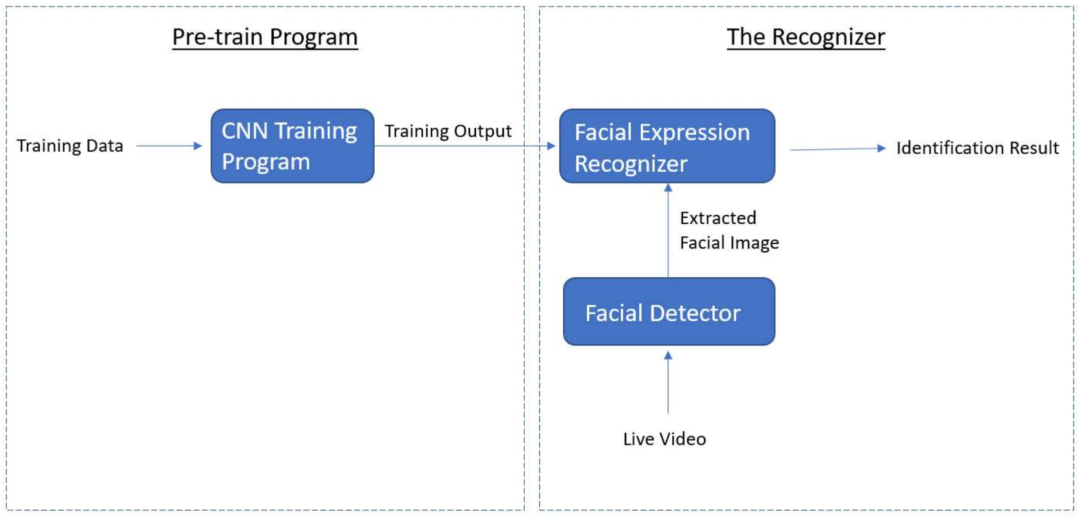

**Figure 2.** PFEMS overall system design

The pre-train program uses a set of labeled facial images to train on a CNN deep learning training model. Figure 3 shows the required input and the output files type for the pre-train program. Images collected from user is option and chapter 4 will discuss this option. The recognizer uses an RGB camera to obtain real time video. The facial detector detects the largest face and crops a facial image from the frame. The facial image is used as input for the facial expression recognizer. The facial expression recognizer processes the facial image and shows the result on the command line. The inputs of the recognizer are an RGB real time video from a camera and checkpoint file from the pre-train program. The output of the facial detector is a fixed size image which is an input for the facial expression recognizer that uses it for the identification of the facial expression. The facial detector reads frames from the live video and uses OpenCVs cascade classifier to detect faces from the frames. While faces are being detected, a square facial image of the largest face is extracted from frame. Figure 4 shows an example detection of the facial detector.

If the facial expression recognizer is available, then the facial image will be re-sized to a fixed size and sent to the facial expression recognizer for evaluation. Figure 5 shows the required input and the output from the facial expression recognizer. The facial detector will not send the image to the facial expression recognizer until the facial expression recognizer finishes processing the current image.

The facial expression recognizer requires the complete result from the pre-train program to validate the facial image.

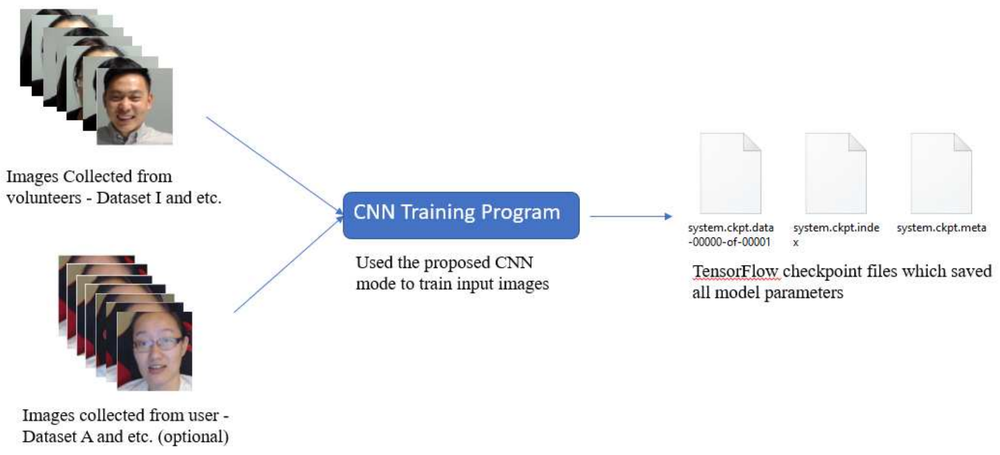
Images Collected from volunteers - Dataset I and etc.
Images collected from user - Dataset A and etc. (optional)
CNN Training Program
Used the proposed CNN mode to train input images
system.ckpt.data-00000-of-00001
system.ckpt.index
system.ckpt.meta
TensorFlow checkpoint files which saved all model parameters

**Figure 3.** The facial expression recognizer requires the complete result from the pre–train program to validate the facial image.

