LFW Faces in the Wild
Face Recognition with CNN

Julio Gonzalez, Ting Lu, & Kalea Sebesta

***Abstract*- This project was conducted to become familiar with building Convolutional Neural Networks (CNN) for image classification. Neural networks work by simulating that of brain neurons in a computer. The algorithms then learn by recognizing patterns. This project focuses on small neural network problems where the models only need to recognize a limited number of known people. This type of artificial intelligence is at the fore front of technology and such examples include face recognition glasses for Alzheimers patients and home surveillance systems. Models were created to compare accuracy and identify the parameters that would lead to the optimal classification result on a small network.**

I. Introduction

THIS project uses the LFW: Labeled Faces in the Wild dataset which has 13233 images of 5749 people, where 1680 people have two or more images in the dataset. The focus of this project is to construct a small neural network; a network that will recognize 10 or fewer faces with a high level of accuracy.

A. Problem Definition

This project focuses on small neural network problems where the models only need to recognize a limited number of known people. Two such scenarios where a small neural network would be employed include face recognition glasses and home surveillance systems. Creating glasses that would hold face recognition software on them that could identify loved ones information, could be leveraged for Alzheimers patients when they come in contact with their loved ones. Home surveillance systems could use this type of network to train and recognize 5 to 10 faces of those living in the house. Then, when an unrecognizable face is seen in the home it could alarm the system.

B. Methods & Experiments

1) Process: For this project we leveraged Keras and TensorFlow to create a CNN, and Tensorboard to display accuracy and loss of both the training and test set. Since the problem deals with a small network we limited our data to only include images of people whose faces were seen frequently. We created a bash script that went through the CNN only fetching faces of people who were seen at a minimum 2 to 70 times in the dataset (this was done in intervals of 1). The bash script also ran on 20 and 50 epochs, and batch sizes of 64, 128, and 256. It is important to note that the date was retrieved using sklearn dataset lfw. The sklearn library was leveraged often throughout this project for data formatting and classification reporting.

Before building the CNN there was preprocessing of the data which occurred. This included formatting and defining the variables, converting images to grey scale, and splitting the data into its respective training and testing X and y components. From there the labeled components of the data needed to be encoded. Following that the data was reshaped. This included converting each 28 x 28 image of the train and test set into a matrix of size h x w x 1 which later was fed into the network. In order for the neural network to handle this data it first needed to be normalized. To accomplish this the data was converted to a float32 data type and the pixel values were rescaled to be in the inclusive range of 0 to 1.

Once the preprocessing of the data was finished it was fed into the neural network building function that was created. This function took in batch size, number of epochs, number of classes, X and y training and test encoded components, and the checkpoint parameters. The checkpoint parameters are essentially where the best weight results are stored and is what the new results are compared to. Lastly, the accuracy and loss values that were calculated with the predefined functions were displayed. In order to utilize the TensorBoard functionality, the script was passed parser arguments which created logs for the various runs of parameters which then could simultaneously display the accuracy and loss results on the TensorBoard dashboard.

2) CNN Structure: The structure of the CNN model can be seen below in Fig. 1. It is comprised of three convolution layers in addition to three max-pooling layers. The first layer has 32 3x3 filters, the second layer has 64 3x3 filters and the third layer has 128 3x3 filters. Each of the max pooling layers are of size 2x2. In order to reduce overfitting the technique of dropout regularization was utilized. The first and second iteration randomly ignored 25% of the neurons in the model, the third ignored 40% and the last iteration ignored 30%. To optimize the prediction we employed the cross-entropy loss function and compiled the model by using the Adam optimizer for the gradient descent.

