Baskin Engineering
UC SANTA CRUZ

UNIVERSITY OF CALIFORNIA, SANTA CRUZ

TIM 209 FINAL PROJECT

Facial Expression Recognition

Yunzhe Li, Guanming Shen, Zhujian Miao

December 8, 2018

Abstract

Automated detection, recognition, and analysis of faces and facial expression are being implemented in an increasingly broad number of environments, from social media to surveillance. One of the most challenging and ethically charged subdomains within automated facial analysis is Facial Recognition of Emotions (FRE). Although significant efforts have been made in methodology development in FRE, existing approaches lack generalizability. We discuss some inherent problems and uncertainties with FRE, both technical and ethical. As proof of concept, we implement a convolutional neural net on the dataset from the Kaggle competition *Challenges in Representation Learning: Facial Expression Recognition Challenge*, and evaluate our results. We then compare our results to those from other challenge participants. We end with a discussion regarding the implications of both accurate and erroneous emotion detection in a variety of contexts.

1 Introduction

Human facial expressions play a vital role in our communication with each other, as well our understanding of each other's emotional states. We use this understanding to make a variety of important judgments, such as deciding who is trustworthy or dangerous, or knowing when a friend could use emotional support. It is arguably the most important nonverbal communication method, and is vital to any true understanding of human interaction. It is thus unsurprising that there have been attempts to schematize human emotions and their associated facial expressions in an attempt to understand them better. Ekman and Friesen identified six facial expressions that map to six basic emotions, that are universal among human beings . Although this schema has been criticized by cultural anthropologists as being non-universal, Ekman noted that no quantitative data has arisen to support their contention that human emotional expression is culture specific. Further, Ekman's cross-cultural research on facial expression indicated that no instance in the literature in which 70% or more of one cultural group labels an expression of one of the six universal emotions differently than another . This schema is thus typically used when trying to create classification models for machine learning purposes.

Just as accurate interpretation of facial expressions of emotion is central to both human interaction and social functioning, that same discernment is crucial for many Human Machine Interaction (HMI) applications. While humans are typically able to recognize emotional expression without particular effort, it is very challenging to train a machine to to the same. Such a system would need to deal with significant variability along several vectors, from skin texture and color, to orientation and posture, to lighting intensity and angle. We use FER2013 image data from the Kaggle competition *Challenges in Representation Learning: Facial Expression Recognition Challenge*, which are labeled according to the Ekman-Friesen schema and normalized to minimize variability.

Numerous algorithms have been developed for FRE. Significant work has been done with support vector machines and Bayesian classifiers. This tend to have modest success only in circumstances with minimal variability as to orientation and lighting (i.e., posed shots in controlled, consistent environments). We propose an approach using convolutional neural nets (CNNs). Although neural nets require substantial computing power and large training sets, they have the ability to extract undefined features. Further, their training tends to be generalizable. After testing our CNN, we compare our results to those of others, and draw conclusions.

2 Overview

FRE algorithms tend to have four steps: face detection, face registration, feature extraction, and classification. In the face detection step, the face is located in the image. In the face registration step, the face in the image is normalized to a template. In the feature extraction step, the "registered" facial image is represented as a numerical feature vector. There are numerous methods to extract features, such as measuring pixel intensities. The FER2013 data comes with the first three steps completed, leaving us with the classification task.

2.1 Data

The FER-2013 dataset includes 28,709 Training examples, 3,589 "Public Test" examples, and 3,589 "Private Test" data. They are based upon 48 by 48 pixel grayscale faces with 7 different emotions (0=Angry, 1=Disgust, 2=Fear, 3=Happy, 4=Sad, 5=Surprise, 6=Neutral). Examples are shown

Figure 1: Image examples from the FER-2013 data set

in Figure 1. The data set was created by Pierre Luc Carrier and Aaron Courville, using the Google image search API. They combined a set of words related to age, gender, race, and ethnicity, then combined them with 184 emotion-related keywords like terrified, depressed, etc. to form 600 search strings. The first 1000 images returned for each query were then cropped using OpenCV face recognition. Finally, human labelers went through the dataset to remove duplicates and correct labels and cropping. The final images were then resized to 48x48 pixels, converted to grayscale, then converted a vectors representing pixel intensities. The final count of the dataset contains 35,887 images, with 4,953 Anger images, 547 Disgust images, 5,121 Fear images, 8,989 Happiness images, 6,077 Sadness images, 4,002 Surprise images, and 6,198 Neutral images. The test sets have the same class proportions as the training set.

3 Methodology

The key architecture of our model is convolutional neural networks. We will try different number of layers, different kernel size and different strides to see which one has the best performance. The activation function of the result will be computed by softmax function. Softmax function can compute the probabilities of different cases in multiclass classification problems.