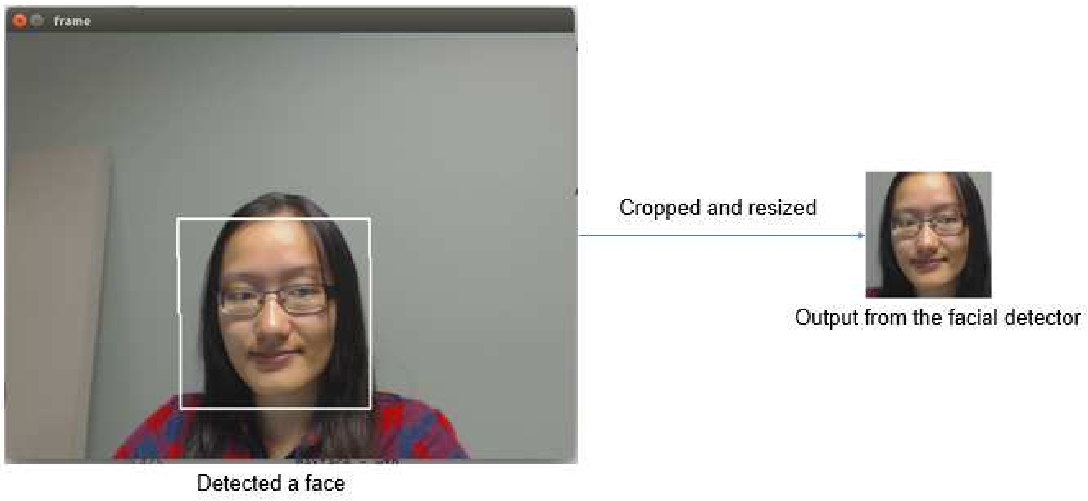

**Figure 4.** Facial Detector - detect and extract image from video.

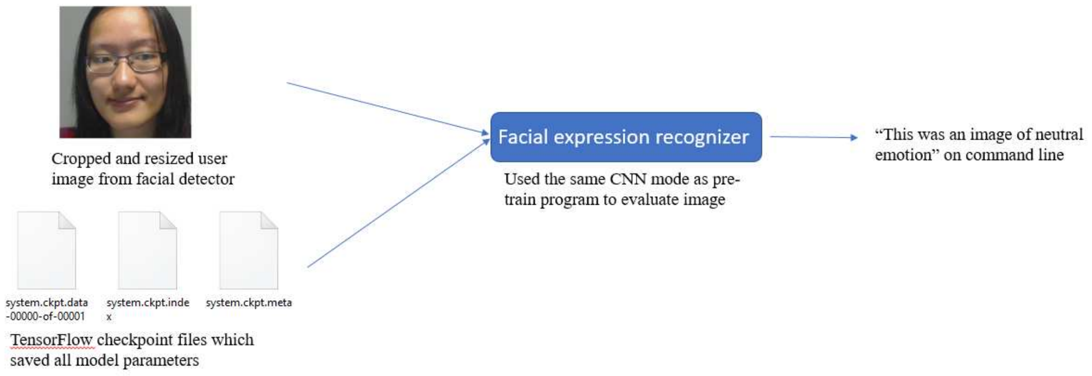
Cropped and resized user image from facial detector
system.ckpt.data-00000-of-00001
system.ckpt.index
system.ckpt.meta
TensorFlow checkpoint files which saved all model parameters
Facial expression recognizer
Used the same CNN mode as pre-train program to evaluate image
"This was an image of neutral emotion" on command line

**Figure 5.** Facial Expression Recognizer design. It uses checkpoint files from pre-train program to evaluate the images extracted from facial detector.

The facial expression recognizer uses the pre-trained data for the facial expression identification. The dataset is trained in the pre-train system using deep learning technology and the results are stored in checkpoint files. While the facial expression recognizer receives an image input, it converts the input image into a tensor and uses the inference from the pre-train program to speculate the input image.

The output from the facial expression recognizer is a single expression label and it is the final system output. The facial detector was developed with OpenCV in python and the facial expression recognizer was developed with Tensorflow using the deep learning technique in python. Besides the six universal emotions: anger, disgust, fear, happiness, sadness and surprise, the method also recognizes a neutral face. The proposed system has four options for user. The first option is a train-only option with command *python system.py -t train_file checkpoint_name input_size.* The checkpoint_name is the name of the checkpoint file without an extension and the name of a folder that is used to save the tensorboad file. The second option is validation-only option with command *python system.py -v validation_ file checkpoint_name input_size.* This option requires a checkpoint file with a completed training data for validation purpose. The input can be either a tfrecords file or an image. The third option is train-validate option with command *python system.py -b train_file validation_file checkpoint_name input_size.* This option is a combination of option one and two. The last option is live option with command *python system.py -l checkpoint_name input_size.* This option requires a checkpoint file and an RGB camera for live video input.

## 3.3 CNN Model Architecture

The method used in the system is a custom CNN method with 18 layers to train and validate facial images. In the training phase, learning rate is set to 1e-4, batch size is set to 20, number of epochs is set to 60 and drop out is set to 50%. All input data is divided into groups of 20 and each group need to be trained 60 times. All images that use for training purpose need to be label manually and be converted into a binary file, tfrecords which is recommended input format for TensorFlow, with labels as input dataset. Test dataset/data can be a single tfrecords file or a single image file. A image converter can convert all labeled images into tfrecords file and it provides image re-size function and image channel transform function. The CNN train program provide re-size function and the input data can be re-sized to a fixed size after the input layer and before the first convolutional layer.

