Face Verification Final Project

Christopher Marley, University of Washington (Seattle)

EE 443: Machine Learning for Signal Processing Applications

Professor Jenq-Neng Hwang

June 6, 2023

Table of Contents:

Abstract

This paper explores the feasability of using a pre-trained model on new data for face verification, especially when the new data is of a limited quantity. We conclude that there is marginal improvement between the validation and testing data, likely because the network has learned to overfit to its limited training data.

Introduction

In an increasingly automated society, the ability to verify the identity of humans becomes immensely critical. Automation plays a role in many human-independent tasks already, for example: factory production, refrigeration processes, braking and steering control, traffic control, data research, and numerical analysis. Face verification is the next step in automating society with regards to controlling access of humans, and for the increased surveillance of our everyday lives. We could permit only those registered in an apartment complex to be able to open the gate and require them to be present to admit guests for the increased security of the complex. We could monitor the locations of people and make conclusions about the normalcy of their actions based on their activity patterns. At a high level, we could install them on highway systems to create an ultimate "Internet of things," where our vehicle registration, personal identity, and banking information are all intimately connected for the purpose of paying tolls. All these possible advances require the development of face verification systems that are quick, simple, and precise.

This project does not aim to make any breakthroughs or fancy discoveries. It is a reapplication of pretrained convolutional neural networks on new facial data. This report will discuss the methods researched and used to evaluate new data, analyze the results of a retrained system, and make conclusions about the accuracy of the system as evaluated by human error.

Methods

This project employs a “Siamese network” architecture, in which the same model is used on two inputs, and a second separate model takes the outputs of the original models and compares them against a known truth. This way, the backpropogation affects only the main network, and does not induce a network to make oscillating changes on two separate branches. Furthermore, the program takes advantage only of the validation data as both training and validation, rather than using a separate database of images.

The program is divided into functional components, with the model and all following evaluation code being global. This is because I discovered the ReLU function used by Tensorflow has a memory leak, resulting in the program surpassing the 12 GB memory limit after 6 runs. It is an external garbage collection issue, and the solution was simply to create a singleton rather than recreate the model for each pair of images to verify. After an initial testing of the validation data, it was trained in batches of 64 for 20 epochs to avoid overfitting.

The Siamese network itself defines the loss by how far the Euclidean distance between embedded facial vectors differs from 0 and 1, the measure of whether the images match or not. In hindsight, I notice that I do not normalize the vectors, but the use of a softmax activation layer keeps the output coordinates in the range [0, 1]. Lastly, the verification function uses a cosine metric instead of a Euclidean metric, which may be invalid for this case because the magnitude of a component indicates the strength of a facial feature as defined by the CNN.

Summary of verification: cosine similarity measure with a threshold of 0.4

Finally, I manually labeled the test data to get an idea of how the computer performs in alignment with human error.

Results

Prior to training, the network had an 85% accuracy on the validation data. After retraining the final layer of the model, the network had a 98% accuracy. The two pairs of images that the network got wrong are presented in Table 1.

Comparing the program’s labeling of the test images with my labeling, the program appeared to have 86.75% accuracy, but I will admit freely that my face recognition skills are very poor, as is evident by my failure to recognize classmates from even a year ago.

