HW4 DeepLearning

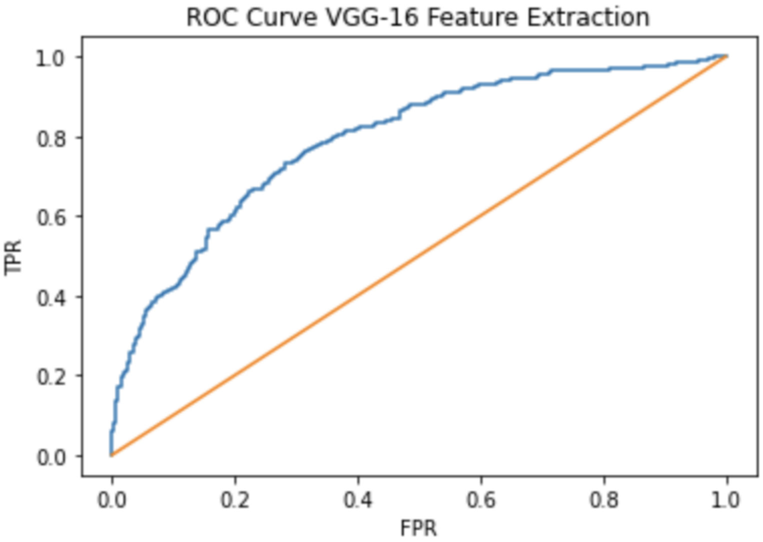
Vedant Jain

April 2021

1 Part 2: Models, VGG-16

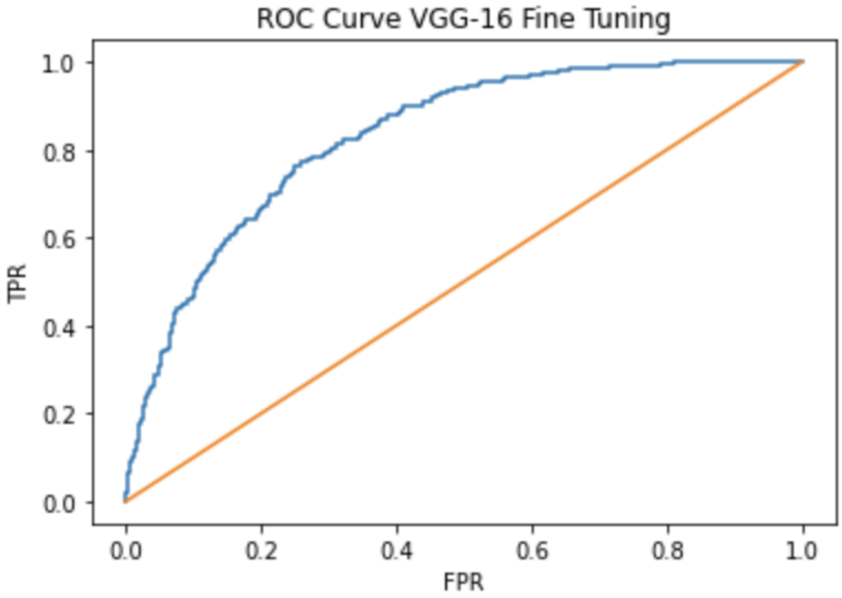
In the VGG models for training I used a binary cross entropy loss function with learning rate of 10^{-5} using the Adam optimizer. I used the cosine similarity measurement to compute the similarity between model outputs for the image pairs.

2.1 ROC curve for feature extraction from VGG-16



In this section I took a pretrained VGG-16 network and removed the last layer of it. Then ran the lfw pairs through it and used the cosine angle similarity to determine the similarity values for each pair. This is highlighted in the ROC curve above. The AUC for this was: .7914