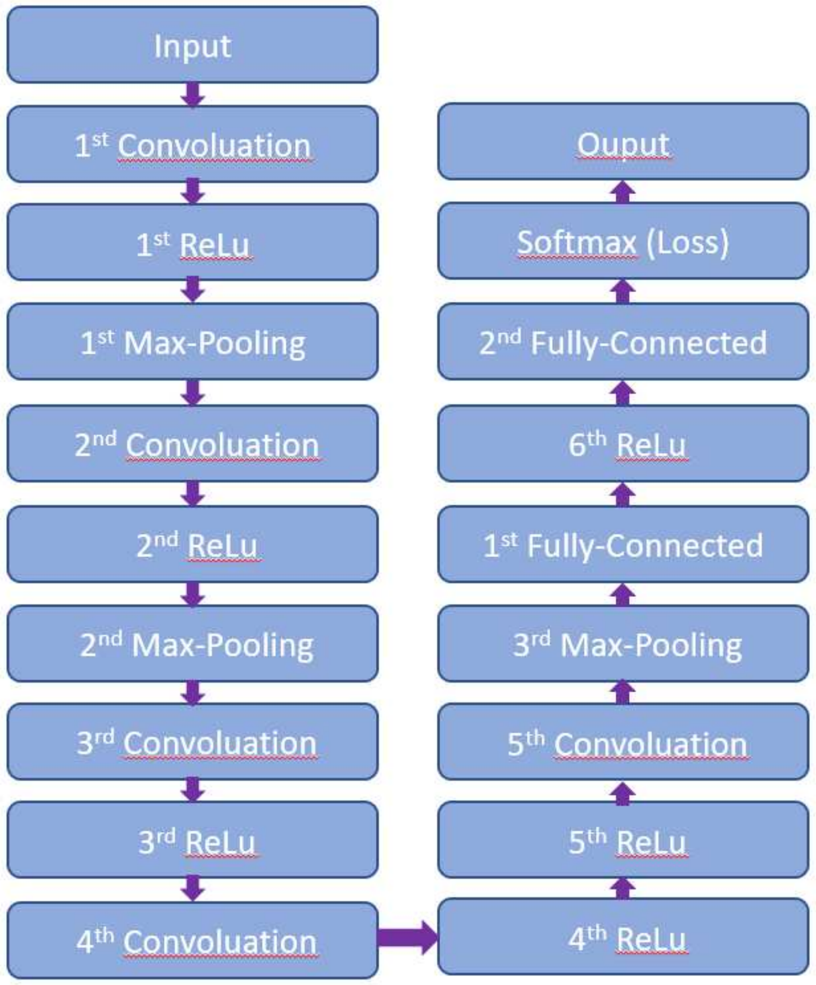

**Figure 6.** The architecture of the custom CNN model with layers.

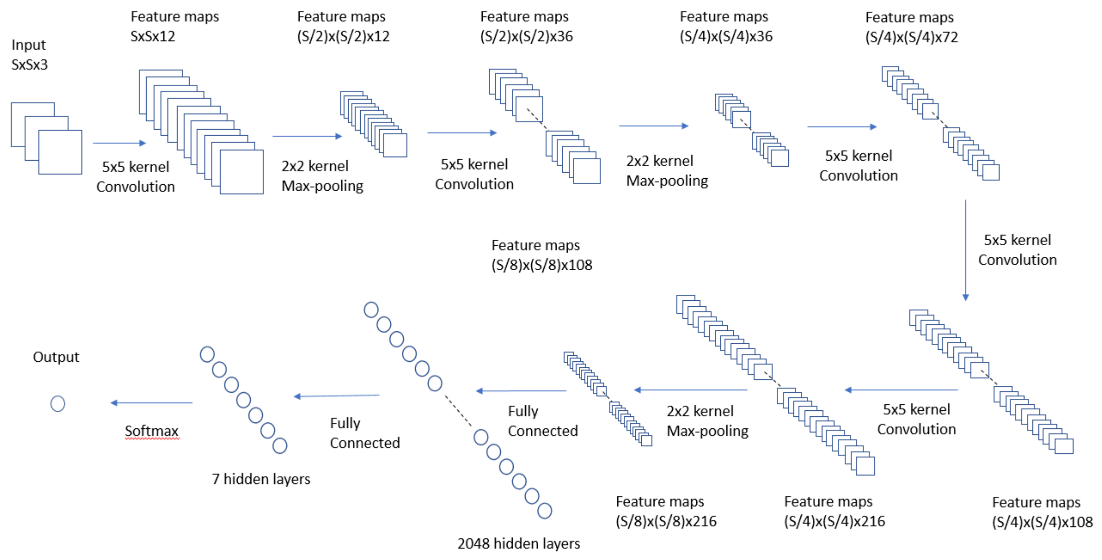

**Figure 7.** The architecture of the custom CNN model with future maps. The non-linear activation functions (ReLu) does not shows in the figure.

Figure 6 shows the architecture of the custom CNN model used in layers order for the pre-train program. Figure 7 shows architecture of the custom CNN model with future maps. Firstly, the CNN train program reads the training data to a list of tensor and shuffle tensors before applying CNN, so the output may be slightly different due to the training order of the images. Secondly, applying the first convolutional layer, which will compute 12 features for each 5x5 patch, on the input tensor. Next, a Relu layer is used on the output of the first convolutional layer which will increase the nonlinear properties of the decision function and the overall network without affecting the receptive fields of the convolution layer. Then, max pooling with 2x2 filter is applied to reduce the tensor size to 1/2 of the original tensor.

Subsequently, the second convolutional layer is applied on the reduced tensor which computes 36 features with a 5x5 filter on the previous tensor. After that, the second Relu layer is used on the output of the second convolutional layer. Next, the second max pooling with 2x2 filter is applied to reduce the tensor size to 1/4 of the original tensor.

And then, the third convolutional layer is applied on the reduced tensor which computes 72 features with a 5x5 filter. After that, the third RelU layer is used on the output of the third convolutional layer. Then, the fourth convolutional layer is applied on the reduced tensor which computes 108 features with a 5x5 filter. Next, the fourth Relu layer is used on the output of the fourth convolutional layer. Subsequently, the fifth convolutional layer is applied on the reduced tensor which computes 216 features with a 5x5 filter. Then, the fifth Relu layer is used on the output of the fifth convolutional layer. After that, the third max pooling with 2x2 filter is applied to reduce the tensor size to 1/8 of the original tensor.

And then, a fully connected layer with 2048 neurons is used to allow processing on the entire image. Subsequently, the sixth ReLu layer is applied on the first fully connected layer and tried to increases the nonlinear properties and applied drop out 50% to avoid overfitting problem during training. Next, applies the second fully connected layer for the final output. Finally, the readout layer,softmax layer, is the last layer that gives the class predictions that produce the final output of the network.

# CHAPTER 4

# EVALUATION AND RESULTS

## 4.1 Dataset

All samples are assigned to one of the seven facial expression categories and being labeled before use. Table I shows the seven facial expression categories and the corresponded labels.

| | Angry | Disgust | Fear | Happy | Sad | Surprise | Neutral |
|---|---|---|---|---|---|---|---|
| Label | 0 | 1 | 2 | 3 | 4 | 5 | 6 |

Table I. Seven facial expression categories and the corresponded labels.

### 4.1.1 Samples Collected From Four Volunteers

A dataset, named Dataset I, is collected from four volunteers, two females and two males. Each volunteer filmed approximately two minutes of video for each emotion with a full HD webcam. A two minutes video with 60 frames per second can extract at least 7000 images with 1920x1080 pixels. After clearing up useless images, there are 14011 useful images. 9809 images, which is 70% of Dataset I, is used for training and 4202 images, which is 30% of Dataset I, is used for validation. After applying on the facial detector without re-size, the smallest cropped facial image is 144x144 pixels. Since the machine learning requires all the input data to be the same size, all facial images are resized to 144x144 pixels. All images are properly labeled after being cropped and resize. Figure 8 shows some example images samples from dataset I. Table II shows the samples quantity in each category for both training and validation. Table III show the percentage of each category over training samples or validation samples.

| | Angry | Disgust | Fear | Happy | Sad | Surprise | Neutral | Total |
|---|---|---|---|---|---|---|---|---|
| Training samples | 1373 | 848 | 1291 | 2223 | 1101 | 1153 | 1820 | 9809 |
| Validation samples | 586 | 362 | 557 | 952 | 472 | 494 | 779 | 4202 |
| Total samples | 1959 | 1210 | 1848 | 3175 | 1573 | 1647 | 2599 | 14011 |

Table II. Samples quantity in each category. In each category, 70% of the samples use marked as training samples and 40% of the samples marked as validation samples.

| | Angry | Disgust | Fear | Happy | Sad | Surprise | Neutral | Total |
|---|---|---|---|---|---|---|---|---|
| Training samples | 14.00% | 8.65% | 13.16% | 22.66% | 11.22% | 11.75% | 18.55% | 100.00% |
| Validation samples | 13.95% | 8.61% | 13.26% | 22.66% | 11.23% | 11.76% | 18.54% | 100.00% |

Table III. The percentage of samples in each category over training samples/validation samples.

In order to find out the proper image size for training purposes, we modify the images from dataset into different images sizes. Dataset II has the same images in Dataset I with 128x128 pixels images, Dataset III contains 96x96 pixels images, Dataset IV contains 48x48 pixels images and Dataset V contains 32x32 pixels images. To find out the effect of the image type, we converted Dataset I into gray-scale images and named Dataset VI. Table IV shows the samples detail for Dataset I to VI. All image samples converted into different image size and image type for comparison purpose.

| | Dataset I | Dataset II | Dataset III | Dataset IV | Dataset V | Dataset VI |
|---|---|---|---|---|---|---|
| Image Size | 144x144 | 128x128 | 96x96 | 48x48 | 32x32 | 144x144 |
| Image Type | RGB | RGB | RGB | RGB | RGB | Gray |
| Image Channel | 3 | 3 | 3 | 3 | 3 | 3 |

Table IV. Dataset I to IV specification.