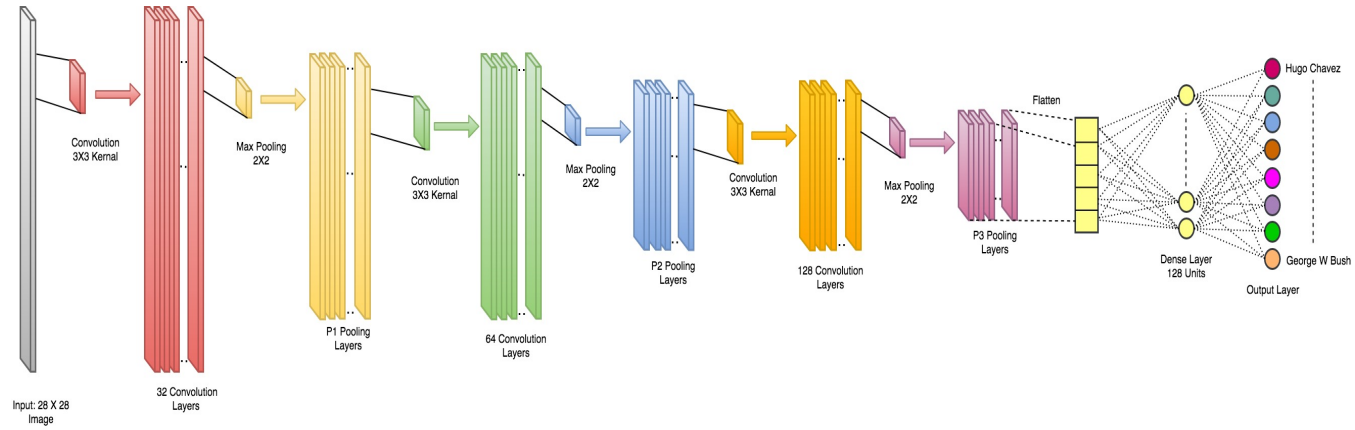


Fig. 1. CNN Model Structure: Output from model run with 70 minimum faces, a batch size of 128, and 50 epochs.

C. Results

Below Fig. 2 shows the TensorBoard display of the model run with various parameters. For the purpose of comparison it is displaying results when the minimum faces seen in the dataset is 10, 20, 30, 40, 50, 60, and 70. Other parameters such as batch size and number of epochs are held the same for these displays. It is seen that with a minimum number of faces being 70 the test accuracy rate is roughly 90%. It is important to note that the difference between accuracy rates when the minimum faces is between 30 and 70 is only 9%. 30 minimum faces resulted in a large amount of classes (taking it outside the bounds of the small network this project aims to work with) thus it was dropped from future model comparisons. In attempts to increase accuracy rate, data image generator was used to augment data. For this particular network the addition of augmented data lead to an overfitting to occur on the training set and thus the accuracy of the test set was lower than expected. Ultimately it did not prove to be an improvement over the original model with the original dataset.

