Convolutional Neural Network for Facial Expression Recognition

Liyuan Zheng
Department of Electrical Engineering
University of Washington
liyuanz8@uw.edu

Shifeng Zhu
Department of Electrical Engineering
University of Washington
sfzhu@uw.edu

Abstract

In this project, we developed Convolutional Neural Networks (CNN) for a facial expression recognition task. Our goal is to classify each facial image into one of the seven facial emotion categories. Different depth structures of CNN models are trained using gray-scale images from the Kaggle facial expression challenge [1]. To reduce over-fitting, we utilized different techniques including dropout and $L2$ regularization.

1 Introduction

In recent years, due to the fast growing social networks, photos and videos that include faces constitute a great proportion of visual data over the Internet, using machine to do the facial recognition is becoming commonly used. Facial expression, as the most expressive and direct way to communicate emotion in humans, draws a lot of attractions. However, although facial expressions can be easily recognized by human beings, reliable facial expression recognition by machine is still a great challenge. Moreover, it is an interesting and challenging problem due to its wide range of applications such as human–computer interaction and data-driven animation. Therefore, there has been substantial research interest in computer vision systems to recognize facial expression. The recent success of Convolutional Neural Networks (CNN) in tasks such as object classification extends to the problem of facial expression recognition. In this project, we would like to present an approach based on CNN for facial expression recognition. The objective of this project is to classify images of human faces into discrete expression categories and compare the result of different CNN layer structures.

2 Related Work

In the past decades, research on the facial expression recognition has significantly advanced with the help of computer vision techniques. Many approaches to this problems were raised, including using pyramid histograms of gradients (PHOG) [2], AU aware facial features [3], boosted LBP descriptors [4] , and RNNs [5]. Recently, due to data source available and computation power increasing, the machine learning technique of neural networks has seen resurgence in popularity in the field, and much of the recent achievements in face recognition have been due to the deep Convolutional Neural Networks (CNN).

By carefully designing the local and global features and training through convolution, pooling and layered architecture, it shows that CNN is a very strong tool for expression recognition. One of the first papers to apply neural nets to Facial Expression Recognition [6]. In this paper it performed Gabor filtering on the raw images followed by various transformations and PCA before applying a 3 layer neural net.

30th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain.

In Yu and Zhang recent work of EmotiW 2015 [7], CNN has yielded excellent performance. In their paper, they have used an ensemble of CNNs with five convolutional layers each. They have achieved excellent recognition performance and proposed a data perturbation and voting method that further increases the recognition performance.

Mollahosseini et al. have also achieved state of the art results in facial expression recognition [8], their network consisted of two convolutional layers, max-pooling, and 4 inception layers. Inception layer structure were used in their paper, which gains significantly on local feature performance and also enhance the facial expression recognition performance.

In Deepali Aneja's work [9], they proposed a novel expression transfer approach from humans to multiple stylized characters. They first train two Convolutional Neural Networks to recognize the expression of humans and stylized characters independently and then utilize a transfer learning technique to learn the mapping from humans to characters to create a shared embedding feature space. They retrieved characters using perceptual model mapping and human geometry.

3 Data

The dataset for this project is from Kaggle facial expression challenge [1], which is comprised of 48 $\times$ 48 pixel gray-scale images of human faces. The training set consists of 28,709 examples, while both the test and validation sets are composed of 3,589 examples. Each image is categorized into one of the seven classes that express different facial emotions: "anger", "disgust", "fear", "happiness", "sadness", "surprise", and "neutral".

Figure 1: Facial expression figures

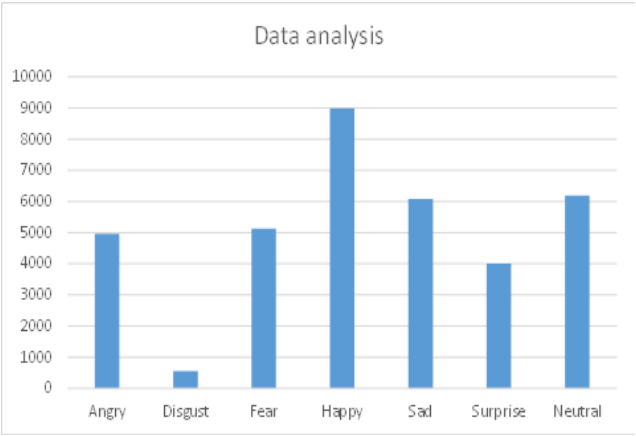

Figure 2: The statistics of the data

Figure 1 depicts one example for the facial expression categories and as illustrated in Figure 1, the data set's images vary considerably in scale, viewpoint, and illumination. We note that all of the images are preprocessed as they are mostly centered and adjusted so that the face occupies about the same amount of space in each image.