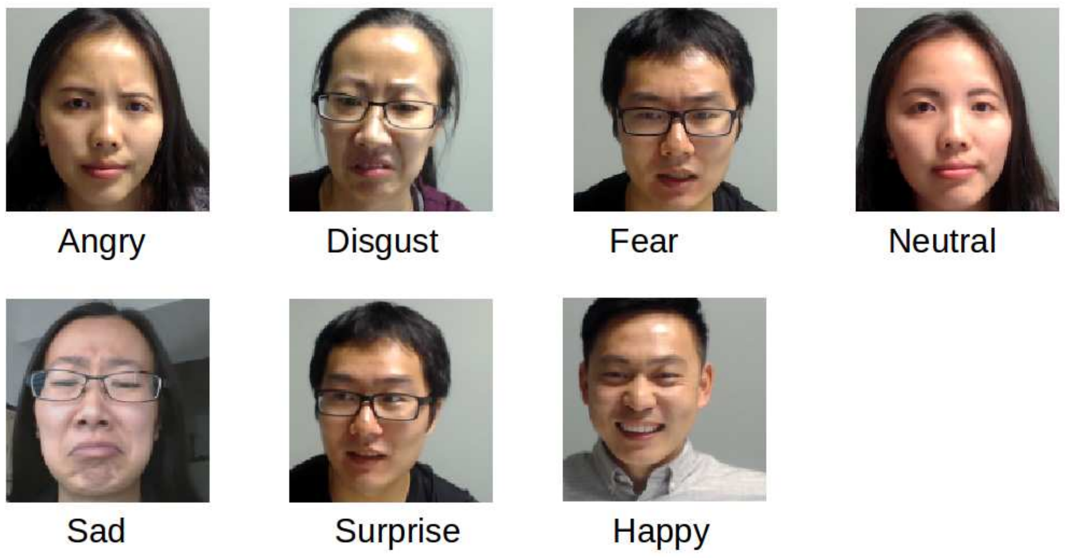

**Figure 8.** Examples of the Dataset I.

### 4.1.2 Samples Collected From User

Total 280 different image samples collected from user and each category has 40 samples. All image samples are resized to 144x144 pixels and named Dataset A. Images in Dataset A converted to gray-scale images, named Dataset B. Images in Dataset A resized to 96x96 pixels, named Dataset C, and resized to 32x32 pixels, named Dataset D. Table V shows the samples detail for Dataset A to D. Figure 9 shows some example images from dataset A. A collection of 70 images samples made Dataset T which used for the test purpose only. Dataset T contains 10 images from each category and it collected few weeks after Dataset A to D has collected. The original size of Dataset T is 144x144 pixels and it needs to be resized or convert to gray scale before uses as the test dataset.

| | Dataset A | Dataset B | Dataset C | Dataset D |
|---|---|---|---|---|
| Images Size | 144x144 | 144x144 | 96x96 | 32x32 |
| Image Type | RGB | Gray | RGB | RGB |
| Image Channel | 3 | 3 | 3 | 3 |

Table V. Dataset A to D specification.

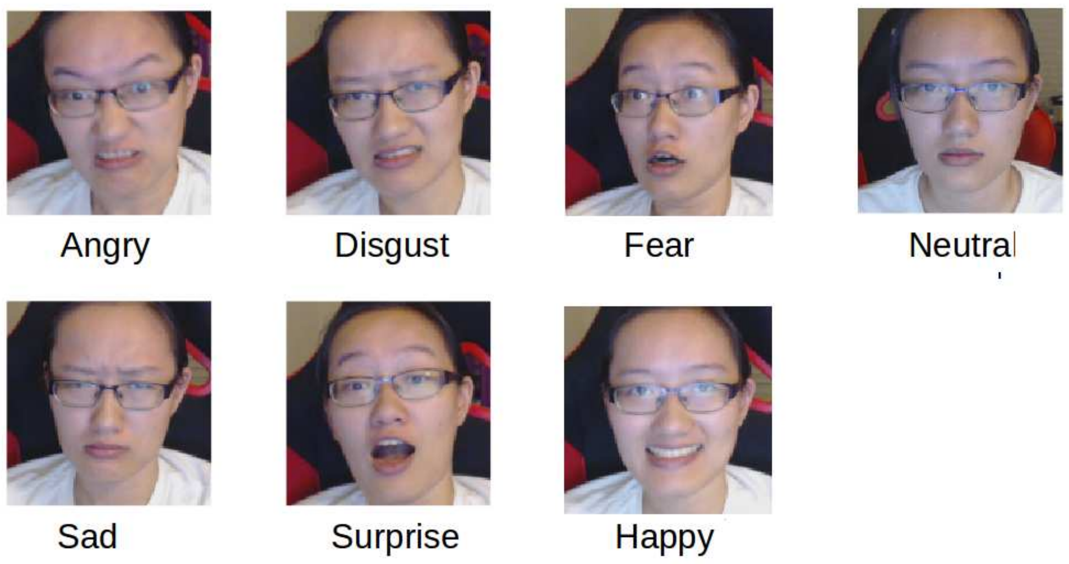

**Figure 9.** Examples of the user collection.

## 4.2 CNN Training Program

### 4.2.1 Different Input Image Size

As mentioned in the challenges section, there are two requirements for the database: quantity and quality. The quantity issues are directly related to the complexity of the problem. Normally, more complex problems need a larger database. There are two ways to solve the quantity issues. One is decreasing the complexity of the problem, and the other is collecting a larger dataset. Ng, et al. [28] discovered that 256x256 pixels images provided better speculation than 48x48 pixels images as training input data. Dataset T used as test dataset. The goal of this experiment is to find out the most effective image size for training, which means without changing the accuracy, what is the smallest size we can use for training. Dataset I to V used to train on the CNN model and compared their results.

The learning rate was set to 1e−4, batch size was set to 20, number of epochs was set to 60 and drop out was set to 50%. The results showed the smaller the input images were, the shorter the training time. Figure 10, Figure 12, Figure 14, Figure 16 and Figure 18 shows the result of the overall training accuracy of the five datasets. All of the training accuracy were about 20% at the beginning, and smoothly approached to one at the end. While the training reached certain step, accuracy became stabilized with some minor fluctuation. The most likely cause of those minor fluctuation is the neural network discovered new future during the learning. Figure 11, Figure 13, Figure 15, Figure 17 and Figure 19 shows the loss during the training phase and all losses are smoothly approached to 0 at the end.

