Countenance Classifier: How are you feeling today?

1 Introduction

The study of facial expressions and micro expression recognition had gained a lot of traction and interest in recent times given their potential of success in analyzing and predicting the emotion being depicted by the facial expression [1]. Many law enforcement and security industries are keen in capturing the micro and facial expressions through video surveillance and use the input to predict the true emotion being expressed so that they can take the required action necessary. For e.g. In the sales and marketing industry, the system can capture the emotion of new customers and make recommendations based on the predicted expression. This could dramatically help improve generating the appropriate response. There are a vast number of new and available techniques which can be used to train models and benchmark results to analyze which model is giving the highest accuracy. In our project, we have experimented with 3 facial expression recognition approaches: Traditional Face recognition techniques, a Simple Convolution Neural Network, and a Neural Network with pre-trained Inception V3 Outputs. In addition, we also developed a live demo to capture a live stream from a video camera, apply our predative model on the capture images and display the final predicted facial expression via a Live Feed using the laptop camera.

2 Approach

2.1 Data Collection

Initially, we selected a facial expression dataset from Kaggle [2] for the project. The dataset has approximately 35, 000 images containing 7 emotion classes such as Angry, Disgust, Fear, Happy, Sad, Surprise and Neutral. Without GPU computation resources, we can only use CPU to train our models, which makes the training very time-consuming as shown in Table 1:

Table 1: Results of experiments performed using the Kaggle Dataset

Description	Accuracy on Training	Accuracy on Test	Time spent
Neural Network with pre-trained InceptionV3 Output (Inception training disabled)	0.71	0.62	9 hours
Neural Network with pre-trained InceptionV3 Output (Inception training enabled)	0.86	0.71	16 hours

Such time-consuming requirement makes Kaggle dataset infeasible for our experiment. Hence, we decided to use the CK Dataset [3] supplemented with our self-created images and focus our models to predict 3 facial expressions: Happy, Neutral and Surprise. Our final dataset contains 400 images of these 3 facial expressions.

2.2 Predictive Model Development

2.2.1 Traditional Face Recognition Techniques

There are two techniques, namely Eigenface and Fisherface, which have been traditionally used in face recognition tasks:

- Eigenface: Eigenface is a set of eigenvectors used in the computer vision problem of

human face recognition. The Eigenface technique uses Principal Component Analysis(PCA) to get a smaller set of basis images which represents the original training images. Classification can then be done by comparing how the basis set represents faces.

- Fisherface: Fisherface technique uses Linear Discrimination Analysis(LDA) to find a subspace that maps the sample vectors of the same class in a single spot of the feature representation and those of different classes as far apart from each other as possible.

Kriegman D. compared the two methods and concluded that "Fisherface method has error rates that are lower than those of the Eigenface technique for tests on the Harvard and Yale face databases" [4]. This is understandable because Linear Discriminant Analysis explicitly models the difference between the classes of data (supervised learning), while Principle Component Analysis does not consider any difference in class (unsupervised learning). In our experiment, which is detailed further in the coming sections, we will train our dataset on these two models and observe the results and analyze which performed better and study the comparative results.

2.2.2 Simple Convolutional Neural Network

Deep Learning techniques have been shown to excel at image classification tasks. Convolutional networks are the state of-the-art solution for most of computer vision tasks, since 2014 with the deep neural network generations this technique significantly improves the results obtained in various benchmarks. According to Wojna J., "One interesting observation was that gains in the classification performance tend to transfer to significant quality gains in a wide variety of application domains. This means that architectural improvements in deep convolutional architecture can be utilized for improving performance for most other computer vision tasks that are increasingly reliant on high quality, learned visual features." [5].

We implemented a simple Convolutional Neural Network to perform facial expression classification. Our Convolutional Neural Network architecture is as below:

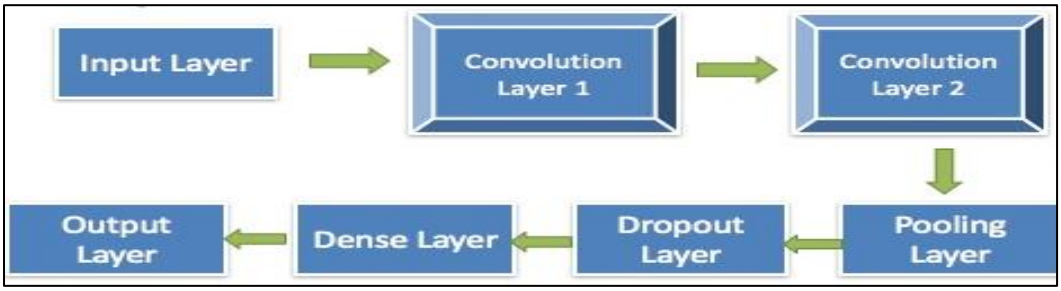


Figure 1: Architecture of Simple Convolutional Neural Network

Details information of each layer:

- Input layer: 2-D matrix represents 48x48 grayscale image
- Convolution layer: 32 filters with size 3x3 with 'ReLu' Activation function
- Convolution layer 2: 32 filters with size 3x3 with 'ReLu' Activation function
- Pooling Layer: Max Pooling with size 2x2
- Dropout Layer 1: 25% dropout rate
- Dense Layer: 128 hidden nodes with 'ReLu' Activation function
- Drop out Layer 2: 50% dropout rate
- Output Layer: 3 nodes corresponding to 3 classes
- Loss Layer: Softmax loss

2.2.3 Neural Network with pre-trained InceptionV3 Output

We also performed experiments with the inception V3 model which is state of the art deep convolutional neural network. We started from pre-trained weights which were trained using ImageNet 2012 Challenge training data set. Then, we added a few neural network layers top of the inception V3 model. Figure 2 illustrates this architecture:

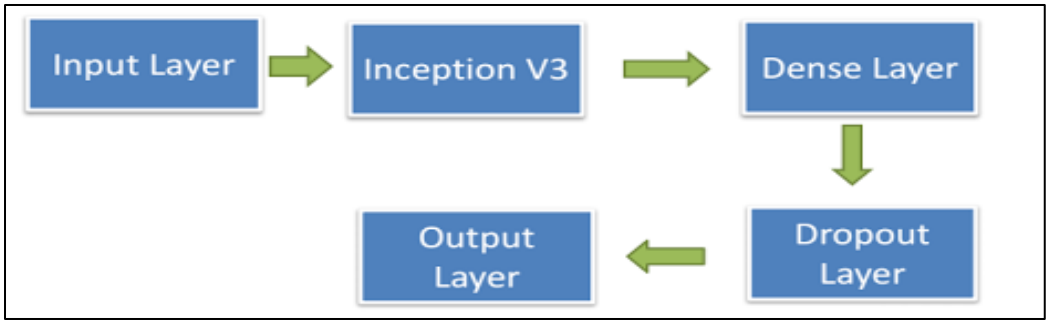


Figure 2: Architecture of Neural Network with Pre trained Inception V3 output

2.2.4 Data Augmentation

Since our data set is not large (400 images), data augmentation might help to boost the performance. We performed data augment for our data with: Rotation Range – 20, Width Shift Range – 0.2 and Height Shift Range – 0.2, as illustrated below:

Original Picture **Augmented Picture Samples**

2.3 Live Demo

We implement a real-time demo to see how our facial expression recognition models perform in real-life setting where there are many challenging factors affecting the image captured for classification, such as blur image (due to motion), lightning, facial angles, and facial wearables (eg. glasses).

3 Experiments and Methodology

3.1 Resources and Technologies

The hardware resources used in our experiments are as follows:

- 16 CPUs with 2.9 Ghz
- 32 Gigabytes of memory
- 8 Gigabytes of storage SSD Amazon AWS machine.

The source code for the project is located on our Github repositories [6].

3.2 Experiment Results

3.2.1 Traditional Face Recognition Techniques

The OpenCV library has an out-of-the-box implementation for both EigenFace and FisherFace techniques [7]. We trained both the models using our dataset with 10 times and report the average validation accuracy.

Table 2: Validation accuracy using traditional approaches

Model	**Validation Accuracy** (without Data Augmentation)	**Validation Accuracy** (with Data Augmentation)
Fisherface	0.87	0.88
Eigenface	0.66	0.68

As we can see, the FisherFace model performs better in the facial expression classification task compared to the EigenFace model. This result is inline with conclusion made by D.J. Kriegman [3]. In addition, we found that the data augmentation process helps improve overall validation accuracy, but not significantly.

3.2.2 Simple Convolutional Neural Network

We used Keras on Tensorflow to implement a simple Convolutional Neural Network to perform facial expression classification. We train our network with Adadelta optimizer in 20 epochs:

