Emotion Recognition using Convolutional Neural Networks

Chieh-En James Li
li3261@purdue.edu

Lanqing Zhao
zhao584@purdue.edu

The Application of CNN on Emotion Recognition

Purdue – Vertically integrated Projected

Author: Lanqing Zhao, Chieh-En Li Team Advisor: Shaoyuan Xu, Prof. Jan Allebach

School of Electrical and Computer Engineering, Purdue University

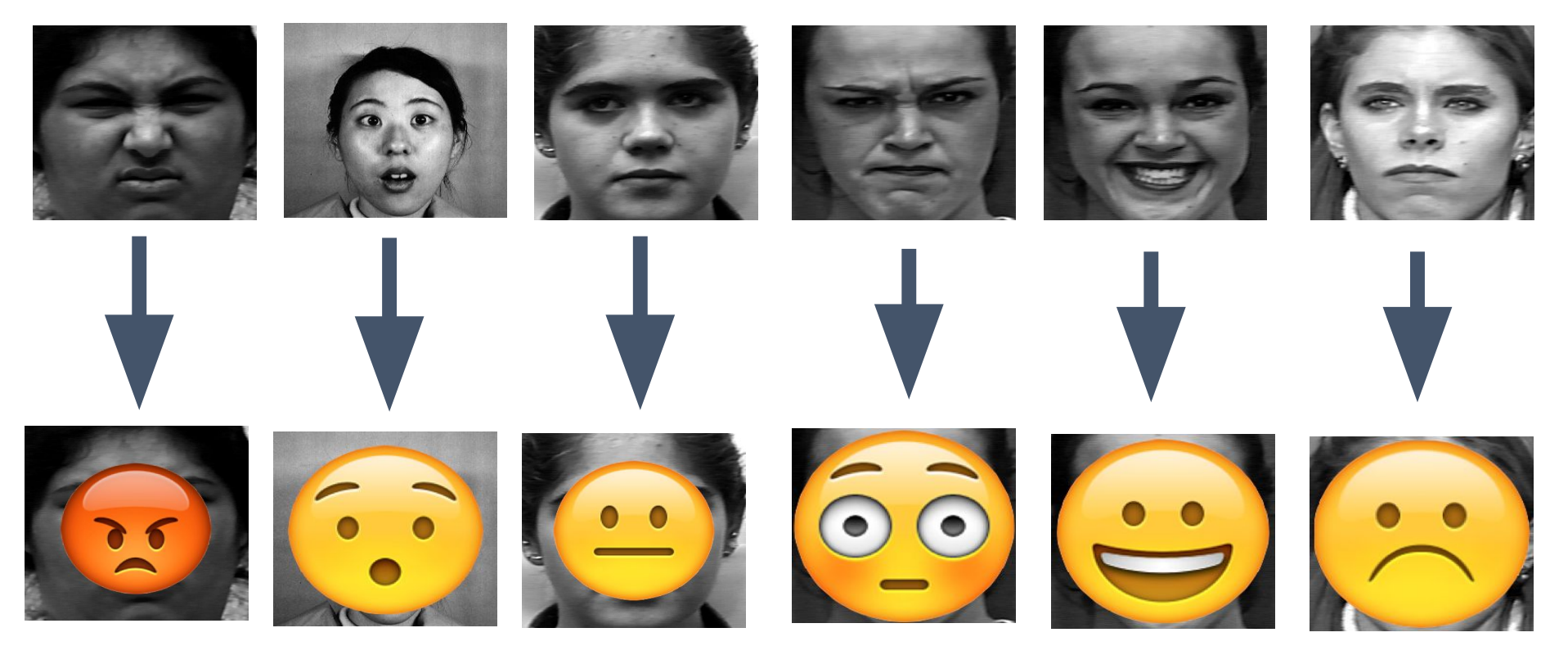
Abstract

Emotion recognition becomes a popular topic of deep learning and provides more application fields in our daily life. The primary idea of our project is to process the input images of human facial emotion to train the pre-trained models on datasets .The existing conventional method requires a handcrafted feature extractor of facial Action Units(AUs) to extract feature from designated Facial Landmark regions, and these extracted AUs codes are processed through traditional machine learning algorithm such as Nearest Neighbors and SVM, which is a typical type of linear classifier. The problem with conventional method is that the lighting variations and different position of object may corrupt the feature vector so that the accuracy is greatly reduced.Furthermore, it is typically difficult to conduct feature-engineering to fit the demand of facial recognition. In this project, we approach the problem by taking deep-learning method of Convolutional Neural Networks(CNNs), which integrates the step of handcrafted feature extraction with training of classifier . This system is able to achieve the relatively most optimal solution through the process of backpropagation in which the algorithm learns the weights through modified stochastic gradient descent that can find the directions that best minimize the loss from the ground truth. The numerical result of the algorithm will show a probabilistic result of each labeled class. In order to reduce computational expense, the technique of fine-tuning is applied so that a pre-trained model can adapt the variance of our local dataset with benefit of reducing computational expense. As results, such method best resolves the issues of lighting variations and different orientation of object in the image and thus achieves a higher accuracy.

Introduction

The facial expression of human emotion is one of the major topics in facial recognition, and it can generate both technical and everyday application beyond laboratory experiment. This projection constructs a system of deep learning model to classify a given image of human facial emotion into one of the seven basic human emotions . The approach we take to build the model is through transfer learning of an existing pre-trained model, and the testing result will be evaluated based on accuracy of the model.

The tasks in this project include preprocessing the image data, augmentation to enlarge the existing small dataset, test before training the model, training process, and predication with evaluation. The visual demonstration of the result will be similar to the figure below. The baseline of the test will be around 14.7% (1 in 7), and our objective is to achieve a result better than baseline accuracy.



A demonstration: The emojis are correctly imposed on their corresponding faces. There are seven different emotions:happy,angry,sad,fear,surprise,neutral, and disgust.,The input will be raw image of the expression, and output will be shown as above.The demonstration does not include disgust as its emoji is similar to angry

Method and Procedures

The Method:

In this project, we use a deep learning technique called Convolutional Neural Networks(CNNs) to establish a classification model that combines feature extraction with classification. Since the training typically takes more a week to complete from scratch, we decide to fine-tune the existing model VGG trained by University of Oxford. In this way, we do not have to take large cost of computation in both time and opportunity, and meanwhile the data can assist the model to fit the local variance and condition.

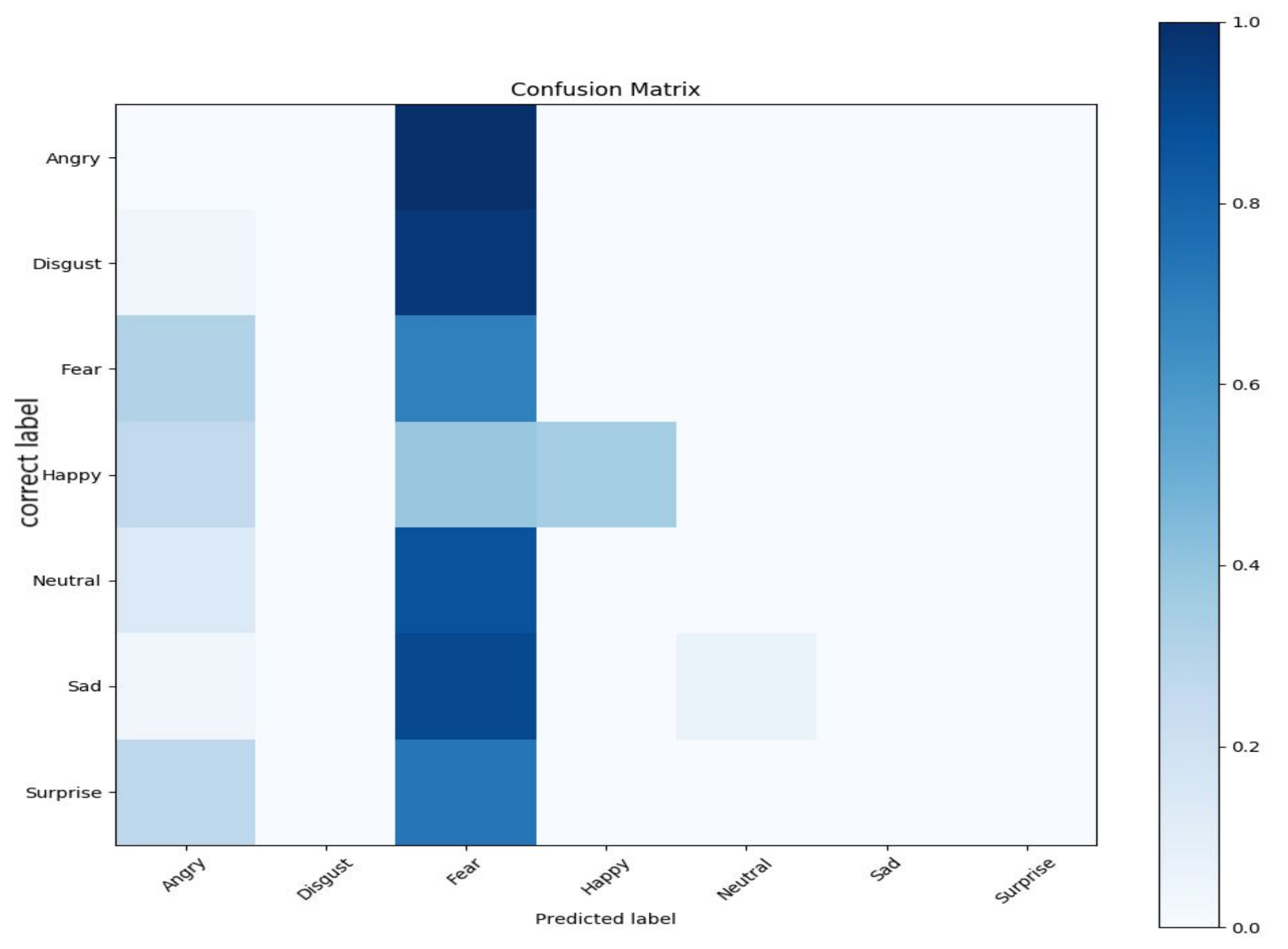
We will preserve most of its original feature-extraction layers (first four blocks of convolutional layers and pooling layers), and the training will retrain the last block of convolution layers as well as the fully connected classifier layers. This project will reply on Keras and Google Colab to complete the training task, and the test before training will be conducted through the framework caffe.