2.2 ROC curve for fine tuning from VGG-16

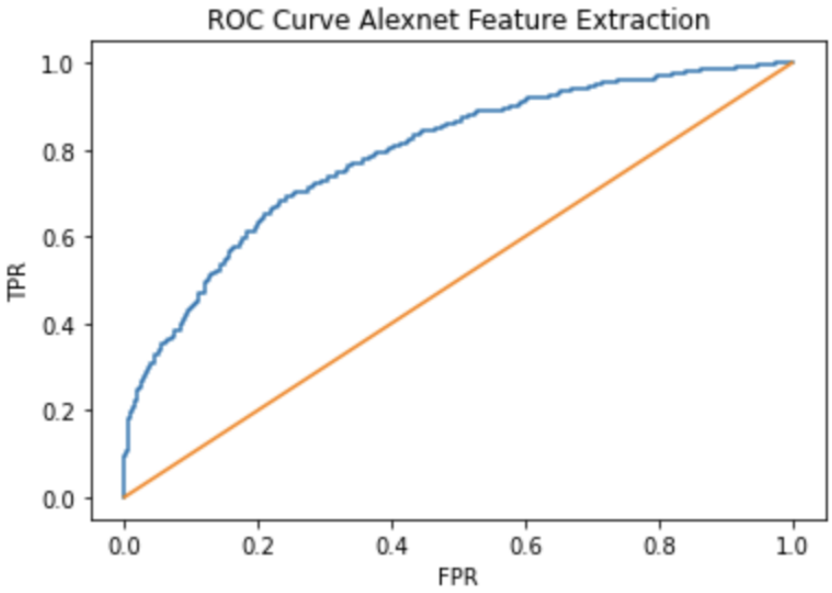


In this section I performed fine tuning of the VGG network, this involved loading the pretrained weights of the model removing the last layer of the model and then training on the dataset for 10 epochs. Training was fed in batches of 60 pairs. The AUC was:.83 , I experienced better performance on the fine-tuned model especially after increasing it up to 10 epochs.

2 Part 2: Models, Alexnet

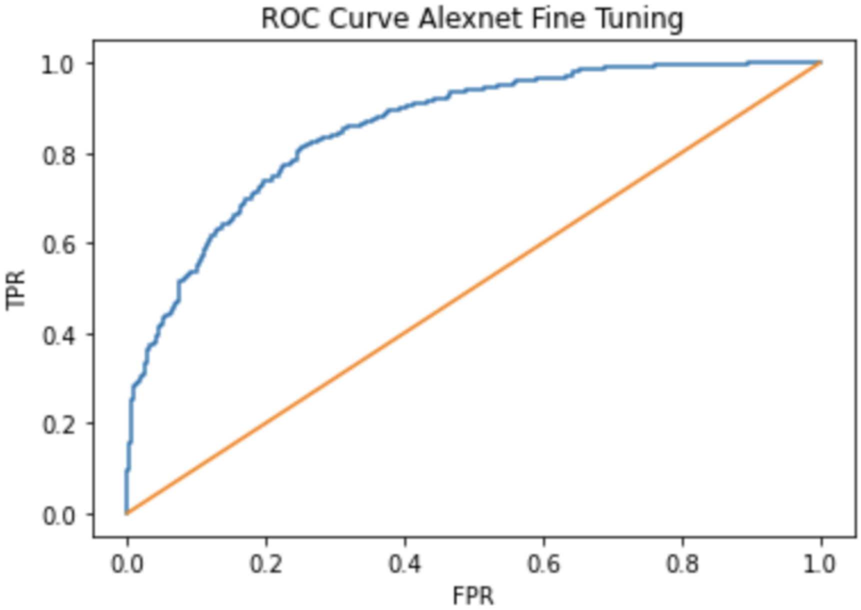
In the Alexnet models I used a binary cross entropy loss function with learning rate of 10^{-4} using the Adam optimizer. I used the cosine similarity measurement to compute the similarity between model outputs for the image pairs.

2.3 Feature Extraction from Alexnet



In this section I took a pretrained Alexnet network and removed the last layer of it. Then ran the lfw pairs through it and used the cosine similarity to determine the similarity values for each pair. This was used in a binary cross entropy loss function as described earlier. This is highlighted in the ROC curve above. The AUC for this was: .79.

2.4 Fine Tuning Alexnet



In this section I performed fine tuning of the Alexnet model, this involved loading the pre-trained weights of the model and then training on the dataset for only 2 epochs. Training was fed in batches of 40 pairs. The AUC was: .85 , I experienced better performance on the fine tuned model and it was interesting to see that smaller epochs gave better results. This probably helped keep the model from over fitting. **This model was my greatest performing one.**

3 Appended Code and References to code

Here is all the code for my work. I wrote everything in python using the framework pytorch on google COLABs Using the GPU setting for faster running. Data was imported using the sklearn libraries and was verified to resemble the lfw dataset.