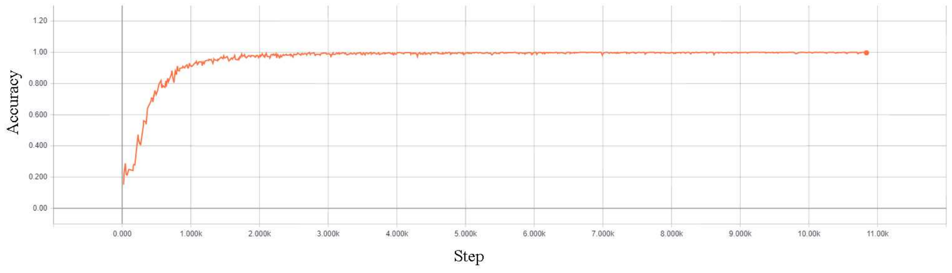

**Figure 10.** Training accuracy with Dataset I.

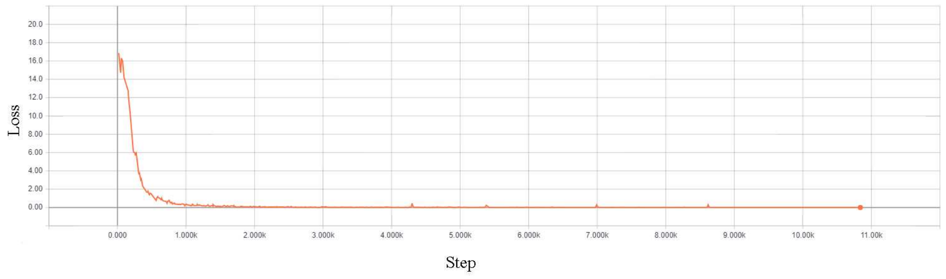

**Figure 11.** Training loss with Dataset I.

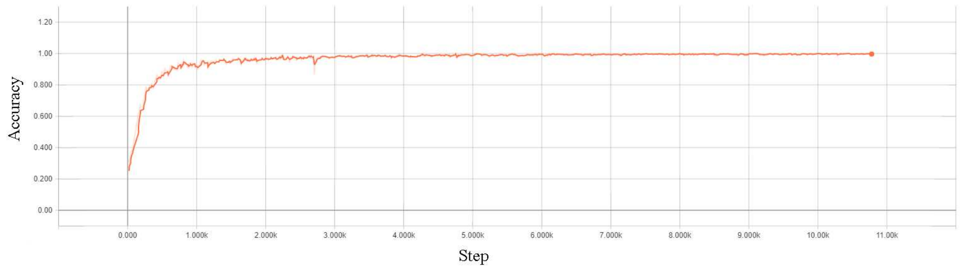

**Figure 12.** Training accuracy with Dataset II.

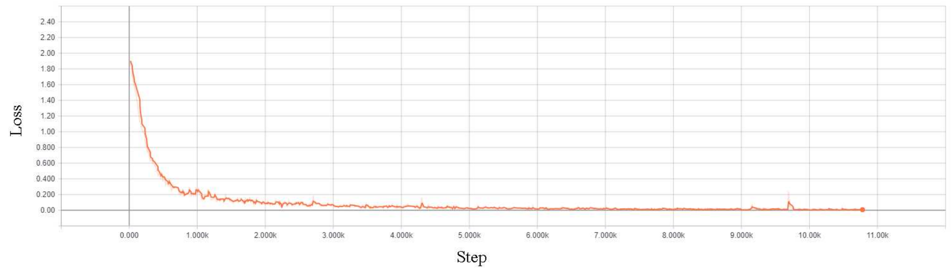

**Figure 13.** Training loss with Dataset II.

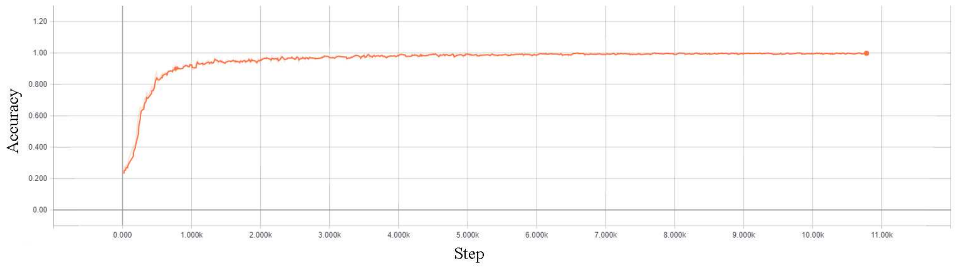

**Figure 14.** Training accuracy with Dataset III.

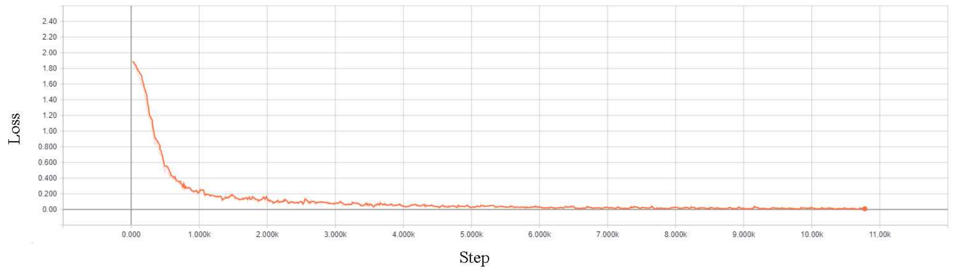

**Figure 15.** Training loss with Dataset III.

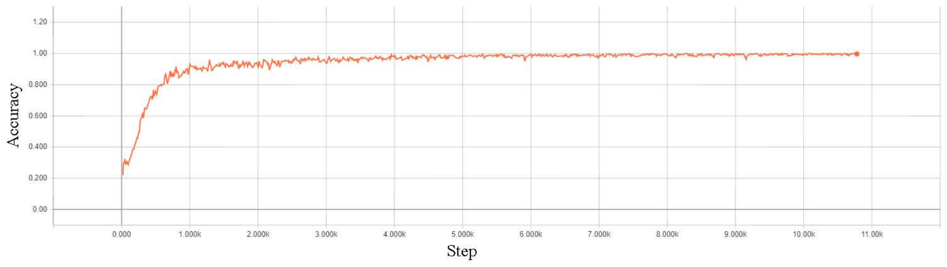

**Figure 16.** Training accuracy with Dataset IV.

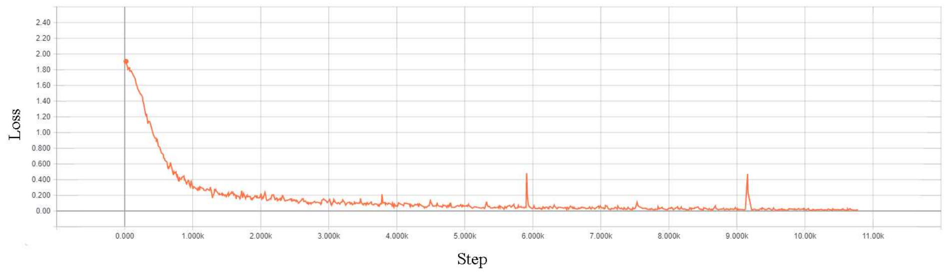

**Figure 17.** Training loss with Dataset IV.

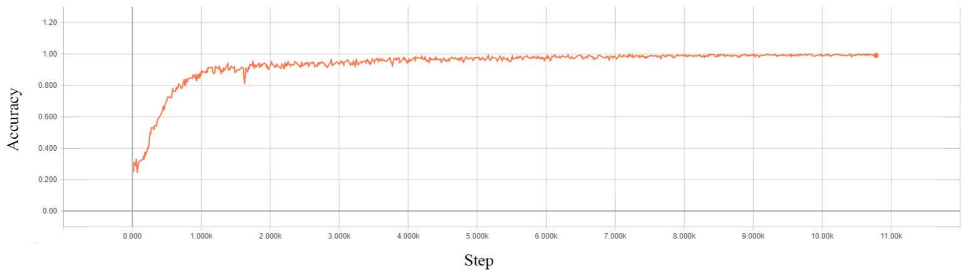

**Figure 18.** Training accuracy with Dataset V.

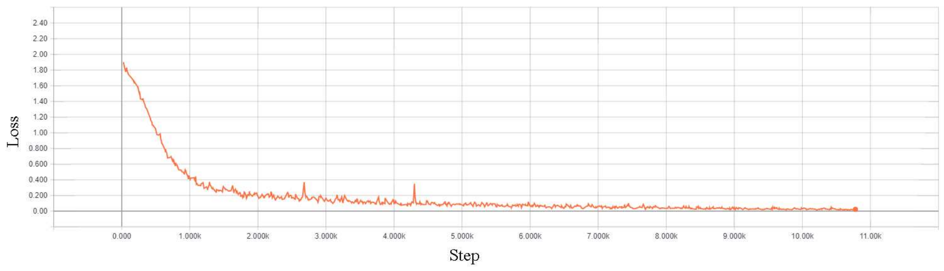

**Figure 19.** Training loss with Dataset V.

Table VI shows the result of the accuracy with training dataset and validation dataset. The training accuracy is very close between five datasets It is hard to summary the advantage between the datasets if we just focus on training accuracy. The accuracy of the validation dataset has a little decreased along with the decreased of the input images size. The test accuracy has more obvious decrease that validation accuracy along with the decreased of the input images size The dataset I with 144x144 pixels images has the best result for the system.

| | Dataset I | Dataset II | Dataset III | Dataset IV | Dataset V |
|---|---|---|---|---|---|
| Training accuracy | 99.66% | 99.94% | 99.85% | 99.67% | 99.10% |
| validation accuracy | 99.95% | 99.94% | 99.81% | 98.18% | 97.88% |
| test accuracy | 32.28% | 28.57% | 25.71% | 21.42% | 18.57% |

Table VI. Training and validation accuracy for Dataset I to V.

### 4.2.2 Differnt Input Image Type

The experiment above summarized the effect of the input image size. The following experiment discusses the effect of image types, color and gray-scale. The goal of this experiment is to find out the most effective image type for training. We converted Dataset I to gray-scale images dataset, named Dataset VI. The specification of Dataset VI discussed in Dataset session. In the training phase, learning rate was