The dataset we use includes: JAFFE, CK+, and augmented forms of both generated by our team.

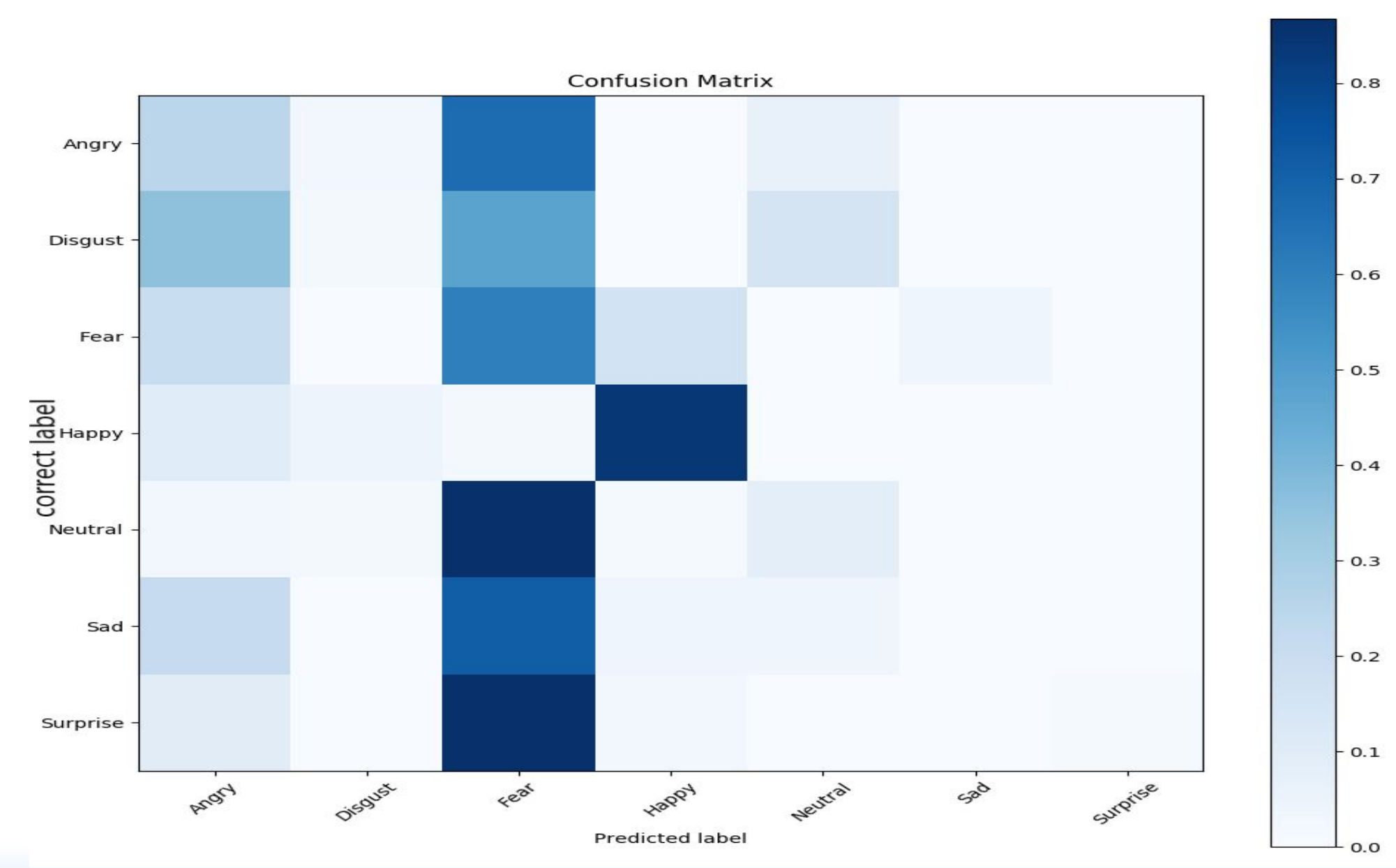
The Procedure:

1. **Obtain the raw dataset and transfer them into JAFFE format of cropped face and correct label**
2. **Generate lmdb database for the test before training**
3. **Obtain the confused matrix (shown as below)**
4. **Obtain model and train on Google Colab**
5. **Evaluate the training results**

Base on CK+ dataset. the result before training



Base on JAFFE dataset. the result before training



Experiment

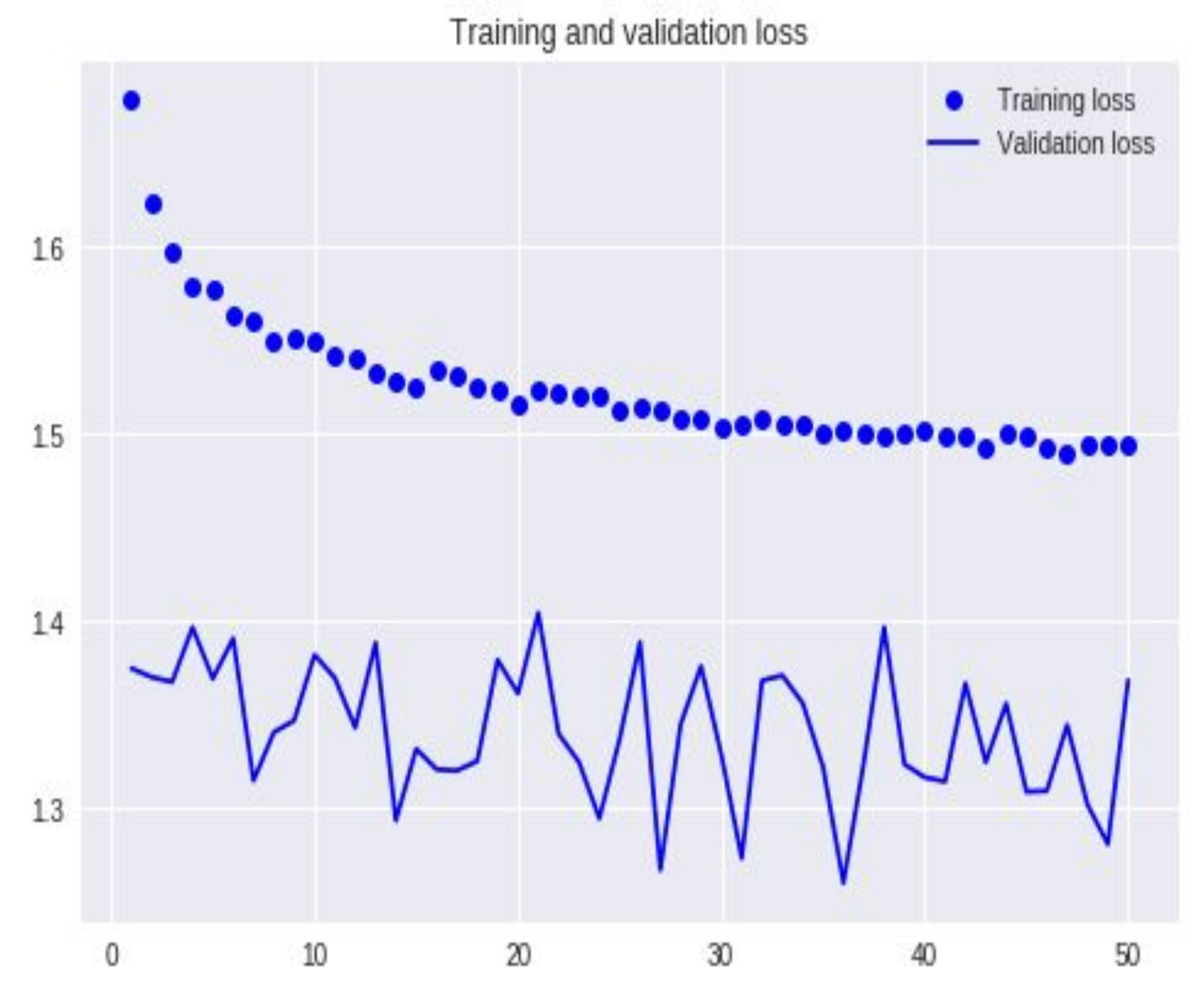
Test before training :