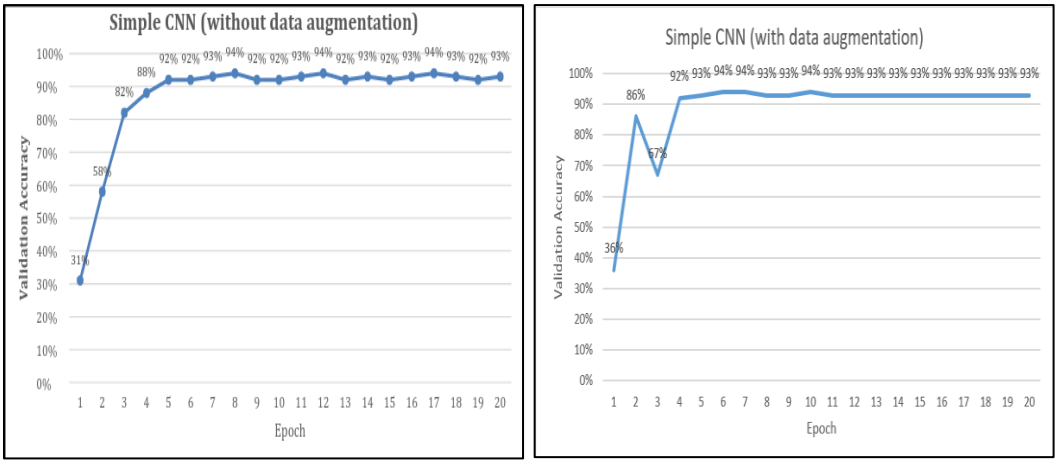


Figure 3: Validation Accuracy with Simple CNN

The best validation accuracy is 0.94. We also observed that with data augmentation, we get almost the same result. As we can see, using a simple Convolution Neural Network with 2 convolutions, we manage to achieve a significantly higher validation accuracy compared to traditional face classification techniques.

3.2.3 Neural Network with pre-trained InceptionV3 Output

The experiments were performed with this architecture and we tried to find the best parameters for this complex model to achieve the best results on the proposed task. Training parameters used in all the tests are:

- Loss - Categorical Crossentropy
- Optimizer - Stochastic Gradient Descent, except when in some cases we are using Adadelta
- Metric - Categorical_accuracy
- Batch Size – 10 & Samples per epoch – 308
- Validation Samples – 100 & No. of Epochs – 15
- Classes – Happy, Neutral & Surprise

Table 3: Validation accuracy using model with Inception V3 (without data augmentation)

Description	**Training Accuracy**	**Validation Accuracy**

Inception training disabled (Mini-batch GD)	0.916	0.78
Inception training disabled (Adadelta)	0.97	0.88
Inception training enabled (Mini-batch GD)	0.99	1.0
Inception training enabled (Adadelta)	0.99	1.0

Table 4: Validation accuracy using model with Inception V3 (with data augmentation)

Description	Training Accuracy	Validation Accuracy
Inception Enable/Augmentation only in the training images (Mini-batch GD)	0.98	0.35
Inception Enable/Augmentation in the training and test images (Mini-batch GD)	0.87	0.68
Inception Enable/Augmentation in the training and test images (Adadelta)	0.99	1.0

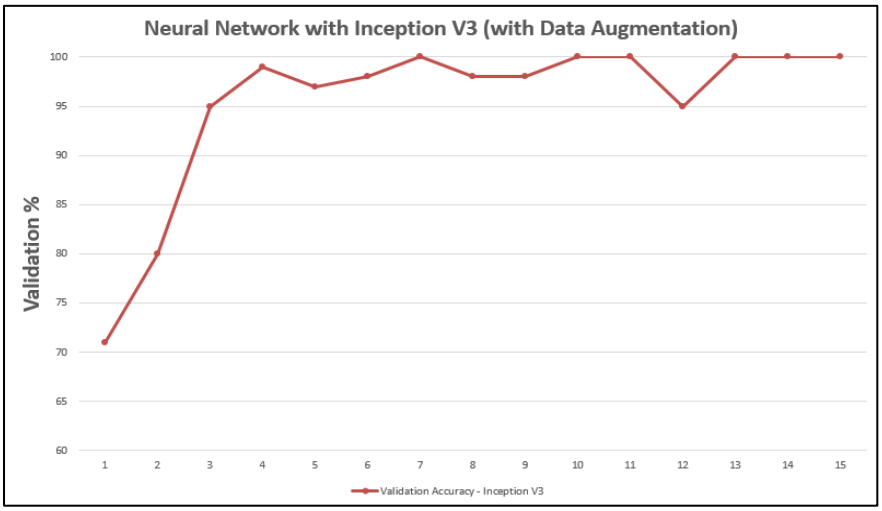


Figure 4: Validation Accuracy for Neural Network with Inception V3

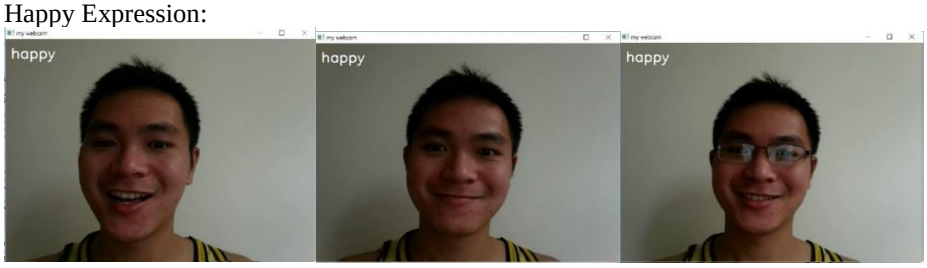
Neural Network with Inception V3 achieve the best performance with 1.0 validation accuracy when we enable Inception layers training and train the model with Adadelta optimizer.