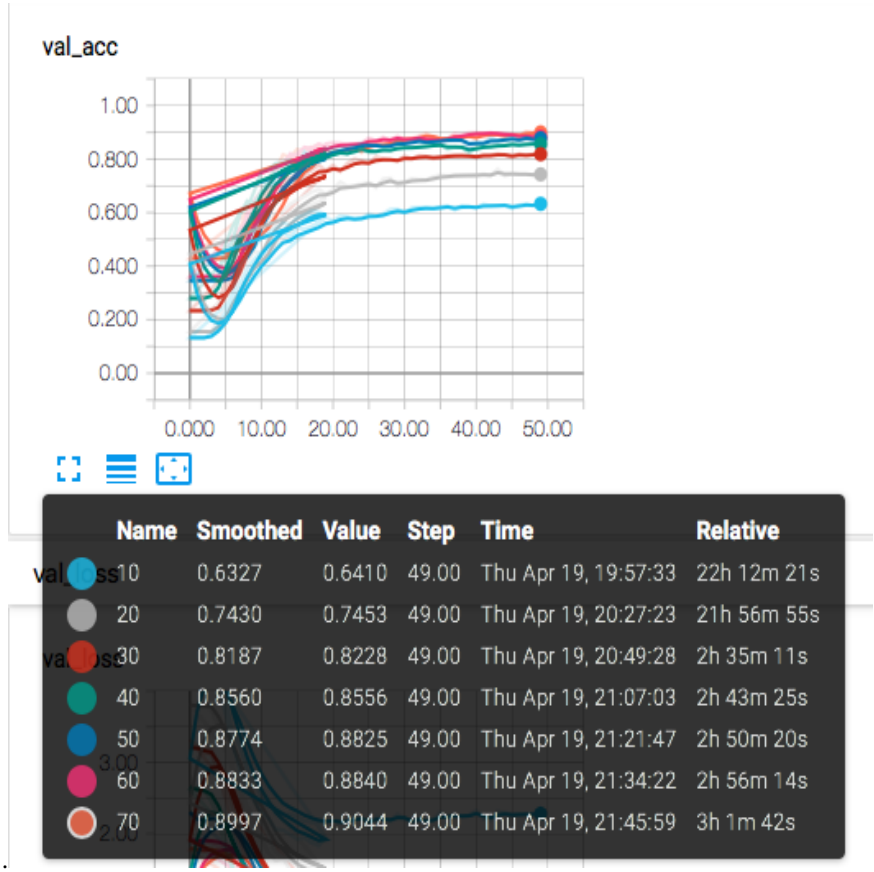


Fig. 2. TensorBoard Test Accuracy Model Output. This figure displays the results for when the minimum faces ranges between 10 and 70 with 10 face increments.

Due to the closeness of the accuracy rates, the following four models were ran and the classification reports are below. Again, we held the number of epochs constant at 50 and the batch size at 128 to truly see the effects of manipulating the minimum number of faces.

Table I displays the classification results for the model run with 40 minimum faces, a batch size of 128, and 50 epochs. For this particular model the test loss was 0.7593, test accuracy was 0.8307. It also found 451 correct labels and 110 incorrect labels.

TABLE I
MIN-FACES=40 BATCH-SIZE=128 EPOCH=50

Class	Precision	Recall	F1-Score
Ariel Sharon	0.32	0.76	0.45
Arnold Schwarzenegger	0.71	0.33	0.45
Colin Powell	0.90	0.91	0.90
Donald Rumsfeld	0.89	0.78	0.83
George W Bush	0.93	0.89	0.91
Gerhard Schroeder	0.71	0.76	0.74
Gloria Macapagal Arroyo	1.00	0.80	0.89
Hugo Chavez	0.82	0.70	0.76
Jacques Chirac	0.58	0.54	0.56
Jean Chretien	0.86	0.75	0.80
Jennifer Capriati	1.00	0.55	0.71
John Ashcroft	0.73	0.62	0.67
Junichiro Koizumi	0.88	0.88	0.88
Laura Bush	0.77	0.91	0.83
Lleyton Hewitt	0.79	0.85	0.81
Luiz Inacio Lula da Silva	0.80	0.75	0.77
Serena Williams	0.92	0.69	0.79
Tony Blair	0.79	0.87	0.82
Vladimir Putin	0.50	0.64	0.56
Average	0.84	0.80	0.81

Table II displays the classification results for the model run with 50 minimum faces, a batch size of 128, and 50 epochs. For this particular model the test loss was 0.4742, test accuracy was 0.8739. It also found 406 correct labels and 62 incorrect labels.

TABLE II
MIN-FACES=50 BATCH-SIZE=128 EPOCHS=50

Class	Precision	Recall	F1-Score
Ariel Sharon	0.46	076	0.57
Colin Powell	0.95	0.95	0.95
Donald Rumsfeld	0.74	0.81	0.78
George W Bush	0.97	0.88	0.92
Gerhard Schroeder	0.86	0.80	0.83
Hugo Chavez	0.94	0.71	0.81
Jacques Chirac	0.93	0.82	0.87
Jean Chretien	0.75	0.88	0.81
John Ashcroft	0.85	0.89	0.87
Junichiro Koizumi	0.85	0.85	0.85
Serena Williams	0.73	0.85	0.79
Tony Blair	0.89	0.91	0.90
Average/Total	0.89	0.87	0.87

Table III displays the classification results for the model run with 60 minimum faces, a batch size of 128, and 50 epochs. For this particular model the test loss was 0.5045, test accuracy was 0.8840. It also found 355 correct labels and 50 incorrect labels.