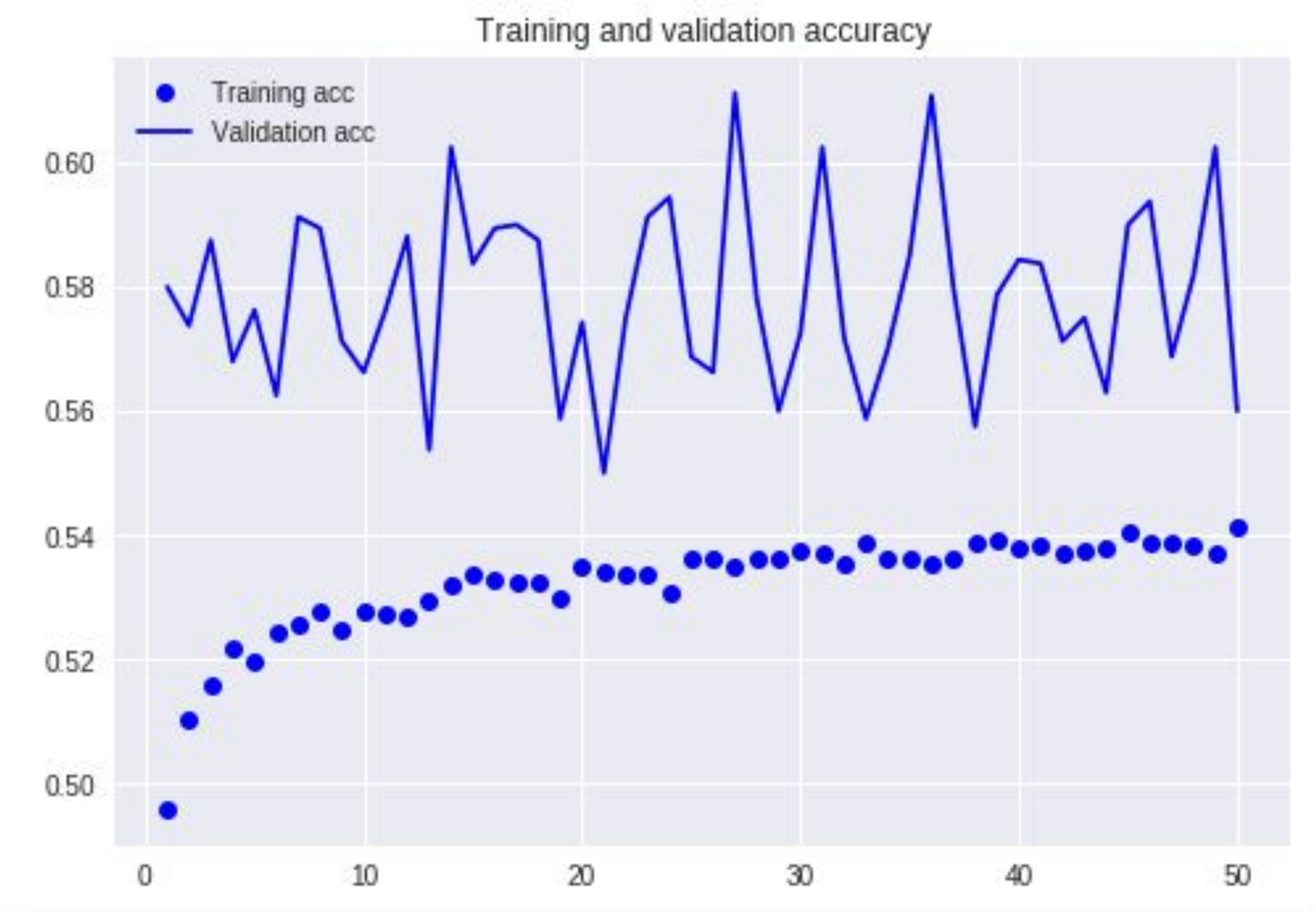
The data is preprocessed to change the format of filename to include labels, and CK+ is cropped to separate from background.
The test was conducted on pre-trained VGG_S model fine-tuned by G Levi et.al. The confused matrix is generated as well as demonstrations.

Data Augmentation and Training:

The dataset is divided into training,validation, and test. The team also enlarges the dataset by 30 times through addition of random noise, rotation of angles, and horizontal flip (like mirror). The dataset is also divided based on labels into 7 classes.

The training is conducted on VGG_16 model. The GPU is Tesla K80 GPU with accessible memory from 8.2 GB to 10.7 GB. The RAW is about 2-3GB
The batch size is 64 for training and 16 for validation. The total epochs of this training is 50, and the steps of epoch is 448.
The optimizator is Stochastic Gradient Descent (SGD), and Nesterov Momentum was applied with base learning rate of 1e-4.
The training takes about 6 and half hours to complete.





Result and Demonstration

The forward pass test before the training archives 15.4% accuracy, and most of the emotion is classified as fear (shown in the figure). The effect of demonstration is like what figures show.

After 6 and half hours training by GPU, we get an training accuracy of 55%, and validation accuracy from 55-58% . The result indicates that the trained model has issue of overfitting. The potential reasons are lack of regularization and data normalization as well as biased dataset with 15000 of 28000 images are neural.(figure shown left). The plot has a significant vibration and early convergence

Expectation and Conclusion

In conclusion, our model will not be considered to achieve the objective as the dataset is biased with issue of overfitting.

In the future, we will focus on refine the dataset and model to reduce the effect of overfitting and early convergence of loss function. We will add more regularization and normalize the dataset.

Reference and Acknowledgement

This project originates from the ideas of HappyNet, a stanford CS231n project, and its source code also comes from HappyNet. We modify them based on our local machine and demand to perform the test before training and effect of demonstration

1. LeCun, Y., Bengio, Y., & Hinton, G. (2015, May 27). Deep learning. Retrieved from https://www.nature.com/articles/nature14539
2. Levi, G., & Hassner, T. (2015). Emotion Recognition in the Wild via Convolutional Neural Networks and Mapped Binary Patterns. Proceedings of the 2015 ACM on International Conference on Multimodal Interaction - ICMI 15. doi:10.1145/2818346.2830587
3. - Kanade, T., Cohn, J. F., & Tian, Y. (2000). Comprehensive database for facial expression analysis. Proceedings of the Fourth IEEE International Conference on Automatic Face and Gesture Recognition (FG'00), Grenoble, France, 46-53.
4. - Lucey, P., Cohn, J. F., Kanade, T., Saragih, J., Ambadar, Z., & Matthews, I. (2010). The Extended Cohn-Kanade Dataset (CK+): A complete expression dataset for action unit and emotion-specified expression. Proceedings of the Third International Workshop on CVPR for Human Communicative Behavior Analysis (CVPR4HB 2010), San Francisco, USA, 94-101.
5. Simonyan, Karen, Zisserman, & Andrew. (2015, April 10). Very Deep Convolutional Networks for Large-Scale Image Recognition. Retrieved from https://arxiv.org/abs/1409.1556
6. Coding Facial Expressions with Gabor Wavelets, Michael J. Lyons, Shigeru Akamatsu, Miyuki Kamachi & Jiro Gyoba Proceedings, Third IEEE International Conference on Automatic Face and Gesture Recognition,April 14-16 1998, Nara Japan, IEEE Computer Society, pp. 200-205.
7. Automatic Classification of Single Facial Images,Michael J. Lyons, Julien Budynek, & Shigeru Akamatsu ,IEEE Transactions on Pattern Analysis and Machine Intelligence 21 (12): 1357-1362 (1999).
8. Duncan, D., Shine, G., & English, C. (2016). Facial Emotion Recognition in Real Time. Retrieved April 2, 2019, from http://cs231n.stanford.edu/reports/2016/pdfs/022_Report.pdf
9. Chollet, F. (2018). Deep learning with Python. Shelter Island, NY: Manning Publications.