set to 1e−4, batch Size was set to 20, number of epochs was set to 60 and drop out was set to 50%. There are 10780 in the training phase and Figure 10 shows the accuracy graph and Figure 11 shows the loss graph during training phase for Dataset I. Figure 20 shows the accuracy graph and Figure 21 shows the loss graph during training phase for dataset 21. Both loss graphs are approaching 0 smoothly, but the accuracy with gray-scale input was very jumpy and it is not approaching anything along with training. The validation accuracy for Dataset VI is 99.90%.

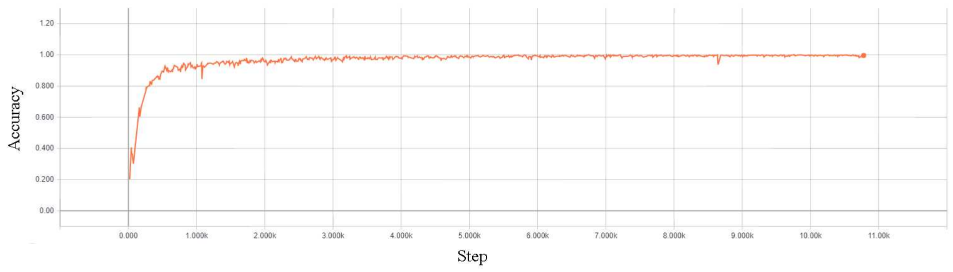

**Figure 20.** Training accuracy with Dataset VI - Gray-scale images.

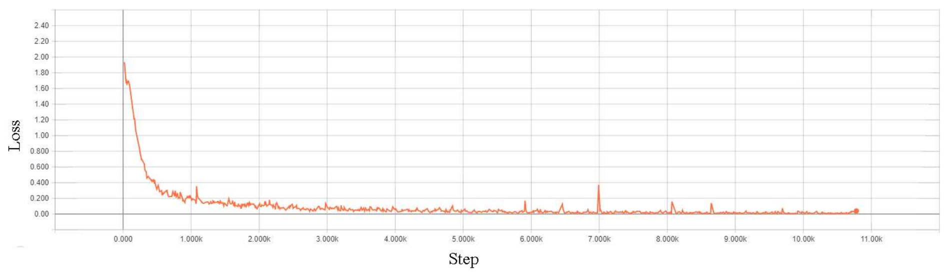

**Figure 21.** Loss with Dataset VI - Gray-scale images.

### 4.2.3 User Model

While we used the result from Dataset I for the facial expression recognizer in PFEMS, the overall accuracy is lower than 30%. In order to increase the accuracy, we used some combination of dataset collected form four volunteers and dataset collected from user for training. Dataset I, III. Vi and V contain images collected from four volunteers and Dataset A, B, C and D contains images collected from user. Table VII shows the combination of the experiment and the training, validation and test accuracy. 70% of the images in the combined dataset was used for training and 30% of it used for validation. Dataset T used as the test dataset.

| | Dataset I and A | Dataset VI and B | Dataset III and C | Dataset V and D |
|---|---|---|---|---|
| Training accuracy | 98.99% | 98.53% | 98.80% | 96.96% |
| Validation accuracy | 92.76% | 96.65% | 91.91% | 89.45% |
| Validation test | 72.56% | 71.99% | 65.15% | 60.31% |

Table VII. Traning and validation accuracy for combined dataset.

For 144x144 images, RBG images and gray scale images did not have significant effect on training and validation accuracy. However, the images size effect the accuracy more than images type.

## 4.3 Facial Detector

The facial detector detected and cropped the front face accurately exceed 95% of the real time testing. Figure 22 shows some of the incorrect detection from the facial detector. While a face turns 30 degrees or more, the facial detector can not detect face in the frame.