3.1 2-Layers CNN Architecture

For comparison purpose, we first developed a 2-layers CNN to see how it works.This network had two convolutional layers and one FC layer. In the first convolutional layer, we had 32 3x3 filters, with the stride of size 1, along with batch normalization and dropout, but without maxpooling. In the second convolutional layer, we had 64 3x3 filters, with the stride od size 1, along with batch normalization and dropout and also max-pooling with a filter size 2x2. In the FC layer, we had a hidden layer with 512 neurons and

Softmax as the loss function. Also in all the layers, we used Rectified Linear Unit (ReLU) as the activation function.
For the training process, we used all of the images in the training set with 30 epochs and a batch size of 128. We set the learning rate to 0.0001, and using the Adam Optimizer for the its fast converge rate. It can get up to 49% accuracy on the test set, which is not that good.

3.2 4-Layers CNN Architecture

To test the effect of adding more convolutional layers and FC layers to the network, we build a architecture which is introduced by paper Convolutional Neural Networks for Facial Expression Recognition. AS shown in Figure.2. The first convolutional layer had 64 33 filters, the second one had 128 55 filters, the third one had 512 33 filters and the last one had 512 33 filters. The input of the overall model is a 48*48 gray scale face and the output an array with 7 elements, and index of the array corresponds the emotions. Take this array as the input of softmax function to compute the probability of the each emotion. The batch normalization here is to solve the brightness diversity problem. For example, if we input two same faces with two different exposure time, i.e. the gray scale value of each pixel in one picture is larger than the other picture a fixed value. These two should output the exactly same results. To offset the brightness difference, we use the batch normalization technique to normalize the result of each layer. To prevent overfitting, we also do dropout at the end of each layer and add a L2 regularizer for all the weights in our model.

During the training process, we use the Adam Optimizer in TensorFlow with learning rate 0.001, to minimize the loss which is computed by the softmax loss of the output 7-element array of the model. We also set up a Google Cloud Computing engine to train our model with a 8-cores VM instance. When training, the whole train set batch was traversed 30 epochs. At each epoch, we batched the whole train set with the mini-batch size of 128. Since the overall train set has size of 32,298, we need to run 253 time steps to traverse the whole train set batch in each epoch. We tested our prediction accuracy on test set every time we finished one epoch to see how the accuracy improved. Besides that, we also computed the softmax loss on test set for analysis purpose. This time, we get 58% accuracy. The detailed results will be discussed in next section.