Final Report of Emotion Recognition Team

VIP Imaging and Printing

Lanqing Zhao
E-mail: zhao584@purdue.edu

Chieh-En Li
E-mail: li3261@purdue.edu

Figure 1: photo of Lanqing Zhao

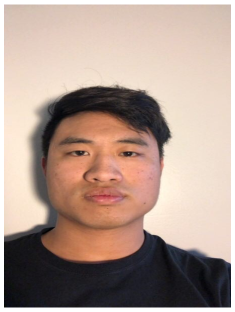


Figure 2: photo of Chieh-En Li

School of Electrical and Computer Engineering
Purdue University
Spring 2019

Contents

1 Abstract

Emotion Recognition is a task to process a human facial expression and classify it into certain emotion categories. Such task typically requires the feature extractor to detect the feature, and the trained classifier produces the label based on the feature. The problem is that the extraction of feature may be distorted by variance of location of object and lighting condition in the image. In this project, we address the solution of the problem by using a deep learning algorithm called Conventional Neural Network (CNN) to address the issues above. By using this algorithm, the feature of image can be extracted without user-defined feature-engineering, and classifier model is integrated with feature extractor to produce the result when input is given. In this way, such method produces a feature-location-invariant image classifier that achieves higher accuracy than traditional linear classifier when the variance such as lighting noise and background environment appears in the input image [1] . The evaluation of the model shows that the accuracy of our lab condition testing data set is 94.63%, and for wild emotion detection it achieves only around 37% accuracy. In addition, we also hope to demonstrate our project shown as figure 3 [2].

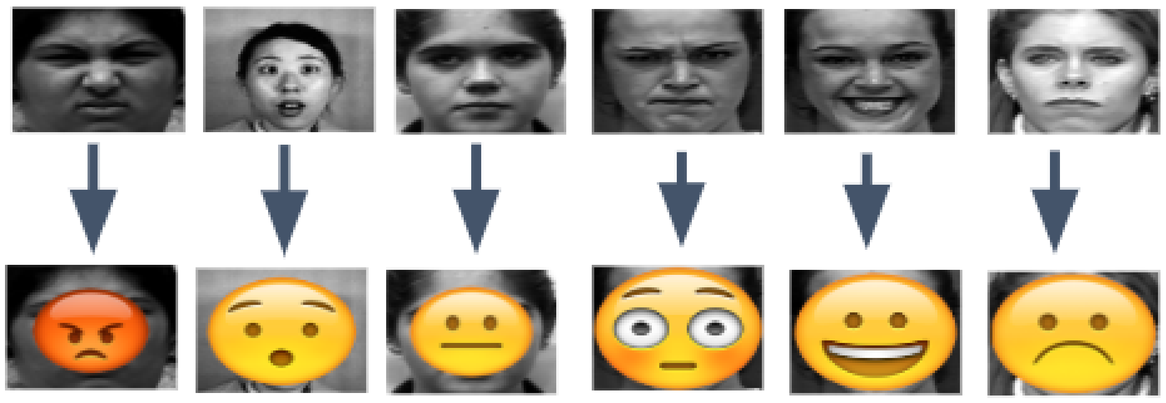


Figure 3: An Example of demonstration that the emojis are imposed on the input images [3] [4] [5] [6] [7]

2 Goals of the Project

The goals of this project are to establish a model that can classify 7 basic emotions: happy, sad, surprise, angry, disgust,neutral, and fear and to achieve the accuracy better than the baseline 14.29%. In addition to this, our project also aims to analyze the results of our model in terms of accuracy for each class. In the future, the model is expected to perform wild emotion recognition that has more complex variance of condition than lab condition images.

3 The General Problem

In real life, people express their emotion on their face to show their psychological activities and attitudes in the interaction with other people. The primary focus of this project is to determine which emotion an input image that contains one facial emotion belongs to. Because human face is complex to interpret, emotion recognition can be specifically divided into classification of basic emotion and classification of compound emotion [8]. For the goals of our project, the essential problem is to focus on the classification of 7 basic emotions(shown at below):

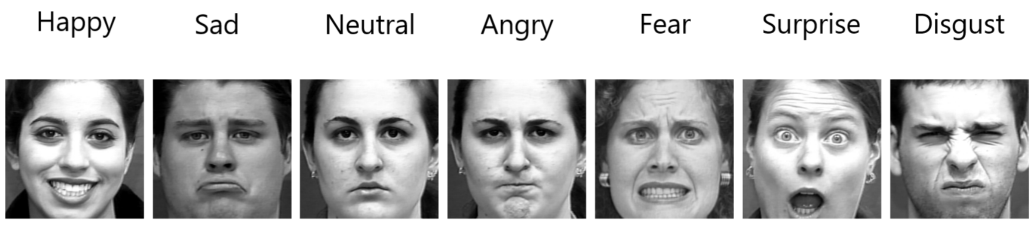


Figure 4: The examples of 7 basic emotions:Happy, Sad, Neutral, Angry, Fear, Surprise, and Disgust [3] [7]

In conclusion, we want to construct a system by which an input image that contains one expression belonging to one of the 7 basic emotions can generate an output that correctly labels the input image.

4 Conventional Methods and Their Issues

4.1 A Short Introduction to Conventional Method

Like every other classification problems, the emotion recognition problem requires an algorithm to complete feature extraction and categorical classification .In order to classify an emotion, we need to extract certain feature from data and build an model that can classify the input based on the feature. The procedure can be outlined as following [8]:

1 **Data Pre-processing:**
The data pre-processing is to standardize the data. The typical way is to set the mean of the data to 0 and to also divide the data by the standard deviation [9]

2 **Feature Extraction:**
The typical conventional method is to detect the face and extract the Action Units(AU)(shown in figure 5) from the face, and certain emotion contain the combination of AUs code as feature.

3 **Model Construction:**
The conventional classifier can be either supervised or unsupervised algorithm. A typical example of supervised algorithm is Support Vector Machine, and the examples of unsupervised

algorithm include Principle Component Analysis(PCA) and Linear Discriminant Analysis (LDA) [1] [8]

4 **Label or Result Generation**

The typical way to generate label or result is to find which decision boundary has the minimum Euclidean distance from the data.

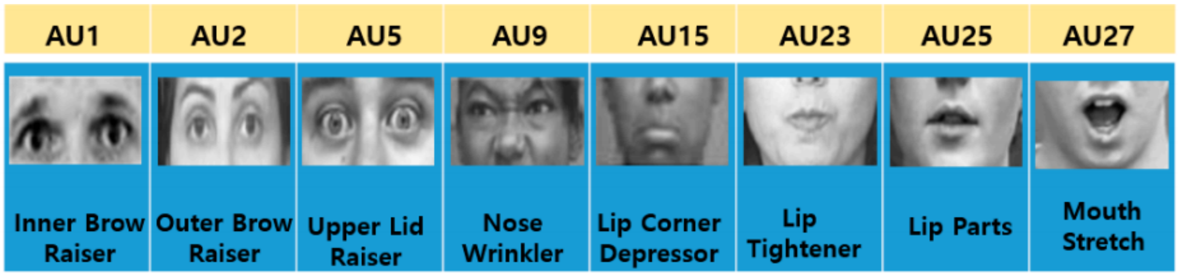


Figure 5: Examples of some Action Units [8]

4.2 The Issue with Conventional Method

The issues of Conventional method are [1] [2]:

A. **Variance of Lights**

Since each image is taken in the completely different background and lighting conditions, the intra-class noise of lights will distort the model to classify the emotion. As results, the same type of emotions may be classified differently because of the effect of lighting noise.

B. **Variance of Location**

Since the feature is typically extracted by filters such application of Local Binary Pattern in [10], the location of the feature, therefore, may affect the functionality of the feature extraction. As results, the AU may be extracted incorrectly if the face has is rotated or is in different part of the image.

These two issues are major problems faced by conventional algorithms.

5 Solution to the Issues of Conventional Method

5.1 Summary of Solution

In order to solve the issues of the conventional methods, the solution is to apply the algorithm of Convolutional Neural Network(CNN) to perform classification of emotion. In contrast to conventional method,the key differences of algorithm are [1] :

1. **Automatic Generation of Feature Extractor:**
   The feature of images can be captured automatically without users' built feature extractor because the feature extractors are generated in the process of training based on the given ground truth.

2. **Differences of Mathematical Model:**
   The conventional method is typically performing classification through linear transformation, so they are typically referred as Linear Classifier. In comparison, CNN as well as other Deep Learning algorithm typically combines the linear transformation with nonlinear function such as sigmoid (Logistic Function) and Rectified Linear Unit(ReLU) to distinguish differences in process of classification.

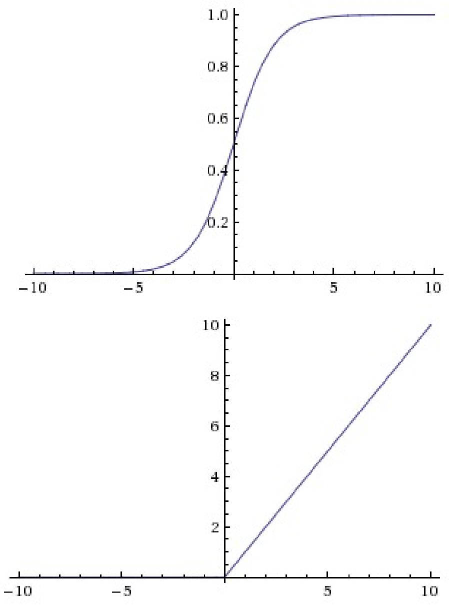


Figure 6: The sigmoid function(upper) and ReLU function (lower) [9]

3. **The Deeper Structure:** [9]
   The conventional method typically performs only one layers of operation: for example SVM only has one set of weights.(shown in equation 1) However, CNN as well as other deep learning algorithm does multiple layers of operation in the process of classification.

$$S = W * x_i + b \tag{1}$$

where S is the classification score,W is weights matrix, and b is bias

In the following section, we are going to discuss CNN in details and point out its advantages and disadvantages.