The classes distribution of the dataset can be found in Figure 2. As we can see, the "happy" class has the most training data set and the number of "disgust" is far less than other 6 emotions.

4 Experiment

We developed CNNs with variable depths to evaluate the performance of these models for facial expression recognition. A typical CNN architecture contains all or some of the following layer types:

[Conv(ReLU) $\rightarrow$ Max-pooling] with Dropout$\times$ M $\rightarrow$ [Fully-connected(ReLU)]$\times$ N $\rightarrow$ Softmax.

The first part of the network refers to the first kind of layers, of which usually contains Convolutional layer with ReLU activation function and Max-pooling layer. We can include spatial max-pooling, dropout and even batch normalization and ReLU nonlinearity in the first step. After using M times of the first layers, the network is led to Fully-Connected layers that always have affine operation

and ReLU nonlinearity, and can also include batch normalization and dropout. Finally, the network is followed by the affine layer connects to the class nodes, in which the scores are computed and then softmax loss function is used to calculate the probability. This CNN model gives us a lot of freedom to customize the number of Convolutional and Fully-Connected layers, as well as the existence of dropout and batch normalization method. Furthermore, the number of filters, size of filters, strides, and zero-padding can be specified by users. Along with dropout techniques, we included $L2$ regularization in our implementation.

We implemented the aforementioned models from scratch, and then using TensorFlow and Keras to build deeper CNN . Due to limitation of our computer CPU, we explored only certain layers of the model.

4.1 Shallow CNN

As a starting point, we built a three-layer Convolutional Neural Network containing

5×5 Conv(ReLU) $\rightarrow$ 2×2 Max-Pooling $\rightarrow$ Fully-Connected(ReLU) $\rightarrow$ Softmax,

which is shown below in Fig 3.

In this model, we are using 32 filters for the Conv layer, 512 hidden nodes are used in the Fully connected layer, the learning rate is 0.0001 with a decay rate of 1×10^{-6}. We used Adam to update the weight.

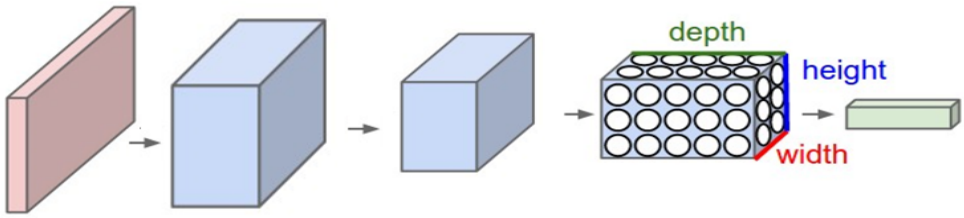

Figure 3: CNN model

Before we go into the full data set, checks on the code should be carried out first. Our first sanity check is the initial loss of small set of data. Since there are 7 classes in total, we are expecting a softmax error around $\ln 7 = 1.946$. The second check is training on a fraction of the training data to make sure our model work properly. After passing above tests, full data set is used to train the model and its result is shown in Fig.4 and Fig.5.

4.2 6-layers CNN

The second step is using TensorFlow to build a 6-layers model. In this model, we have

5×5 Conv(ReLU) $\rightarrow$ 2×2 Max-Pooling $\rightarrow$ 5×5 Conv(ReLU) $\rightarrow$ 2×2 Max-Pooling $\rightarrow$ Fully-Connected(ReLU) $\rightarrow$ Softmax.

In this model, we are using 32 filters for the first Conv layer, 64 filters for the second Conv layer, 512 hidden nodes are used in the Fully connected layer, the learning rate is 0.0001 with a decay rate of 1×10^{-6}, the whole model has a regularization constant 0.001. We used Adam to update the weight.

4.3 Deeper CNN

Finally we used Keras to develop a deeper CNN model ,in which we have used dropout layers.

3×3 Conv(ReLU) - 2×2 Max-Pooling with dropout rate of 0.25 $\rightarrow$ 3×3 Conv(ReLU) - 2×2 Max-Pooling with dropout rate of 0.25 $\rightarrow$ 3×3 Conv(ReLU) - 2×2 Max-Pooling with dropout rate of 0.25 $\rightarrow$ Fully-Connected(ReLU) with dropout rate of 0.5 $\rightarrow$ Softmax.

In this model, we are using 64 filters for the first Conv layer, 128 filters for the second Conv layer and 256 filters for the third Conv layer, with dropout rate 0.25. 1024 hidden nodes are used in the Fully connected layer with dropout rate 0.5, the learning rate is 0.0001 with a decay rate of 1×10^{-6}, the whole model has a regularization constant 0.001. We used Adam to update the weight.

