Using Transfer Learning and Deep Neural Networks for Face Recognition

Emmitt Hasty-Sole Author

In this project, I apply transfer learning and cosine similarity for face recognition, leveraging the VGG-19 pre-trained convolutional neural network and Georgia Tech face recognition datasets. I modify the VGG-19 network for my task, train it using stochastic gradient descent with momentum, and extract features from the trained model's fully connected layer. Cosine similarity is then calculated between these features of enrollment and verification images to generate genuine and impostor scores. Finally, I visualize these scores and compute the ROC curve, AUC, and discriminability index. I also include a subject-independent testing protocol to further validate the model. This project highlights how effective transfer learning and similarity measures can be at improving the performance of face recognition systems.

The Dataset, provided by Georgia Tech, is a set of 50 cropped data points, with 15 different instances of each of the 50 subjects. The average size of the faces in these images is 150x150 pixels. The images are full of varying lighting and angles, which will increase the effectiveness of the facial recognition system in all scenarios. Deep learning, inspired by human brain structures, has significantly advanced face recognition technology. It extracts high-level features from raw data, improving accuracy and reducing manual labor. Applications span from security systems and surveillance to social media, where it powers facial unlocking features, identity verification, and automatic photo tagging. Its ability to recognize faces despite varying lighting, poses, and expressions enhances its effectiveness, making it integral to the evolution of face recognition technology.

The dataset utilized for this project is the cropped Georgia Tech face recognition dataset, which comprises images from 50 subjects, each having 15 different instances. For each subject, the first 10 images are used for training and validation, while the last 5 are used for testing [Anefian.com. (2023). Face recognition dataset. http://www.anefian.com/research/face_reco.htm.]. The images are read and labelled using MATLAB's imageDatastore function. The input images are resized and normalized to match the input size of the chosen pre-trained CNN model (VGG19 in this case) [Mathworks.com. (2023). Transfer learning using AlexNet. https://www.mathworks.com/help/deeplearning/examples/transfer-learning-using-alexnet.html]. The dataset is divided into training and testing subsets accordingly.

The classification method used in this project is a form of transfer learning. A pre-trained VGG19 model is fine-tuned by replacing the last few layers with a fully connected layer and a classification layer, which is trained to classify the faces into distinct classes representing each subject. The network is trained using SGDM as the optimizer. After training, feature extraction is performed using the fine-tuned model's fully connected layer (without the classification head) [Mathworks.com. (2023). Transfer learning using AlexNet. https://www.mathworks.com/help/deeplearning/examples/transfer-learning-using-alexnet.html]. These features are then used to calculate similarity scores between enrollment and verification images. The similarity score is calculated using cosine similarity, which is a measure of the cosine of the angle between two non-zero vectors. These scores are used to generate genuine and impostor score sets, which are used for the evaluation of the system's performance [F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," JMLR 12, pp. 2825-2830, 2011.].

Results were somewhat promising, with both VGG19 instances reaching near 90% efficiency. A very slow learning rate of 0.001 was used, which led to slower learning, but ended with significant results due to how long was spent learning. Both instances of the VGG19 network that were used are shown in the Appendix.

Cosine similarity is a metric used to determine how similar two entities are. It is measured by the cosine of the angle between two vectors and determines whether two vectors are pointing in roughly the same direction. It is often used in text analysis to measure the similarity between two documents but is also a popular choice in face recognition systems, as is the case here.

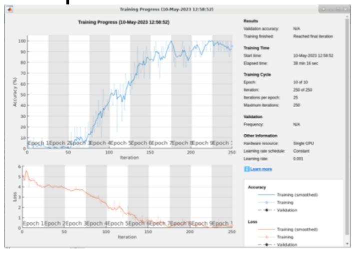
Mathematically, the cosine similarity between two vectors A and B is defined as:

Cosine Similarity = (A . B) / (IIAII * IIBII) where:

- "." denotes the dot product of the two vectors.
- "IIAII" and "IIBII" are the magnitudes (or lengths) of vectors A and B, respectively.