5.2 Introduction of CNN

5.2.1 The Structure of CNN

In general, CNN contains 2 blocks of layers: convolutional layers and fully-connected Layers(also known as Dense Layers). The convolutional layers are used to perform the task of feature extraction as it detects and produces signals of feature by doing dot product between convolutional kernels and the feature map or images [1]. On the other hand , the fully-connected layers consists of units as neuron cells that mathematically represents the linear operation and the operation of activation function(shown in figure 7). The last layer, a softmax classifier (equation 2) will generate the output

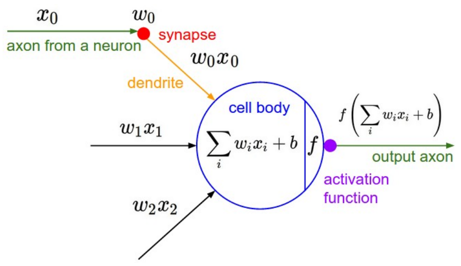


Figure 7: The Illustration of a Neuron Unit in Dense Layers [9]

in the format of the probability of each class in multinomial distribution, and the correct label will be the one with the maximum probability.(equation 2 and 3 show the softmax classifier model)

$$P(y = j|\theta^i) = \frac{e^{\theta^i}}{\sum_{j=0}^{k} e^{\theta_k^i}} \tag{2}$$

where :

$$\theta = w_0x_0 + w_1x_1 + ... + w_kx_k = \sum_{i=0}^{k} w_ix_i = w^Tx \tag{3}$$

For the Convolution layers, the structure includes convolution filters, activation function, and maxpooling filters. The convolution filters are kernels with certain size that perform dot product between the images or feature maps and filters [1]. In such process, one filter represents one specific feature the model intends to detect, and this feature can be located anywhere on the image or feature-map since the kernel is connected with every region with size of kernel on the image or feature maps [1](shown in figure 8). In addition, the kernel is sensitive enough to detect the feature since the feature is the region with high cross-corelation [1]. Besides convolutional layers, the activation function will serve as a threshold that only allows the acceptable signal from previous convolutional layers to pass through, thereby making the system more complex so that the feature can be easily distinguished [9] [1]. The last type of convolutional block is the Maxpooling layer. The maxpooling layer performs the task to downsample the image or feature map, thereby reducing the computation and directing the next layers to focus on more detailed features. In conclusion, these 3 types of layers construct the uniqueness of convolution block.

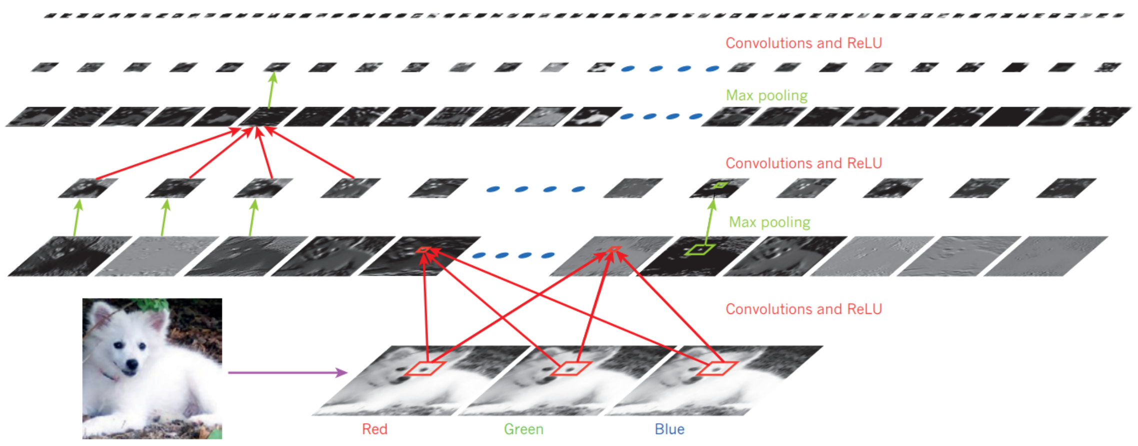


Figure 8: Example of how convolutional filters work. In this example, the image is in RGB color space, and the filters are detecting the their corresponding features [1]

In addition to convolutional block, the fully connected layers will serve as classifier. Each unit of the layer contains the weight matrix, and through the linear transformation and activation function the output becomes the input of next layers of units [9]. In contrast to traditional linear transformation, the activation function ReLU(shown as equation 4) will act in the same way as the convolutional layers to make the system more easily distinguish the feature.

$$output = \max(input, 0) \tag{4}$$

In addition, the last layer is normally a softmax classifier, and the result is typical the one with maximum probability in multinomial distribution(shown as equation 5 and 6).

$$class_i \sim Multinomial(\phi_i) \tag{5}$$

$$label = \arg\max_i(\phi_i) \tag{6}$$

where

$$\sum_{i \in <class>} \phi_i = 1$$

5.2.2 The Optimization and Training of CNN

In order to train the model to correctly predict the result, the model must minimize the difference between the predicted result and the actual result. Mathematically, we construct the loss function to calculate the difference between the true label and the predicted one. In the case of the softmax classifier as mentioned in 5.2.1, the loss function is cross-entropy shown as equation 7,8,and 9 [9].

$$L_i = -\log(\frac{e^{f_{y_i}}}{\sum_j e^{f_{y_j}}}) \tag{7}$$

which is equivalent to:

$$L_i = -f_{y_i} + \log\sum_j e^{f_{y_j}} \tag{8}$$

The loss function of entire dataset:

$$L_{total} = \sum_i L_i \tag{9}$$

In order to optimize the loss function, we need to find the derivative that leads us to the minima. In the process to find derivative of the loss function, we apply backpropagation to trace back each layers and use chain rule to deduct the derivative in respect to weights of whole system. [1] [9]

After that, the remaining task is to search for the minima of the loss function such that the loss function is minimized to show the most accurate labels. The common method is called Gradient Descent shown as the following algorithm 1 [9].

Algorithm 1 The Algorithm of Gradient Descent [9]

```
Initialize the weights of a CNN and run forward pass of the network:
Obtain the loss function:
    $L_{total} = \sum_i -\log(\frac{e^{f_{y_i}}}{\sum_j e^{f_{y_j}}})$
while epochs< max steps do
    Find the gradient: $\nabla_\theta L_{total}$ in respect to weights $\theta$
    Update the weight: $\theta_{new} = \theta - \alpha * \nabla_\theta L_{total}$, where $\alpha$ is the learning rate
    re-run the forward pass and obtain the new Loss Function
end while
```

Because it is not practical to plug every data points into the loss function to update the weights, algorithm normally takes randomly sampled batch to estimate the path of descent. This method is called Mini-Batch algorithm, and it is commonly referred as Stochastic Gradient Descent [9] shown as following algorithm 2. To notice, the batch size n is a multiple of 2. Furthermore, completion of

Algorithm 2 The Mini-Batch Algorithm

```
while The data is not completely run through do
    Select a Mini-Batch of size n and do Algorithm 1
    Do not return these data points back to the entire dataset
end while
```

this process is called 1 epoch.

The advanced way to do Gradient Descent also includes setup of a value γ whose meaning is how much the learning rate decay after one epoch to better search for the path of descent. [9]. Additionally, the analogy of kinetic energy and momentum can also serve as an advanced version of SGD algorithm to further improve the efficiency of search.

5.2.3 The Advantages and Disadvantages of CNN

The major advantages of this algorithm are:

A. The feature can be captured regardless of its location [1]

B. The users do not have to design the filters to extract feature as mentioned in summary [8]

C. The negative effects of variance of lights can be reduced because the model is trained to learn the effect of noise. [2]

However, it is apparent that this algorithm has several major issues:

A. The model requires a very large dataset to train because it has to cover the as many situations as possible [9]. However, it is difficult to collect the dataset of emotion. [2].

B. The model takes a very long to train from beginning(2 to 3 weeks) [2]

C. The training requires machine with very good handware, and they are expensive and consumes a large amount of energy. [9] [2]

In the next section, we are going to approach these issues by designing our own solutions.

6 Our Approach to Apply CNN

From previous section, we understand CNN does solve the issues of conventional method such as variance of feature location and noise surrounding the face. It, however, has its own requirement in formation of training process. In this session, our own solution will be presented to improve the model to fit the local dataset and constraints.