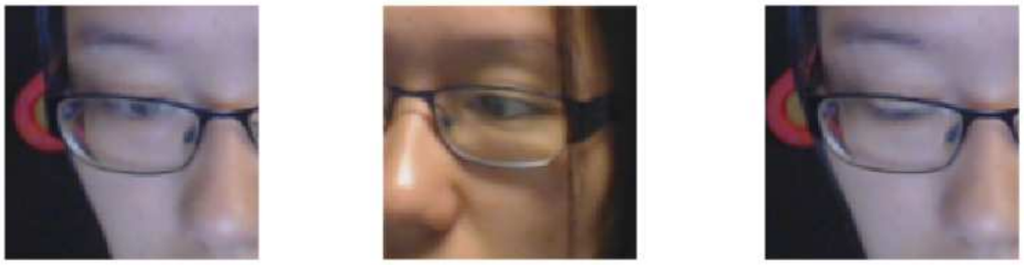

**Figure 22.** Incorrect detection from the facial detector.

## 4.4 Facial Expression Recognizer

The facial expression recognizer used the training result from the combination of Dataset I and collection of the user (Dataset A) as input. The live accuracy, which is only 73.8%, is lower than the accuracy for both combined dataset on Dataset C. Table VIII shows the collected results from facial expression recognizer. There were 151 images collected from the facial detector and 110 images recognized correctly.

| Labels | Angry | Disgust | Fear | Happy | Sad | Surprise | Neutral | Error | Total |
|---|---|---|---|---|---|---|---|---|---|
| Angry | **21** | 0 | 3 | 0 | 0 | 0 | 6 | 1 | 31 |
| Disgust | 0 | **19** | 0 | 0 | 0 | 0 | 0 | 0 | 19 |
| Fear | 0 | 0 | **11** | 0 | 0 | 3 | 0 | 0 | 14 |
| Happy | 0 | 0 | 0 | **15** | 0 | 1 | 2 | 1 | 19 |
| Sad | 0 | 1 | 0 | 2 | **22** | 0 | 0 | 1 | 26 |
| Surprise | 0 | 0 | 3 | 0 | 0 | **11** | 0 | 4 | 16 |
| Neutral | 1 | 0 | 1 | 5 | 1 | 4 | **11** | 1 | 24 |
| Total | 22 | 20 | 18 | 22 | 23 | 19 | 19 | 6 | 151 |

Table VIII. Collected results from real time recognizer.

Facial expression recognizer reaches 100% accuracy on disgust expression. Happy and sad expression had over 80% accuracy with or without error images. Surprise had 78.57% accuracy without error, but only 61.1% accuracy with the error image. Neutral is the most difficult expression for the facial expression recognizer, only 45.8% with the error and 47.8% without error image.

There are some confirmed factors that reduce the accuracy, such as incorrect cropped images and low quality cropped images. The incorrect cropped images definitely decrease the accuracy of the facial expression recognizer. This is inescapable in the current system. The only way to avoid or reduce the effect is improving the facial detector. However, a better the facial detector may have more complex operations which may increase the run time. Low quality cropped images maybe caused by the position of the user and the movement of the user. When user is far from the camera, the actual cropped image has a smaller size than requested. After re-size the actual cropped image, the new image becomes blurry. When user moved in front of the camera, due to the quality of the camera, the camera cannot change focus fast enough and frame is blurred until the camera sharp focuses. A Blurred image is harder to be recognized.

Figure 23 shows some misclassified facial expression images. Figure 23.a is a fear expression, but misclassified as angry. Figure 23.b is a surprise expression, but misclassified as happy. Figure 23.c is a neutral expression, but misclassified as sad. Figure 23.d is a happy expression, but misclassified as neutral. For this particular userthe expressions, Figure 23.a, did look like the other trained angry images. Figure 23.b and Figure 23.d are images in transition between two different expression. Figure 23.b is in a transition between happy and surprise. Figure 23.d is in a transition between neutral and happy. Figure 23.c looks neutral, but also appears a little sad from the image. The facial expression recognizer did not handle images which contain more than one expression well enough.

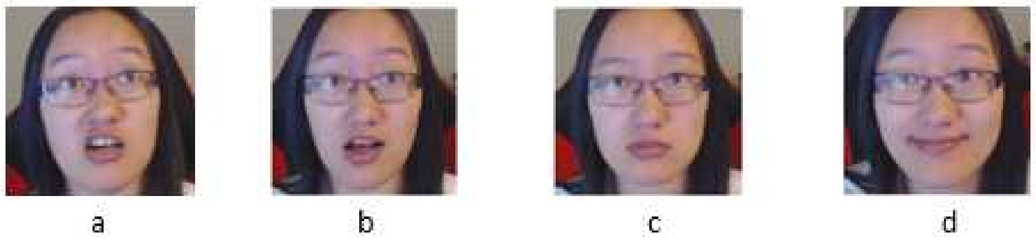

**Figure 23.** Misclassified facial expression images

# CHAPTER 5

# CONCLUSION AND FUTURE WORK

We develop a system that recognizes facial expression can be used to determine the emotion state of a user there by creating a personal facial expression monitoring system (PFEMS). We accomplish this by design a CNN model for recognized facial expression of the six universal emotions plus neutral using TensorFlow. We designed and run experiments to determine CNN depth and image resolution required for effective recognition. PFEMS is able to train images, detect face from real time video and recognize facial expression. The training phase for each experiment takes approximately two to three hours on a Nvidia 1080 GPU and five to six hours on a Nvidia 980Ti GPU with a solid state drive.

During the process of building the PFEMS, we discovered that the size of the training image affected the results from the training program. The higher resolution produces better result. After extracted facial images from the video that the four volunteers provided, the smallest facial images is 144x144 pixels. In order to avoid up sampling, all facial images are down sampling to 144x144 pixels which is the highest resolution in the experimented dataset. We also did an experiment with gray-scale images input. Gray-scale and color image with the same sizes have similar results for training and validation, but color images have a little bit better result on the test dataset. We summarized that 144x144 pixels, which is the highest resolution used in the experiments, color images made the best training input dataset for our system. The facial detector has an acceptable cropping success rate with front face which is enough for the system to use, but the error rate still has room for improvement. The facial expression recognizer had 73.8% accuracy on fixed user, but the facial expression recognizer had a hard time handling an expression while it is in transition. The accuracy for neutral is only 45.8%. This is the weak link of the facial expression recognizer.

For the future improvement, PFEMS needs a larger dataset for training and more sample for testing. The CNN model can be improved by making it deeper and wider, but the processing time will be longer. And a more complex CNN model is harder to adjust learning rate, drop out and etc. The facial detector has a 5% error rate which results in the facial expression recognizer has an unmendable 5% error rate. The facial detector can be improved by adding eye and mouth detector. Detecting eye and mouth from the cropped images cab double check the result from he cascade classifier. If two eyes and one mouth detected from the cropped image, the cropped image is usable. This can reduce the facial detector error rate and increase the maximum possible accuracy for the facial expression recognizer. The facial expression recognizer needs to process faster to satisfy the need of human computer interaction project. The PFEMS needs to be more compatible for different platform and different human computer interaction system.

# REFERENCES

[1] A. Dhall, O. V. R Murthy, R. G. . J., and T.Gedeon. Video and image based emotion recognition challenges in the wild: Emotiw 2015. In *the 2015 ACM on international Conference on Multimodal interaction* (2015), ACM, pp. 423–426.