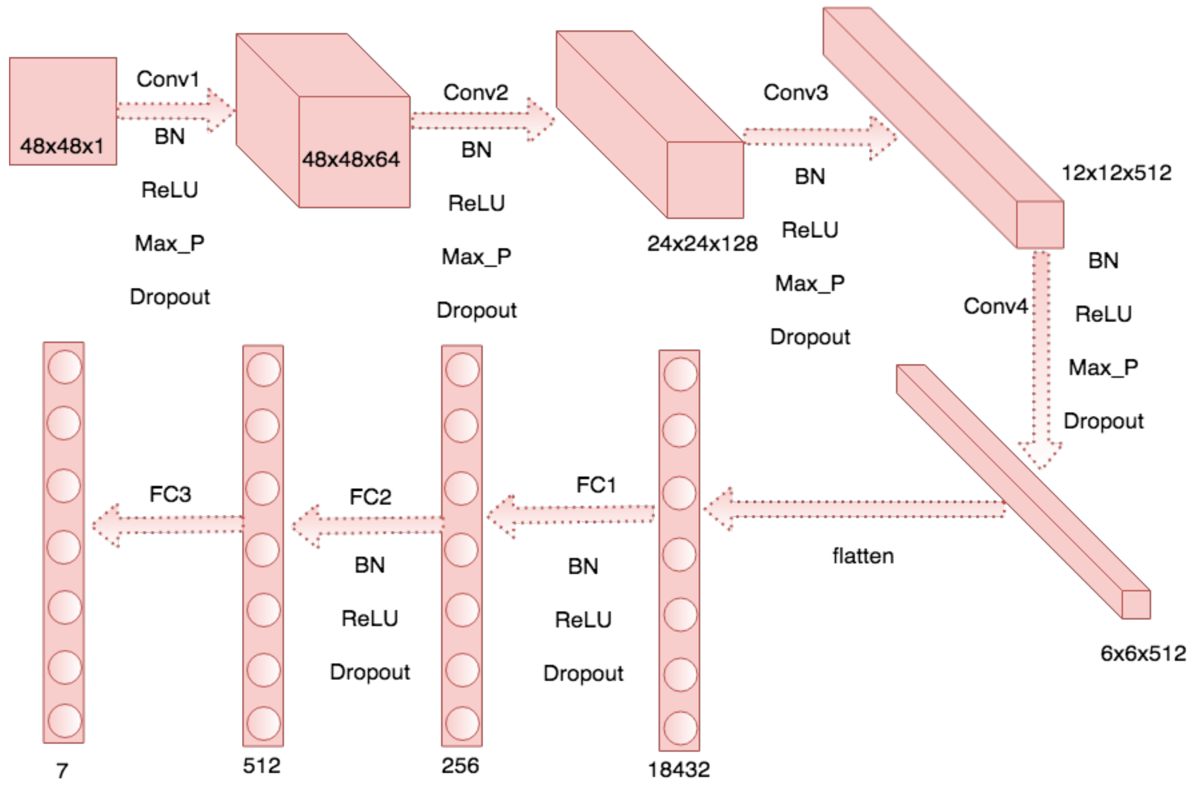

Figure 2: 4-layers CNN Architecture

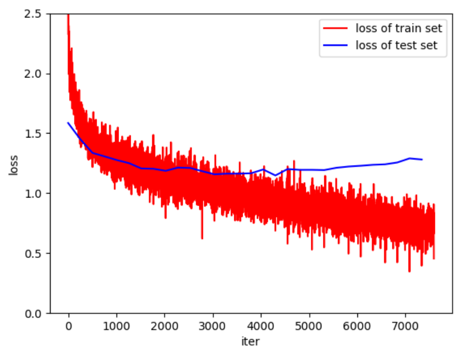

Figure 3: 4-layers CNN Loss

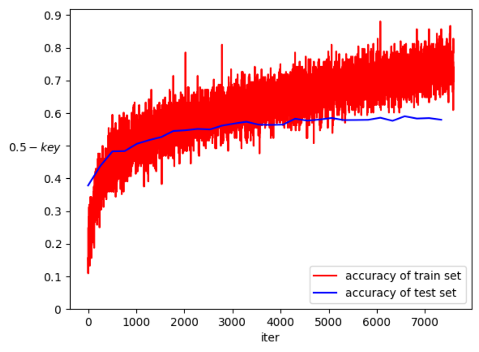

Figure 4: 4-layers CNN Accuracy

3.3 SVM

Besides the neural networks, we also want to test that how it performs when using some statistical learning algorithm. Here we decide to use support vector machine, with a linear kernal, provided by sklearn.The features we used are raw, grayscale pixel values. It were only able to achieve a maximum test set accuracy of 34%

4 4-layers CNN Experimental Results

Apparently, 4-layers CNN performs best on this task. Figure 3 and 4 display the obtained accuracy in different iterations for deep model. As you can see in Figure 3, our loss on the test set flattens out at about 2,000 iterations, even though it continues to improve on our training set. We see a similar thing in regards to our accuracy, as shown in Figure 4. This suggests that the remaining improvements to loss and accuracy are due to overfitting.

Our overall precision, recall, and F1 scores were fairly accurate, with an F1 of 0.57. Our best F1 score was for Happy at 0.80, while our lowest was for Fear at 0.32. See the classification report in Figure 6 for more details.

The confusion matrix in Figure 5 reveals more detail about our guesses, successful and not. It is interesting to see that the model performed well in predicting the happy label, which implies that learning the features of a happy face is easier than other expressions. Additionally, these matrices reveal which labels are likely to be confused by the trained networks. For

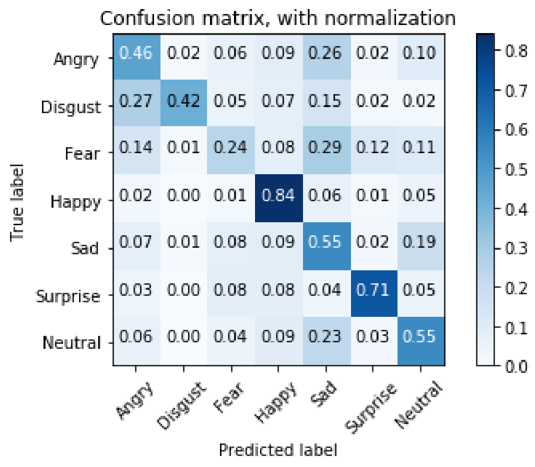

Figure 5: Confusion Matrix

```
             precision    recall  f1-score   support

      Angry       0.52      0.46      0.49       491
    Disgust       0.49      0.42      0.45        55
       Fear       0.46      0.24      0.32       528
      Happy       0.76      0.84      0.80       879
        Sad       0.40      0.55      0.46       594
   Surprise       0.72      0.71      0.72       416
    Neutral       0.55      0.55      0.55       626

avg / total       0.58      0.58      0.57      3589
```