6.1 Transfer Learning and Fine-Tuning

6.1.1 Introduction of Transfers Learning

In response to the issue of long train period, the method of transfer learning is applied to reduce the length of training period. The term transfer-learning refers to the process of training a pre-trained model to better predict the inputs of local dataset. Normally, the convolutional blocks of a pretrained model acts as the feature extractor, and the local training only takes place on the fully connected part to train the weights of classifiers. Commonly we refer this process as fine-tuning of the pretrained model [9]. The training period will be reduced from 2 to 3 weeks to 1 to 3 days.In our project, the VGG_16 network developed by University of Oxford is used to be fine-tuned.

6.1.2 The Structure and Function of VGG_16

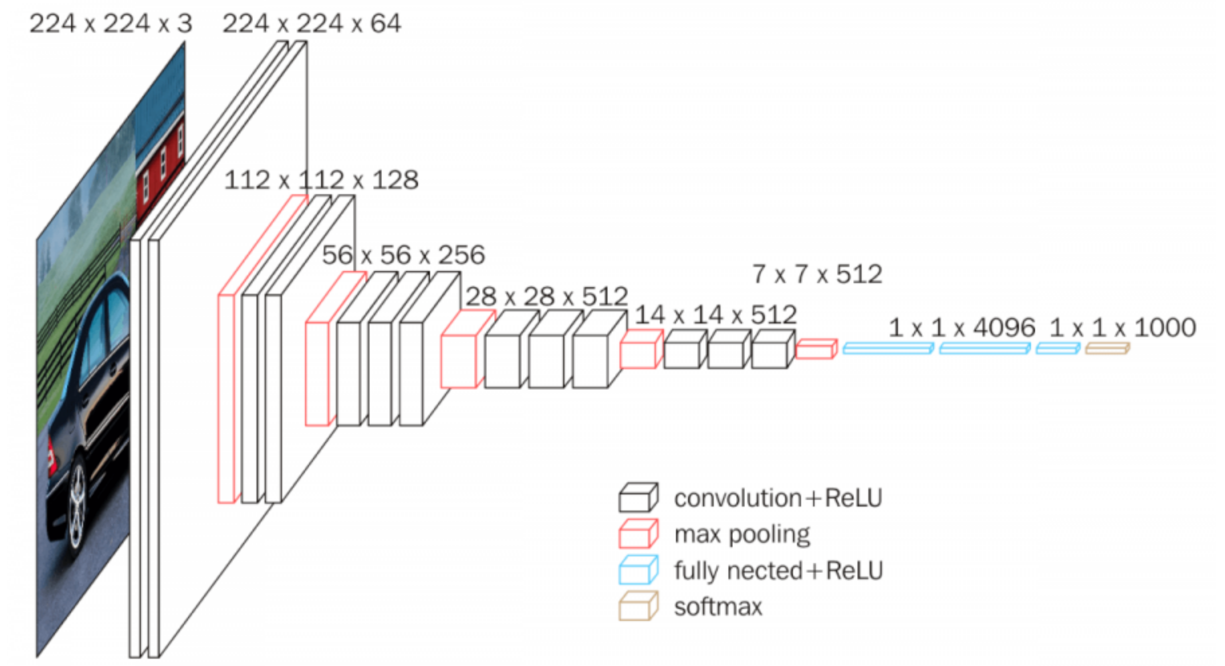


Figure 9: This is the diagram to show the dimensions of filters and layers of VGG_16 [11]

The pretrained model VGG_16 was developed by University of Oxford and introduced on their paper (see [12])in the topic of application of very deep network in the field of object detection. In their paper, the researchers note that network consists of very deep architecture of layers that can detect object from 1000 classes, and their top 1 accuracy rate is around 70% to 73% [12]. The figure 9 illustrates the architecture of VGG_16.

This network consists of 134 millions trainable weights, and the research team used 1.3 million images for training, which indicates in average there are 1300 images per class [12].

6.1.3 Our Method of Fine-Tuning

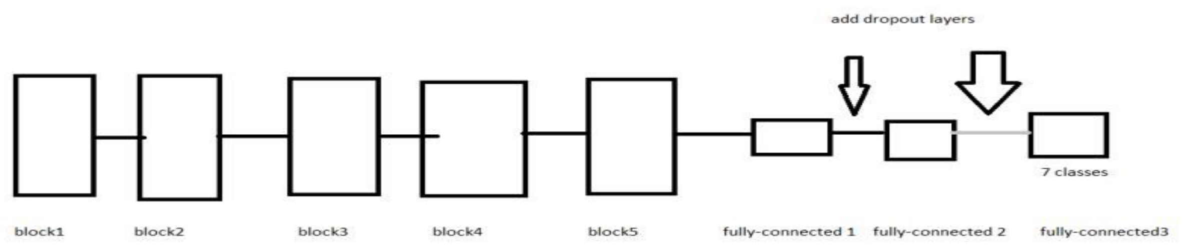


Figure 10: This is the diagram of architecture of the model we fine-tune. Diagram made by Chieh-En Li

Because the network is already capable of captured the some lower-level feature of object in a random background, we will only need to train the block that extracts the high-level detailed feature and the fully connected layers that serve as classifier. In addition to that. we also reset the softmax classifier from 1000 classes to 7 classes since we only have 7 emotions. Furthermore, when the paper was published, the over-fitting prevention technique called dropout layer was not popular [9]. Therefore, our method also adds the dropout layers in between fully connected layers to reduce the effect of overfitting caused too complex system. Finally, the diagram of structure is shown as figure 10.

6.2 Data Preprocessing and Augmentation

6.2.1 The Preprocessing on Original Dataset

The original data is from both The Japanese Female Facial Expression (JAFFE) [4] [5] [6] and CK/CK+ [3] [7]. The detail of original dataset is as following:

- JAFFE: This dataset consists of 200 images of 7 basic emotions with label included in the their file name. The image only consist of the face and do not have a background.
- CK/CK+: This dataset consists of 900 images of 7 basic emotions with label included in some different files. The images have face with background of static video frame.

Because the CK/CK+ dataset does not have label included in the file name and has the background, we apply the code written in [2] to convert the file name and image content in the format of JAFFE. After that, we divide dataset into 3 portions: training, validation, and test.

6.2.2 Our Methods of Augmentation

Algorithm 3 The Algorithm of Data Augmentation

```
Function of Random noise()
if mode = one type of noise then
   Add noise into image
else
   Keep the original state
end if


Function of Flip()
if mode = Add flip then
   Perform flip
else
   Keep original state
end if

Function of Rotation()
if the degree = certain degrees then
   Perform rotation in such degree
end if

Main function to combine these 3 effects
set up array with mode of random noise
set up array with mode of whether to flip the image
set up array with degrees of rotation
import image file
for each image in entire image dataset  do
   for one degree in degree array do
      for one mode in noise array do
         for one mode in flip array do
            augmented data = flip function(rotation function(random noise function(image)))
            Save the data in corresponding directory of labels
         end for
      end for
   end for
end for
```

In practice of other researches(see [9] [12]), each class typically consists of at least 700 to 1000 images to form the training dataset as mentioned in section 5 of the report. However, we do not have such large amount publicly available dataset for the project to conduct training. We, therefore, apply data augmentation to generate more data from original dataset to artificially create the variance of lights and the shift of locations.

In our project, we combine 3 techniques of effects to complete the augmentation on training and validation dataset.(shown in algorithm 3)

1. **Random noise:**
   We added three kinds of noises for augmentation(Gaussian, Salt, and Poisson)
   - Gaussian noise: The noise is in a Gaussian distribution and randomly added into images
   - Salt: The noise is to add 1 randomly into images
   - Poisson: The noise is in a Poisson distribution and randomly added into images .
2. **Horizontal flip:**
   Adding an effect of flipping the images horizontally.
3. **Rotation:**
   We rotate the image from 0°to 50°by increment of 10°.

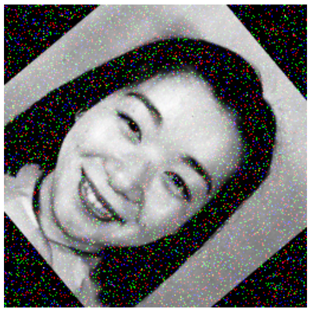
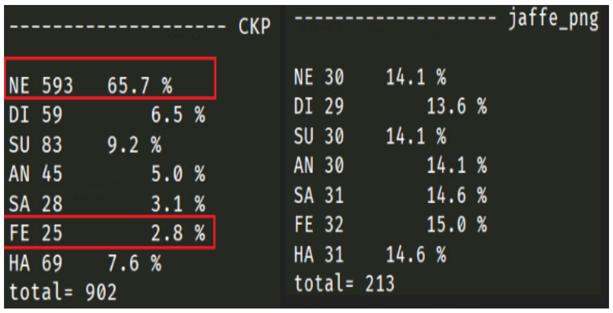


Figure 11: This is an example of augmented data. The original image is from JAFFE [4] [5] [6]

The augmentation generates 48 times of images from our original dataset of training and validation. Therefore, we have 43,000 training images in total.