TABLE III
MIN-FACES=60 BATCH-SIZE=128 EPOCHS=50

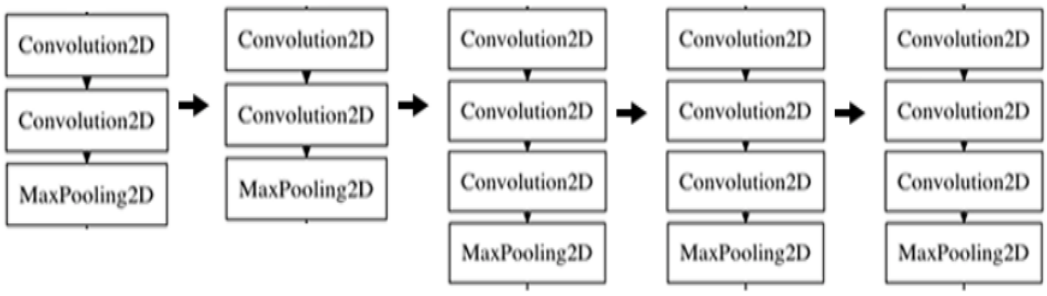
Class	Precision	Recall	F1-Score
Ariel Sharon	0.45	0.76	0.57
Colin Powell	0.96	0.93	0.95
Donald Rumsfeld	0.73	0.92	0.81
George W Bush	0.97	0.88	0.92
Gerhard Schroeder	0.82	0.82	0.82
Hugo Chavez	1.00	0.70	0.83
Junichiro Koizumi	0.80	1.00	0.89
Tony Blair	0.88	0.86	0.87
Average/Total	0.90	0.88	0.88

Table IV displays the classification results for the model run with 70 minimum faces, a batch size of 128, and 50 epochs. For this particular model the test loss was 0.4485, test accuracy was 0.8915. It also found 342 correct labels and 45 incorrect labels. Table IV is also the model for which the CNN structure above in Fig. 1 illustrated.

TABLE IV
MIN-FACES=70 BATCH-SIZE=128 EPOCHS=50

Class	Precision	Recall	F1-Score
Ariel Sharon	0.47	0.83	0.60
Colin Powell	0.90	0.93	0.91
Donald Rumsfeld	0.90	0.76	0.83
George W Bush	0.95	0.96	0.95
Gerhard Schroeder	0.86	0.77	0.81
Hugo Chavez	0.88	0.79	0.83
Tony Blair	0.93	0.78	0.85
Average/Total	0.90	0.88	0.89

D. VGG-Face Methods & Results



.

VGG-FACE model was developed by Parkhi and colleagues (2015). The model was trained over 2 million images and 2600 people, which was a significantly larger dataset than the LFW dataset (13,233 images with 5,749 people). Because of the similarity of the tasks (i.e. face recognition) and differences in sample size, we froze the convolution layers of the VGG model and only trained one fully connected layer in the end. Use of pre-trained model had improved our validation accuracy with fewer minimum faces.

TABLE V
VGG-FACE RESULTS

Min Faces	Train Accuracy	Validation Accuracy
2	0.93	0.70
5	0.87	0.89
10	0.99	0.95
15	1.00	0.97

II. Conclusion

Looking at the classification reports for the various models it is concluded for the purpose of solving a small network, the best model has the following parameters: minimum faces of 50, batch size 128, and epochs of 50. This model gives an accuracy of 0.87 and classifies 12 faces. Although minimum faces of 60 and 70 have a slightly higher accuracy of 0.88 they only classify 7 and 8 faces. It is important to note that the limitation of the project and thus the restriction of the model is the minimum number of faces. Although the VGG seems to be a better solution to this limitation. Future steps for this project could include but are not limited to: modifying the structure of the CNN (adding additional layers), creating and applying triplet loss to the VGG16. It would also be interesting to revisit the augmented data with a more complex CNN structure. This project allowed us to become familiar with building CNN's and understanding the process of layering, filtering, dropout, optimizing, and compiling a model recursively to find the parameters and weights which produce the optimal results for a small face recognition network.

Appendix A
Server IP and Code

Server located at IP: http://129.114.108.20:8888/ From the home tab navigate to code by: [home] → [code] → [Trials]

References

[1] LFW: Labeled Faces in the Wild
http://vis-www.cs.umass.edu/lfw/

[2] TensorFlow & TensorBoard Tutorials:
https://www.tensorflow.org/programmers_guide/summaries_and_tensorboard

[3] Keras Tutorial
https://keras.io

[4] G. Hu, Y. Yang, D. Yi, J. Kittler, W. Christmas, S. Li, and T. Hospedales *When Face Recognition Meets with Deep Learning: an Evaluation of Convolutional Neural Networks for Face Recognition*, ICCV workshop paper, 2015.

[5] D. Yi, Z. Lei, S. Liao, and S. Li *Learning Face Representation from Scratch*, CoRR, vol: abs/1411.7923, 2014.