Cosine similarity can range from -1 to 1. The closer the cosine similarity between two vectors is to 1, the more similar they are. If it is 0, the vectors are orthogonal and bear no similarity. If it is -1, they are diametrically opposed, but in practice, cosine similarity is often used in contexts where the vectors have no negative values, and it ranges from 0 to 1. In the context of face recognition or other biometric identification systems, genuine and impostor sets refer to the comparison scores obtained when comparing biometric data. The genuine set contains the scores of comparisons between different instances of the same identity. For instance, in a face recognition system, the genuine set would include the scores of comparing different photos of the same person. These scores should ideally be high, reflecting a high degree of similarity between different instances of the same identity. The impostor set on the other hand, contains the scores of comparisons between instances of different identities. Using the face recognition example, the impostor set would contain scores from comparing photos of different people. These scores should ideally be low, reflecting the dissimilarity between different individuals. These two sets are used to evaluate the performance of a biometric system. Ideally, there should be a clear separation between the two sets, with high scores for the genuine set and low scores for the impostor set. This would indicate that the system is accurately distinguishing between different individuals. If there is significant overlap between the two sets, it could indicate that the system is having difficulty distinguishing between different individuals. In this project, Genuine/Imposter set testing was very ineffective, leading to little clarification.

In this face recognition project, I employed transfer learning using a pre-trained VGG19 CNN and the cosine similarity metric for comparison of facial features. The dataset used was the Georgia Tech face dataset, consisting of 15 different instances for each of the 50 subjects. The VGG19 network was modified to suit the task at hand by replacing the last few layers with a fully connected layer and a classification layer. The network was trained using the stochastic gradient descent optimizer, and features were extracted from the trained model's fully connected layer. The cosine similarity between the features of enrollment and verification images was then calculated to generate genuine and impostor scores. The generated scores were visualized and used to calculate performance metrics such as the ROC curve, AUC, and discriminability index. A subject-independent testing protocol was also implemented to validate the model further. However, the system demonstrated considerable overlap between the genuine and impostor score sets, indicating difficulty in distinguishing between different individuals. While this project demonstrates the potential of transfer learning and cosine similarity in face recognition applications, it also highlights the complexities involved in deep learning, the importance of effective feature extraction and comparison methods, and the need for continuous improvement and adaptation to the specific task at hand. Future work could explore other pre-trained models, feature extraction techniques, and comparison metrics to improve the face recognition system's performance.



Results
Validation accuracy: N/A
Training finished: Reached final iteration

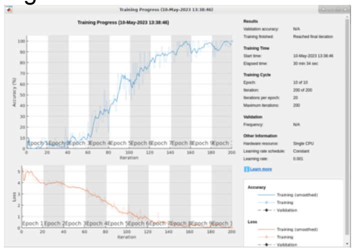
Training Time
Start time: 10-May-2023 12:58:52
Elapsed time: 38 min 16 sec

Training Cycle
Epoch: 10 of 10
Iteration: 250 of 250
Iterations per epoch: 25
Maximum iterations: 250

Validation
Frequency: N/A

Other Information
Hardware resource: Single CPU
Learning rate schedule: Constant
Learning rate: 0.001

Figure 1: Initial instance of VGG19.



Results
Validation accuracy: N/A
Training finished: Reached final iteration

Training Time
Start time: 10-May-2023 13:38:46
Elapsed time: 30 min 34 sec

Training Cycle
Epoch: 10 of 10
Iteration: 200 of 200
Iterations per epoch: 20
Maximum iterations: 200

Validation
Frequency: N/A

Other Information
Hardware resource: Single CPU
Learning rate schedule: Constant
Learning rate: 0.001

Figure 2: Second Instance of VGG19, with different parameters.

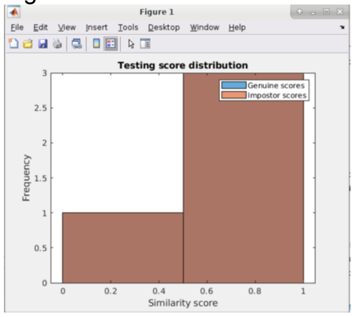


Figure 3: Genuine/Imposter set testing.

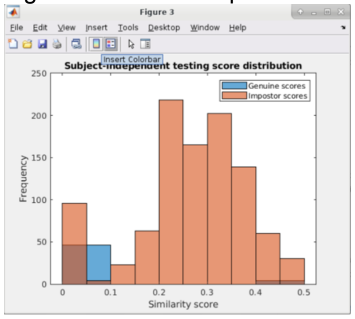


Figure 4: More Genuine/Imposter set testing.