6.2.3 Biased Dataset and Solution

The other issue of dataset is that our original dataset is highly biased in favor of Neutral(shown in figure 12) As results, we randomize each class to similar amount of data (around 2000 to 2200



```
-------------------- CKP     ------------------- jaffe_png

NE 593   65.7 %              NE 30    14.1 %
DI 59        6.5 %           DI 29        13.6 %
SU 83    9.2 %               SU 30    14.1 %
AN 45        5.0 %           AN 30        14.1 %
SA 28        3.1 %           SA 31        14.6 %
FE 25        2.8 %           FE 32        15.0 %
HA 69    7.6 %               HA 31    14.6 %
total= 902                   total= 213
```

Figure 12: This shows the bias of the dataset in favor of neutral. Made by Chieh-En Li

images) so that the data is balanced when we train the algorithm.

6.2.4 The Normalization of Data

Before we start the training, we also have to normalize the data. Though we may not need to normalize data by manually writing program on that, it is still important that these procedures should be included before the model can do either training or prediction.(briefly mentioned in 4.1) The normalization of our solution includes [9]:

- Rescale each pixels to 0 to 1
- center the mean to 0
- divide each point by the standard derivation

Some advanced methods include PCA which removes the correlation between the data and projects the data onto the eigenspace of its co-variance matrix and whitening which normalize the data point on eigenspace. [9] However, we do not consider these two methods in this project.

6.3 The Training

6.3.1 The Environment of Training

One of the disadvantages of CNN is that it requires a very powerful or groups of powerful GPU to complete the training because GPU has dedicated memory to compute the algorithm of Gradient Descent [9].In addition, in order to construct the model for machine to train the algorithm, we also need a framework to build the network. There are several popular frameworks [9]:

- caffe
- Tensorflow
- Keras
- PyTorch

For the purpose of this project, we will use Keras in Tensorflow on Google Colab machine to train our network. Google Colab has accessiable GPU memory from 8GB to 14GB, which satisfies the requirement of hardware.

6.3.2 The Parameters of Training

The following information is the parameter of training

- Training Batch size: 128
- Step/Epoch: 120
- Validation Batch size:16/32

- Step for validation;100/50
- γ: 0.1
- Momentum:0.9
- Nesterov Momentum:True

In practice, we train the model for 4 times in the order of 75 epochs, 25 epochs, 30 epochs, and 30 epochs to obtain a final model. [13]

6.4 Evaluation and Prediction

The evaluation of the model will include testing of model against the test portion of original dataset and comparison between untrained model and trained model. We generate the confusion matrix to analyze the model.

7 Result and Functionality of Solution

7.1 The Result of Training Accuracy and Loss Function

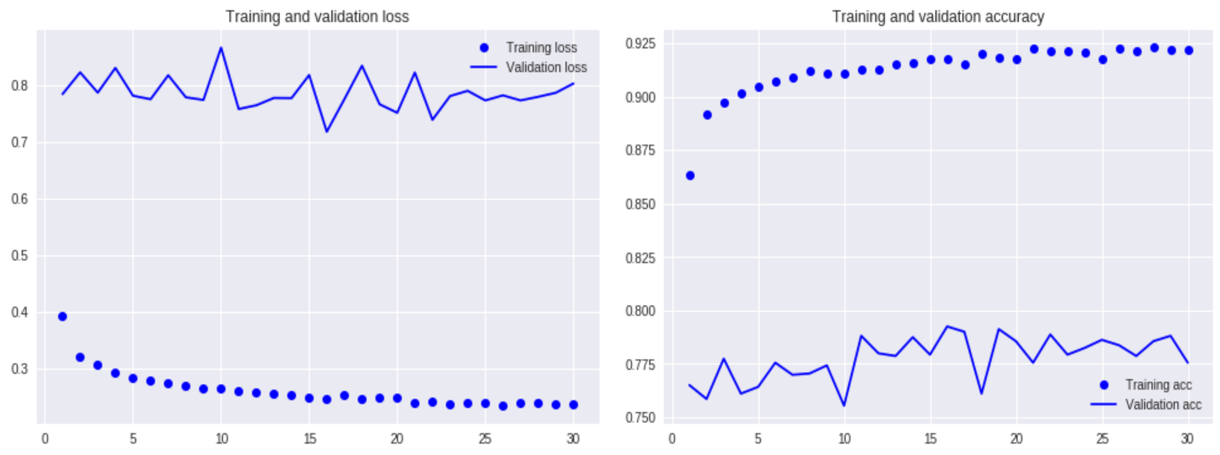
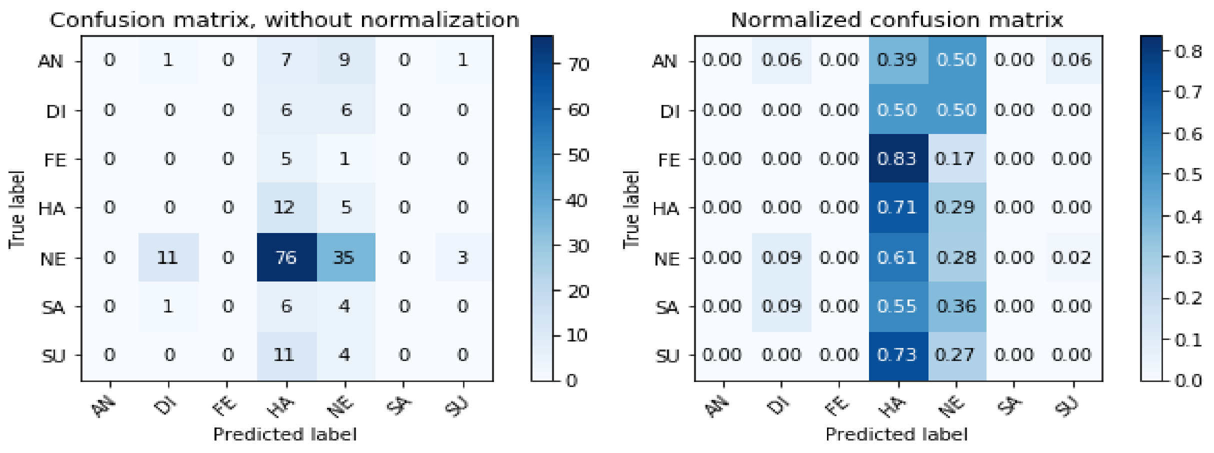


Figure 13: These are plots of training and validation. The left is loss vs epochs, and the right is accuracy vs epochs [13]

After the training, the training accuracy reaches around 95% ,and the validation accuracy reaches around 81%. One of the pairs of plots is shown in figure 13.

7.2 The Results of Test on lab condition images

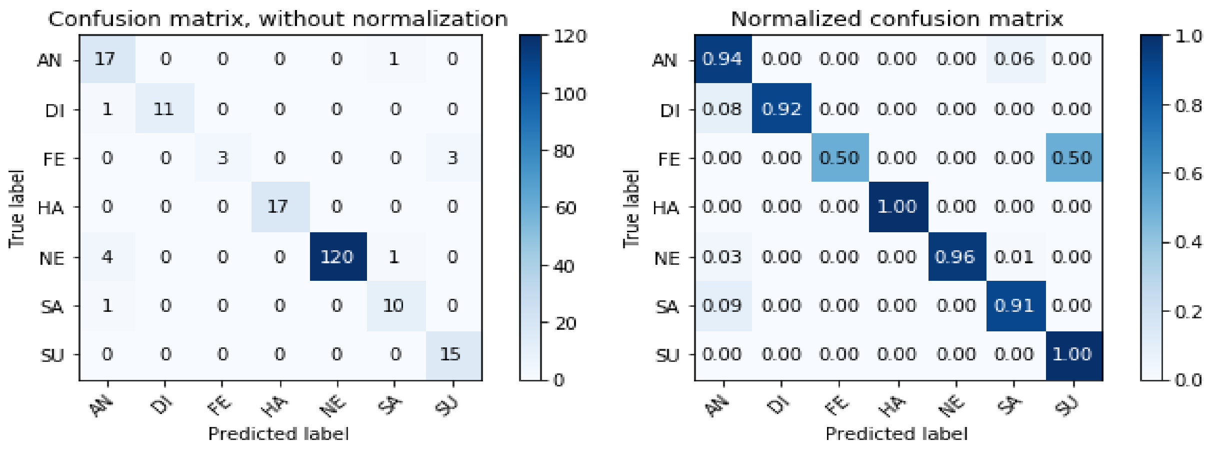
In order to test our model, we will use the test portion mentioned in 6.2.1 to conduct the test of image on label condition. The number of images is slightly more than 200, and since the bias of the original dataset the test dataset consists of mostly neutral images. In general, the accuracy is around 94% for the trained final model.However,the system basically randomly guesses the output when we do the test on the untrained model, thereby generating random results. These are shown as confusion matrices below



	AN	DI	FE	HA	NE	SA	SU
AN	0	1	0	7	9	0	1
DI	0	0	0	6	6	0	0
FE	0	0	0	5	1	0	0
HA	0	0	0	12	5	0	0
NE	0	11	0	76	35	0	3
SA	0	1	0	6	4	0	0
SU	0	0	0	11	4	0	0