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

[3] Abaya, W. F., Basa, J., Sy, M., Abad, A. C., and Dadios, E. P. Low cost smart security camera with night vision capability using raspberry pi and opencv. In *Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management (HNICEM), 2014 International Conference on* (2014), IEEE, pp. 1–6.

[4] Black, M. J., and Yacoob, Y. Tracking and recognizing rigid and non-rigid facial motions using local parametric models of image motion. In *Computer Vision, 1995. Proceedings., Fifth International Conference on* (1995), IEEE, pp. 374–381.

[5] Bradski, G., et al. The opencv library. *Doctor Dobbs Journal 25*, 11 (2000), 120–126.

[6] Bradski, G., and Kaehler, A. *Learning OpenCV: Computer vision with the OpenCV library.* " O'Reilly Media, Inc.", 2008.

[7] Burkert, P., Trier, F., Afzal, M. Z., Dengel, A., and Liwicki, M. Dexpression: Deep convolutional neural network for expression recognition. *arXiv preprint arXiv:1509.05371* (2015).

[8] Chakraborty, A., Konar, A., Chakraborty, U. K., and Chatterjee, A. Emotion recognition from facial expressions and its control using fuzzy logic. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans 39*, 4 (2009), 726–743.

[9] Chan, K.-P., Zhan, X., and Wang, J. Facial expression recognition by correlated topic models and bayes modeling. In *Image and Vision Computing New Zealand (IVCNZ), 2015 International Conference on* (2016), IEEE, pp. 1–5.

[10] Cheng, F., Yu, J., and Xiong, H. Facial expression recognition in jaffe dataset based on gaussian process classification. *IEEE Transactions on Neural Networks 21*, 10 (2010), 1685–1690.

[11] Coelho, G., Kougianos, E., Mohanty, S. P., Sundaravadivel, P., and Albalawi, U. An iot-enabled modular quadrotor architecture for real-time aerial object tracking. In *Nanoelectronic and Information Systems (iNIS), 2015 IEEE International Symposium on* (2015), IEEE, pp. 197–202.

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

[13] Gavrilescu, M. Proposed architecture of a fully integrated modular neural network-based automatic facial emotion recognition system based on facial

action coding system. In *Communications (COMM), 2014 10th International Conference on* (2014), IEEE.

[14] Guo, Y., Tao, D., Yu, J., Xiong, H., Li, Y., and Tao, D. Deep neural networks with relativity learning for facial expression recognition. In *Multimedia & Expo Workshops (ICMEW), 2016 IEEE International Conference on* (2016), IEEE, pp. 1–6.

[15] Hubel, D. H., and Wiesel, T. N. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology 160*, 1 (1962), 106–154.

[16] I. J. Goodfellow, D. Erhan, P. L. C. A. C. M. B. H. W. C. Y. T. D. T. D. H. Y. Z. C. R. F. F. R. L. X. W. D. . S.-T. M. M. . P. R. M. P. C. G. . B. . X. L. B. X. Z. C., and Bengio, Y. Challenges in representation learning: A report on three machine learning contests. In *Neural Networks* (2013), Neural Networks, pp. 59–63.

[17] Kamachi, M., Lyons, M., and Gyoba, J. The japanese female facial expression (jaffe) database. *URL http://www. kasrl. org/jaffe. html 21* (1998).

[18] Kanade, T., C. J. F. . T. Y. Comprehensive database for facial expression analysis. In *Paper presented at the Fourth IEEE International Conference on Automatic Face and Gesture Recognition* (2000), IEEE.

[19] L. Yin, X.Wei, Y. S. J. a. J. R. A 3d facial expression database for facial behavior research. *IEEE Int. Conf. Automat. Face Gesture Recog. Workshops* (2006), 211–216.

[20] Lien, J. J., Kanade, T., Cohn, J. F., and Li, C.-C. Automated facial expression recognition based on facs action units. In *Automatic Face and Gesture Recognition, 1998. Proceedings. Third IEEE International Conference on* (1998), IEEE, pp. 390–395.

[21] Lienhart, R., and Maydt, J. An extended set of haar-like features for rapid object detection. In *Image Processing. 2002. Proceedings. 2002 International Conference on* (2002), vol. 1, IEEE, pp. I–I.

[22] Liu, T., Fang, S., Zhao, Y., Wang, P., and Zhang, J. Implementation of training convolutional neural networks. *arXiv preprint arXiv:1506.01195* (2015).

[23] Lundqvist, D., Flykt, A., and Öhman, A. The karolinska directed emotional faces-kdef. cd-rom from department of clinical neuroscience, psychology section, karolinska institutet, stockholm, sweden. Tech. rep., ISBN 91-630-7164-9, 1998.

[24] Lyons, M. J., Budynek, J., and Akamatsu, S. Automatic classification of single facial images. *IEEE Transactions on Pattern Analysis and Machine Intelligence 21*, 12 (1999), 1357–1362.

[25] M. Pantic, M. F. Valstar, R. R. L. M. Web-based database for facial expression analysis. In *Proceedings of IEEE Int'l Conf. Multimedia and Expo (ICME'05).*

[26] Manoharan, R., and Chandrakala, S. Android opencv based effective driver fatigue and distraction monitoring system. In *Computing and Communications Technologies (ICCCT), 2015 International Conference on* (2015), IEEE,

pp. 262–266.

[27] Mohri, M., Rostamizadeh, A., and Talwalkar, A. *Foundations of machine learning.* MIT press, 2012.

[28] R. Gross, I. Matthews, J. C. T. K., and Baker, S. Multi-pie. *Image Vis. Comput. 28* (2010), 807–813.

[29] Rivera, A. R., Castillo, J. R., and Chae, O. O. Local directional number pattern for face analysis: Face and expression recognition. *IEEE transactions on image processing 22*, 5 (2013), 1740–1752.

[30] Schmidhuber, J. Deep learning in neural networks: An overview. *Neural Networks 61* (2015), 85–117.

[31] Shin, M., Kim, M., and Kwon, D.-S. Baseline cnn structure analysis for facial expression recognition. In *Robot and Human Interactive Communication (RO-MAN), 2016 25th IEEE International Symposium on* (2016), IEEE, pp. 724–729.

[32] Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. Mastering the game of go with deep neural networks and tree search. *Nature 529*, 7587 (2016), 484–489.

[33] Terence, S., Simon, B., and Maan, B. The cmu pose, illumination, and expression (pie) database. In *Proc IEEE Int Conf Autom Face Gesture Recognit* (2002), pp. 46–51.

[34] Viola, P., and Jones, M. Robust real-time object detection. *International Journal of Computer Vision 4* (2001).

[35] Zhang, T., Zheng, W., Cui, Z., Zong, Y., Yan, J., and Yan, K. A deep neural network driven feature learning method for multi-view facial expression recognition. *IEEE Trans. Multimed 99* (2016), 1.