5 Result

5.1 Shallow CNN

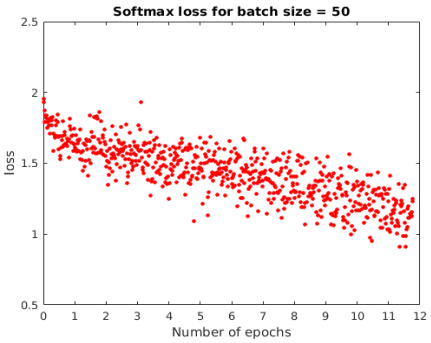

Figure 4: Loss of shallow CNN

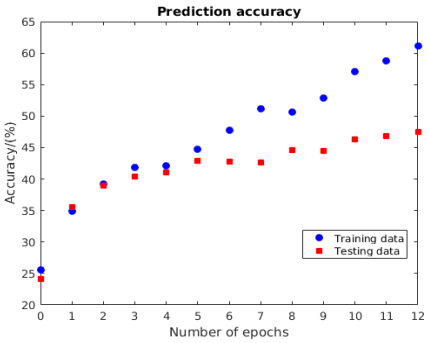

Figure 5: Loss of shallow CNN

	Anger	Disgust	Fear	Happy	Sad	Surprise	Neutral
Anger	0.34420	0.00204	0.08758	0.16497	0.19959	0.05295	0.14868
Disgust	0.23636	0.12727	0.05455	0.20000	0.10909	0.09091	0.18182
Fear	0.15530	0.00189	0.20265	0.13826	0.23485	0.14015	0.12689
Happy	0.05461	0.00000	0.03072	0.71559	0.09443	0.03868	0.06598
Sad	0.10606	0.00337	0.09764	0.18013	0.37710	0.04040	0.19529
Surprise	0.05048	0.00000	0.06010	0.07933	0.06490	0.68029	0.06490
Neutral	0.09744	0.00000	0.03994	0.18051	0.17252	0.05591	0.45367

Table 1: Confusion matrix for shallow CNN model

An example plot of the shallow CNN accuracy evolution can be found in Fig 5. As it can be seen, the training accuracy increases while the test accuracy remains almost constant after about 10 epochs. This means that we can overfit our data very easily after around 10 epochs. The final accuracy only is about 48% for the shallow CNN model.

The confusion matrix on the validation set is shown in Table 1. As it can clearly be seen, Disgust is the class where our network works the worst, and the emotion Happy is most successful, which is very likely due to the uneven class distribution.

5.2 Comparison of other CNNs

	Anger	Disgust	Fear	Happy	Sad	Surprise	Neutral
Anger	0.5631	0.	0.05995	0.07922	0.1456	0.02997	0.1220
Disgust	0.2678	0.4464	0.1071	0.01785	0.08928	0	0.07142
Fear	0.1048	0.004032	0.3790	0.08266	0.2157	0.09274	0.1210
Happy	0.03910	0.	0.01787	0.8190	0.04022	0.02011	0.06369
Sad	0.1301	0.001531	0.07963	0.09341	0.4885	0.01378	0.1929
Surprise	0.04096	0.	0.05542	0.04819	0.02891	0.7759	0.05060
Neutral	0.09390	0.001647	0.05601	0.1285	0.1383	0.01153	0.5700