	AN	DI	FE	HA	NE	SA	SU
AN	0.00	0.06	0.00	0.39	0.50	0.00	0.06
DI	0.00	0.00	0.00	0.50	0.50	0.00	0.00
FE	0.00	0.00	0.00	0.83	0.17	0.00	0.00
HA	0.00	0.00	0.00	0.71	0.29	0.00	0.00
NE	0.00	0.09	0.00	0.61	0.28	0.00	0.02
SA	0.00	0.09	0.00	0.55	0.36	0.00	0.00
SU	0.00	0.00	0.00	0.73	0.27	0.00	0.00

Figure 14: One of the random results the untrained model generates [14]



	AN	DI	FE	HA	NE	SA	SU
AN	17	0	0	0	0	1	0
DI	1	11	0	0	0	0	0
FE	0	0	3	0	0	0	3
HA	0	0	0	17	0	0	0
NE	4	0	0	0	120	1	0
SA	1	0	0	0	0	10	0
SU	0	0	0	0	0	0	15

	AN	DI	FE	HA	NE	SA	SU
AN	0.94	0.00	0.00	0.00	0.00	0.06	0.00
DI	0.08	0.92	0.00	0.00	0.00	0.00	0.00
FE	0.00	0.00	0.50	0.00	0.00	0.00	0.50
HA	0.00	0.00	0.00	1.00	0.00	0.00	0.00
NE	0.03	0.00	0.00	0.00	0.96	0.01	0.00
SA	0.09	0.00	0.00	0.00	0.00	0.91	0.00
SU	0.00	0.00	0.00	0.00	0.00	0.00	1.00

Figure 15: The confusion matrix of trained model [14]

7.3 The Code Certification Test

However, when we try to test our code in wild test, the results turn out to be significantly lower (around 37%). As results we can only conclude that our model only fits the image of lab condition.

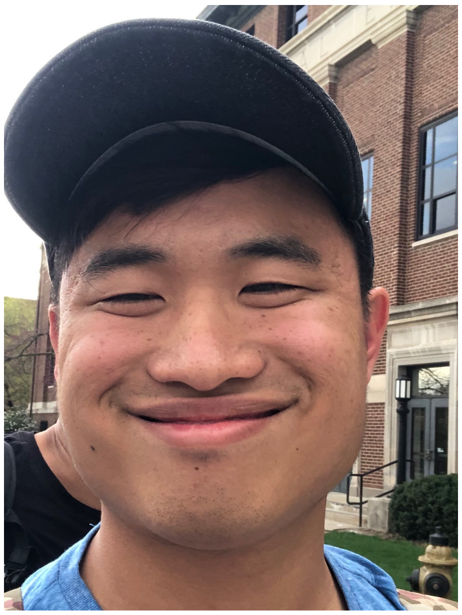


Figure 16: One of the co-author uses his photo to test, and it mistakenly classifies this image of happy as sad.

8 Analysis and Evaluation of Model

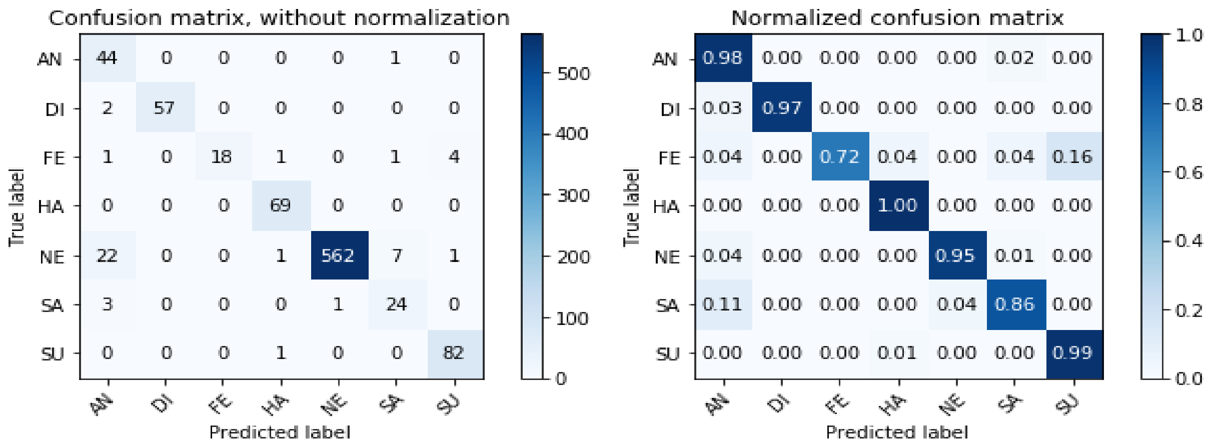


Figure 17: The confusion matrix of original ck plus dataset. The matrix is used for analysis only [14]

In analysis of the model on original CK+(figure 17), we can conclude that the lab condition images in general trains the model extremely well-fit the lab condition images. However, we still can notice that the emotion fear has the worst results among all classes in both ck+ images and test images. The reason for that may be because fear and surprise have very similar AUs in the facial regions. Even if we manually see the images from these two classes, it is possible to get confused on them.

Besides the images under lab conditions. when we try to use system to deliver the prediction of wild emotion classification, we find out that the accuracy is significantly reduced to round 37%. In comparison to [10], which yields to 41% and to [2],which has 51% in testing, our model does not deliver a good result.(figure 16 is an example)

9 Future Expectation and Improvement of The Model

In the future, the model will be trained and tested on a boarder sets of data to not only fit the training dataset or lab condition images but also to achieve better accuracy on wild emotion recognition. In addition to this, the model is also expected to deliver the real life application in education to improve the learning and interactions in class(as [15])

10 Reference and Acknowledgement

This idea of this project comes from the HappyNet, a project for Stanford University CS31n. The source code of data preprocessing comes from the HappyNet [2] directly with some minor modification. The source code for Keras training uses some of the code from the book Deep Learning with Python [13]. The code of confusion matrix comes from sckit-learning, and we did some minor modification. [14] In addition to that, we also read Keras Documentation as a reference [16]

References

[1] B. Y. LeCun, Yann and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, May 28 2015.

[2] D. Duncan, G. Shine, and C. English, "Facial emotion recognition in real time." Available:`http://cs231n.stanford.edu/reports/2016/pdfs/022_Report.pdf` [Accessed: 2019-05-03].

[3] P. Lucey, J. F. Cohn, J. S. Takeo Kanade, Z. Ambadar, and I. Matthews, "A complete expression dataset for action unit and emotion-specified expression," *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition-Workshopss*, pp. 94–101, June 12, 2010.

[4] M. Lyons, S. Akamatsu, M. Kamachi, and J. Gyoba, "A brief review of facial emotion recognition based on visual information," *Automatic Face and Gesture Recognition, 1998. Proceedings of Third IEEE International Conference on*, pp. 200–205, Apirl 14,1998.

[5] M. Lyons, S. Akamatsu, M. Kamachi, and J. Gyoba, "Coding facial expressions with gabor wavelets," *Proceedings of Third IEEE International Conference on Automatic Face and Gesture Recognition, 1998.*, pp. 200–205, April 14 1998.

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

[7] T. Kanade, J. F. Cohn, and Y. Tian, "Comprehensive database for facial expression analysis," *Proceedings of Fourth IEEE International Conference on Automatic Face and Gesture Recognition (Cat. No. PR00580)*, pp. 46–53, 2000.

[8] B. C. Ko, "A brief review of facial emotion recognition based on visual information," *Sensors*, vol. 18(2), p. 401, 2018.

[9] F.-F. Li, J. Johnson, and S. Yeung, "Class notes of cs 231n of stanford university." Available:`http://cs231n.github.io/` [Accessed: 2019-05-03].

[10] G. Levi and T. Hassner, "Emotion recognition in the wild via convolutional neural networks and mapped binary patterns," *ICMI '15 Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*, pp. 503–510, November 09-13, 2015.

[11] M. ul Hassan, "Vgg16 – convolutional network for classification and detection." Available:`https://neurohive.io/en/popular-networks/vgg16/` [Accessed: 2019-05-04].

[12] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv*, pp. 1409–1556, September 4, 2014.

[13] F. Chollet, *Deep Learning with Python.* Manning Publications Company, December 22 2017.

[14] "Confusion matrix." `https://scikit-learn.org/stable/auto_examples/model_selection/plot_confusion_matrix.html#sphx-glr-auto-examples-model-selection-plot-confusion` Accessed: 2019-05-04.

[15] K. L.B and L. P. GG, "Student emotion recognition system (sers) for e-learning improvement based on learner concentration metric," *Procedia Computer Science*, vol. 85, pp. 767–776, 2016. Available:`https://blog.paralleldots.com/product/facial-emotion-detection-using-ai/` [Accessed: May 4 2019].

[16] "Keras documentation." `https://keras.io`. Accessed: 2019-05-04.

11 The Distribution of Work

Lanqing Zhao: Section 1,2,3,4,5,6,7,8,9,and 10

Chieh-En Li: Section 3,6,7,9,10 and bibliography