3.3 Live Demo Results

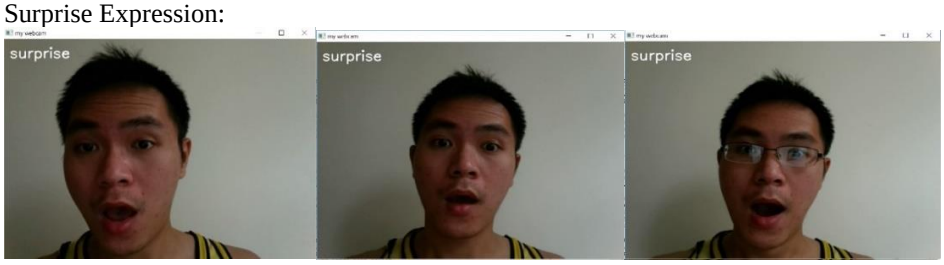
We used the OpenCV library [7] to capture images with the laptop's front-camera and detect faces using OpenCV's Haar Feature-based Cascade Classifiers. The detected faces will be converted to grayscale images, which serves as an input for our facial expression classification model. The predicted emotion will then be displayed on the screen as the output. We also tried to capture several images as inputs for our facial expression classification model and used the voting scheme to predict emotion. This approach does help improve accuracy in the cases of blur and low light conditions, but it makes the real-time prediction much more sluggish. Hence, we removed this approach in our live-demo. Below are some illustrations of the live-demo:

Correct Recognition:

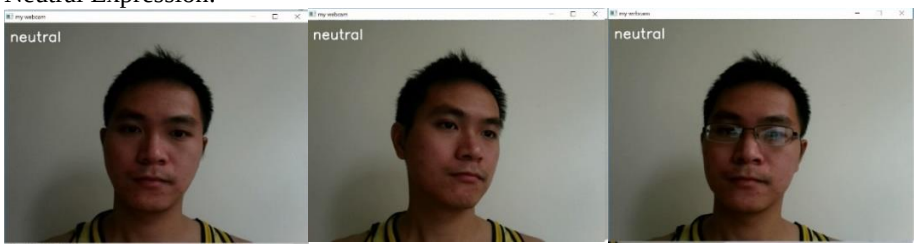
- Happy Expression:



- Surprise Expression:

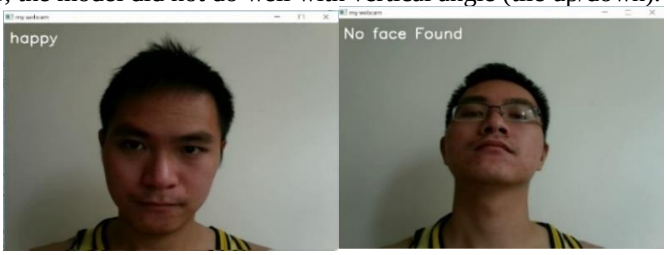


- Neutral Expression:



Incorrect Recognition:

There were also certain cases where we received incorrect recognitions, the prediction kept switching between surprise/neutral and happy/neutral expression as seen below. Based on the live-demo, it seems the model recognizes facial expressions well with or without glasses, even in a considerably low-light condition. The model also does well with horizontal angle (tile left/right). However, the model did not do well with vertical angle (tile up/down).



4 Conclusions

We have described our experiment approaches and results for facial expression recognition. Our experiment shows that among the two traditional face recognition techniques, Fisherface has much better accuracy compared to Eigenface in facial expression classification tasks. The result also demonstrates that a simple Convolution Neural Network with just two layers of convolutions outperforms traditional face recognition techniques, and using neural network with pre-trained Inception V3 output further improves the model performance significantly.

Due to computing resource constraints, we were unable to experiment with a large facial expression data set, which would be interesting to do for model training and validation. Besides,

the Live demo is more challenging because of different conditions (lightning, blur, angles, glasses) and we should explore how to handle these conditions in subsequent experiments.

References

[1] Jun Z., Yong Y. and Martin L. (1997), *Face Recognition: Eigenface, Elastic Matching, and Neural Nets* [link]

[2] *Kaggle Facial Expression Dataset* [link]

[3] *CK Facial Expression Dataset* [link]

[4] Kriegmen D. (2002), *Eigenfaces vs. Fisherfaces: recognition using class specific linear projection* [link]

[5] Wojna J. (2015), *Rethinking the Inception Architecture for Computer Vision* [link]

[6] *Project Source Code* [link] [link]

[7] *OpenCV Facial Recognition documentation* [link]