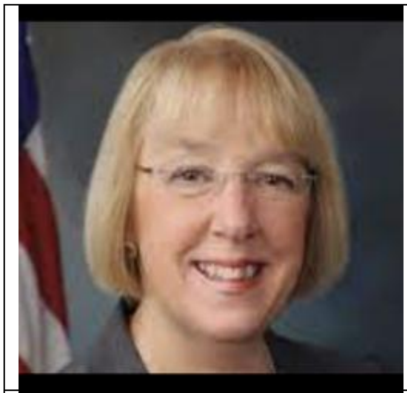
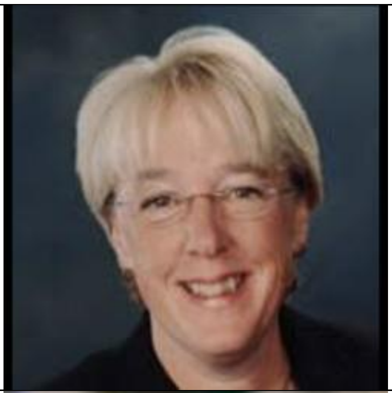
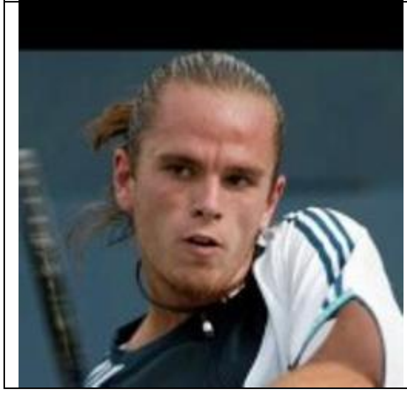
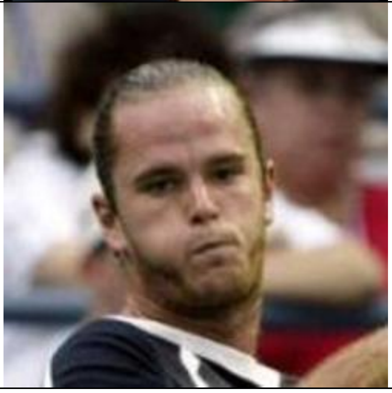
		Expected: False Actual: True
		Expected: True Actual: False

Table 1 – Incorrect answers reported after weight retraining

Also presented is a scatterplot of the training loss and validation loss with respect to epoch. The training loss is plotted in red, and the validation loss in blue.

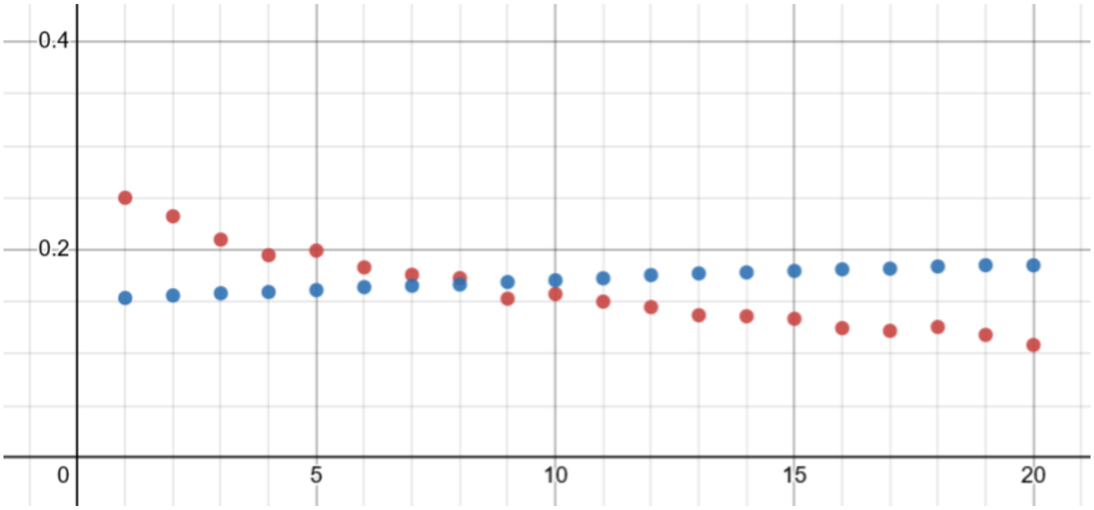


Table 2 – Training loss (red) and validation loss (blue)

Conclusion

The model classifies images accurately with respect to human labeling at a rate higher than the validation data it was trained on. However, the increasing validation loss indicates that the model was learning the data rather than the task of image verification. Nonetheless, we can conclude that the model adapted to a new set of data, albeit marginally, and had we used a larger sample set to train the network, it may have performed more accurately.

The code and results can be found in the following GitHub repository:

https://github.com/cmarley3-14/EE443-Final-Project