Figure 6: Classification Report

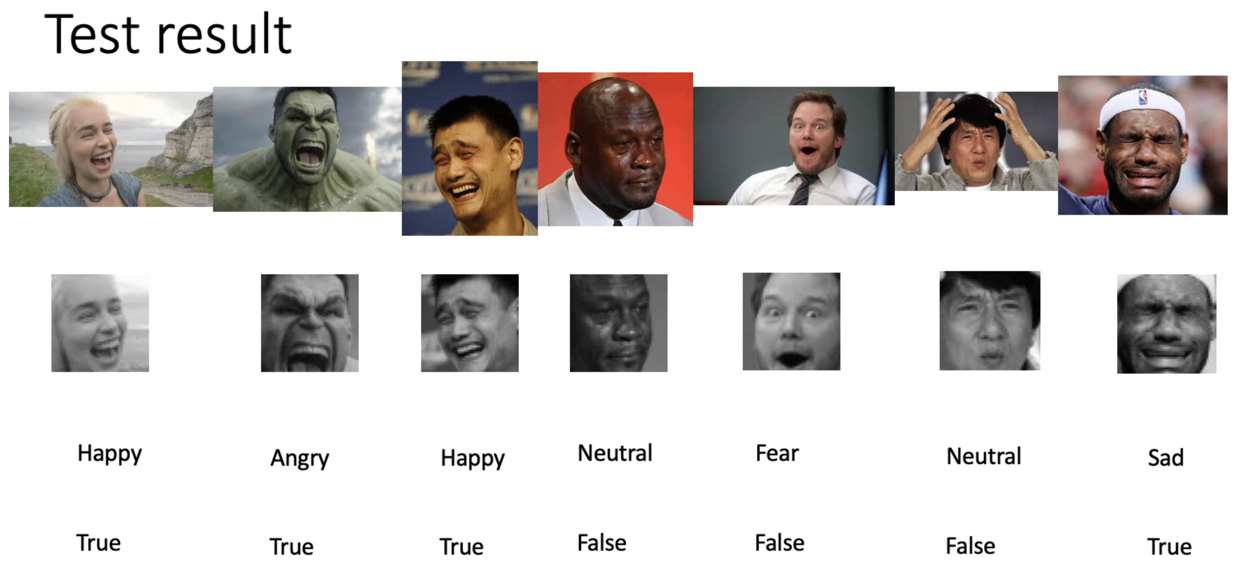

Figure 7: Test Result

example, we can see the correlation of angry label with the fear and sad labels. There are lots of instances that their true label is angry but the classifier has misclassified them as fear or sad. These mistakes are consistent with what we see when looking at images in the dataset; even as a human, it can be difficult to recognize whether an angry expression is actually sad or angry. This is due to the fact that people do not all express emotions in the same way.

5 Test on real world examples

To test our model on the real world faces, we developed a face cropper which is based on the Haar feature-based cascade classifier in OpenCV library.

Haar Cascades Algorithm is a machine learning based approach where a cascade function is trained from a lot of positive and negative images. It is then used to detect objects in other images. OpenCV already contains many pre-trained classifiers for face, eyes, smiles, etc. So here we can the pre-trained face detector that when you input a photo with face, it can return the 4 coordinates of a rectangular that crop that face. After cropping, we compress it to 48*48 gray scale so that it can be used as input of our model.

Figure 7 shows that the true labels and the prediction results of the faces of some famous memes. We can see that the model tends to make the mistakes that predict the the other faces to emotion neutral.

6 Conclusions

We developed a CNN for a facial expression recognition problem and evaluated their performances using different post-processing and visualization techniques.

The results demonstrated that deep CNNs are capable of learning facial characteristics and improving facial emotion detection. The accuracy of top 20 rankings of this competition on Kaggle range from 53% to 71%, while human accuracy on the FER-2013 is $65 \pm 5\%$. We managed 58%. Notable findings:

- Only **Happy (0.84)** and **Surprise (0.71)** were very accurately guessed
- **Sad** was frequently overpredicted, especially when the true label was **Angry (0.26)**, **Fear (0.29)** and **Neutral (0.23)**
- **Fear (0.24)** was our least successful category
- **Disgust** was frequently predicted as **Angry (0.27)**, but not vice-versa **(0.02)**
- The relative accuracy of **Happy** could be useful in marketing settings and in reward-based behavioral manipulation.
- The low accuracy of **Angry** and the very low accuracy of **Fear**, and especially the misprediction of Fear **as** Angry **(0.14)** could be troubling in law enforcement contexts.
- The overall accuracy seems high enough to be used in limited psychotherapeutic contexts.