Table 2: Confusion matrix for deeper CNN model

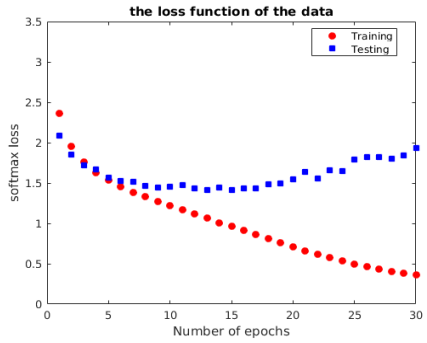

Figure 6: Loss of 6 layer CNN

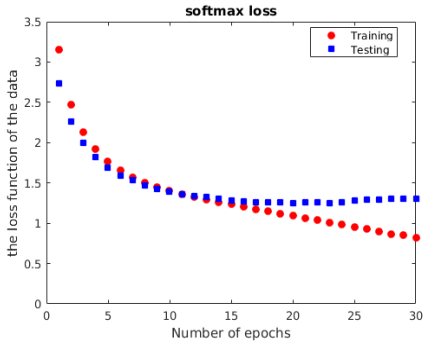

Figure 7: Loss of deeper CNN

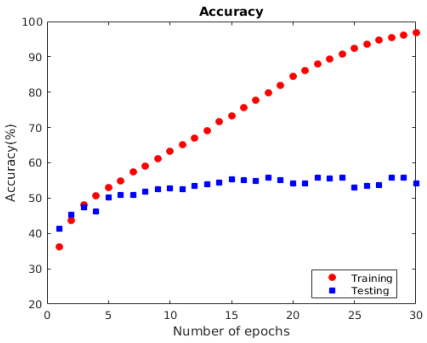

Figure 8: Loss of 6 layer CNN

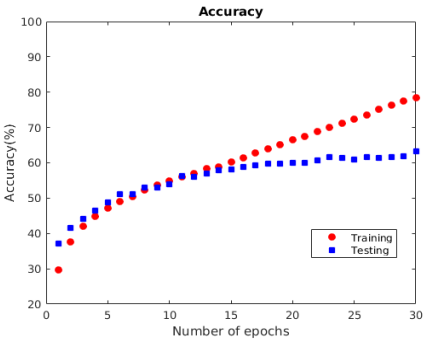

Figure 9: Loss of deeper CNN

To compare the performance of the shallow model with other models, we plotted the loss history and the obtained accuracy in these models. As showed by Fig 6 and 7, we are definitely overfitting 6-layer CNN but we tend to do better on deeper CNN by adding more non-linearity and hierarchical usage of anti-overfitting techniques such as dropout and $L2$ regularization.

From Fig 8 and 9, by comparing 6-layer network to shallow network, we can find that adding one more Conv(ReLU) and Max-pooling layer could help increasing performance. We can also see that the deeper network being used, the slower it converge as there are more weight to update. The final testing accuracy we obtained are 52.72% and 61.19% for 6-layer CNN and deeper CNN respectively. The deeper network achieves much higher accuracy as it contains not only one more Conv(ReLU) and Max-pooling layer but also Dropout layer that prevents overfitting.

We have computed the confusion matrix in Table 2 for the deeper networks. As demonstrated in these figures, the deep network results in higher true predictions for most of the labels than shallow network.

6 Conclusion

In this project, we built several Convolutional Neural Network to recognize facial expression from gray-scale picturess inspired by Kaggle facial expression recognition challenge [1]. We used different depth of Conv(ReLU) - Max-Pooling architecture and Softmax loss. For regularization, we experimented with $L2$ regularization and dropout in our final deeper CNN. To optimize the net, we used Adam with learning rate decay.

The final testing accuracy we obtained using this architecture was 61.19%. Some of the difficulties with improving this is that the images are very small and in some cases it is very hard to distinguish which expression is on each image, even for humans. Due to limitation of our lack of GPU, we have only explore certain architecture but we believe that adding more layers and more filters would further improve the network.

7 Future Work

In this data set, some preliminary work has been done already as the faces are focused into the image center. In our own code, we did a subtraction of the mean image for every single image before further analysis, but it's not enough. Since the nature of face, we can have asymmetry, either because of the pose, orientation or some minor imperfection. To extend our data, we can carry out a horizontal flip transformation, or rotate the image data in a small range of angle, shown in Fig 10.

Figure 10: left-most: original image; middle: image after horizontal flip; right-most: small angle rotation

What's more, we can modify our model by adding more weights to those important parts, such eyes, eyebrows or the corner of the lips, since these parts play important roles to tell what the emotion is. Another direction is to extend it into real time emotion detection, which can be a very tough but practical problem.

References

[1] Ian Goodfellow, Dumitru Erhan, and etc. Challenges in representation learning: A report on three machine learning contests, 2013.

[2] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[3] Anbang Yao, Junchao Shao, Ningning Ma, and Yurong Chen. Capturing au-aware facial features and their latent relations for emotion recognition in the wild. In *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*, pages 451–458. ACM, 2015.

[4] Caifeng Shan, Shaogang Gong, and Peter W McOwan. Facial expression recognition based on local binary patterns: A comprehensive study. *Image and Vision Computing*, 27(6):803–816, 2009.

[5] Samira Ebrahimi Kahou, Vincent Michalski, Kishore Konda, Roland Memisevic, and Christopher Pal. Recurrent neural networks for emotion recognition in video. In *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*, pages 467–474. ACM, 2015.

[6] Matthew N Dailey, Garrison W Cottrell, Curtis Padgett, and Ralph Adolphs. Empath: A neural network that categorizes facial expressions. *Journal of cognitive neuroscience*, 14(8):1158–1173, 2002.

[7] Zhiding Yu and Cha Zhang. Image based static facial expression recognition with multiple deep network learning. In *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*, pages 435–442. ACM, 2015.

[8] Ali Mollahosseini, David Chan, and Mohammad H. Mahoor. Going deeper in facial expression recognition using deep neural networks. *CoRR*, abs/1511.04110, 2015.

[9] Deepali Aneja, Alex Colburn, Gary Faigin, Linda Shapiro, and Barbara Mones. Modeling stylized character expressions via deep learning. In *Asian Conference on Computer Vision*, pages 136–153. Springer, 2016.