COMP 551 Project 3

Yifei Chen, Yuyan Chen, Ing Tian

November 27, 2021

Abstract

This project aims at predicting handwritten English letters and digits in an image. In this study, we have improved the performance progressively from the baseline VGG16 model. Concretely, we have experimented with multiple variants of VGG16 and two preprocessing techniques, namely data augmentation, and noise reduction. Also, we have tried to utilize the unlabeled training set with Expectation-Maximization (EM) algorithm. In the end, we have reached a validation accuracy of 97.1%.

1 Introduction

It is known that convolutional neural networks have been successful with character recognition. In this study, we first experimented with three neural networks, namely Tutorial CNN, AlexNet, and VGG16, and improved the best model's (VGG16) performance progressively. VGG16 is a renowned model designed for a much more complex task, with 1000 output labels and 224x224 input size. Hence, we hypothesized that the original VGG16 is not the best fit for this mission, and we have altered its structure to gain better performance. On top of that, we have recognized overfitting with all of our models. To reduce variances, we tried to increase the kernel size of MaxPooling layers and introduce more training samples via data augmentation. Both attempts are beneficial. Besides, we tried to utilize the unlabeled data with the EM algorithm, yet it brought slight improvement. The final validation accuracy is 97.1%

2 Datasets

The Combo MNIST dataset consists of 30,000 labeled images and 30,000 unlabeled images, each containing one English letter and one digit, both handwritten. We selected 1000 images from the labeled training set for validation. We explored two data preprocessing techniques: noise reduction and data augmentation. In terms of noise reduction, all pixels with an intensity greater than 100 were assigned an intensity of 255, and the rest was made 0 (Figure 7 and Figure 8). We created extra samples for data augmentation by rotating each image 15 and 30 degrees clockwise and counter-clockwise (Figure 9 and Figure 10).

3 Experiments and Results

In this study, we improved the performance of our model progressively. Furthermore, we conducted all of our experiments on a labeled training set except for EM, where we tried to utilize the unlabeled dataset.

We first built three neural networks: one inspired from AlexNet [1], one taken from the tutorial, and one mimicking the famous VGG16 [2]. VGG16 has the highest training and validation accuracy (99.9% and 93.8% respectively), and thus we have set the vanilla VGG16 as the baseline model (Figure 11, Figure 12, Table 1).

Then, we explored the effects of noise reduction with VGG16. As depicted in Figure 14, noise reduction hurt the model's performance, with the maximum validation accuracy decreased to 83.2%. Hence, we did not apply noise reduction for future experiments.

We then tried to improve our model's performance by adjusting its design and tuning its hyperparameters. More specifically, we have explored 5 variants of the vanilla VGG16 model (Figure 1) with SGD. First, we halved the width of VGG16's convolutional layers (Figure 2). Secondly, we added 3 more convolutional layers to VGG16 (Figure 3). Thirdly, we deleted three convolutional layers from VGG16 (Figure 4). For the fourth variant, we tried to increase the kernel size of the *MaxPooling* layers based on the third variant (Figure 5). Last, we replaced *ReLU* with *LeakyReLU* for the fourth variant and denoted the last variant as "VGG-HML" (Figure 6). All variants' trends of training and validation accuracy are shown in Figure 15, Figure 16, and Figure 17. Their maximum training and validation accuracy are listed in Table 2. VGG-HML performs the best, and we have selected this model for following experiments.

To utilize our unlabeled data, we explored the EM algorithm on VGG-HML, whose pseudocode is presented in Algorithm 1. The performance of the model from each iteration of the EM algorithm is recorded in Figure 18, Figure 19, and Table 3. Notice that the EM algorithm is timely expensive with over 20 minutes per iteration, and thus we have only included 10,000 unlabeled samples instead of the entire 30,000 entries.

Finally, we explored data augmentation with VGG-HML, which improved the validation accuracy to 96.2%. Furthermore, we conducted the same experiment with Adam gradient descent instead of the standard SGD and increased the validation accuracy to 97.1%.

4 Discussion and Conclusion

Data Augmentation and Noise Reduction

All of our VGG variants, including VGG-HML, are overfitted to the training set, with approximately 100% training accuracy and less than 96% validation accuracy. To reduce models' variances, we decided to add more training samples via rotating the original images, as rotational invariance is desired for image classification models[4]. Since many letters and digits in the original images are slightly rotated, we expected that rotating our images with less than 30 degrees does not produce misleading samples. As shown by our experiments, data augmentation has successfully increased our VGG-HML's validation accuracy to 96.2% with SGD and to 97.1% with Adam. This result has confirmed our hypothesis that more training samples can reduce variances.

Another commonly-used data augmentation technique is noise injection[3], which introduces a regularization effect and thus improves the robustness of the model. Furthermore, as the training samples vary due to noise, the model is better at generalization. Conversely, noise reduction harms the model's performance, as shown in Figure 14.

Model Tuning

Besides data augmentation, another approach to combat high variances is to have a bigger max-pooling kernel. In our experiments, we have increased the kernel size from 2x2 to 3x3. According to our observations, this move has increased the validation accuracy from 94.1% to 94.8%, as recorded in Table 2. This result coincides with the conclusions from this study[1].

With the ReLU activation function, all neurons with a negative value shall receive a gradient of 0. Using LeakyReLU tolerates negative values and thus makes the model more robust. The former hypothesis was confirmed by our empirical evidence as shown in Table 2. Using LeakyReLU has further increased the validation accuracy from 94.8% to 95.9%.

VGG16 is a complex model designed for large images (224x224), and it is known that overly expressive models may hinder its performance on a simpler dataset [5]. To confirm our hypothesis, we have deleted 3 convolutional layers from VGG16 as in "VGG Delete Tail" and appended 3 more convolutional layers as in "VGG Deeper." As in Table 2, "VGG Delete Tail" outperforms the original VGG16, whereas "VGG Deeper" scores lower than the original. Besides, a thinner model (VGG Half-Width) does not yield a positive result. Again, these figures have demonstrated that the design of neural networks relies on empirical evidence and is more of a heuristic approach.

EM algorithm

The EM algorithm does not yield a noticeable improvement (Table 3). It is possible that we have not included enough unlabeled data(10,000 samples). We might gain better performance by incorporating the entire unlabeled dataset.

Statement of Contributions

Ing Tian - Writing Codes
Yuyan Chen - Experiment Design
Yifei Chen - Report Writeup

References

[1] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." Advances in neural information processing systems 25 (2012): 1097-1105.

[2] Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." arXiv preprint arXiv:1409.1556 (2014).

[3] Shorten, Connor, and Taghi M. Khoshgoftaar. "A survey on image data augmentation for deep learning." Journal of Big Data 6.1 (2019): 1-48.

[4] Quiroga, Facundo, et al. "Revisiting data augmentation for rotational invariance in convolutional neural networks." International Conference on Modelling and Simulation in Management Sciences. Springer, Cham, 2018.

[5] Ravanbakhsh, Siamak, "Some Important Concepts", COMP551, 21 Sep. 2021, McGill University, Montreal

Appendix A

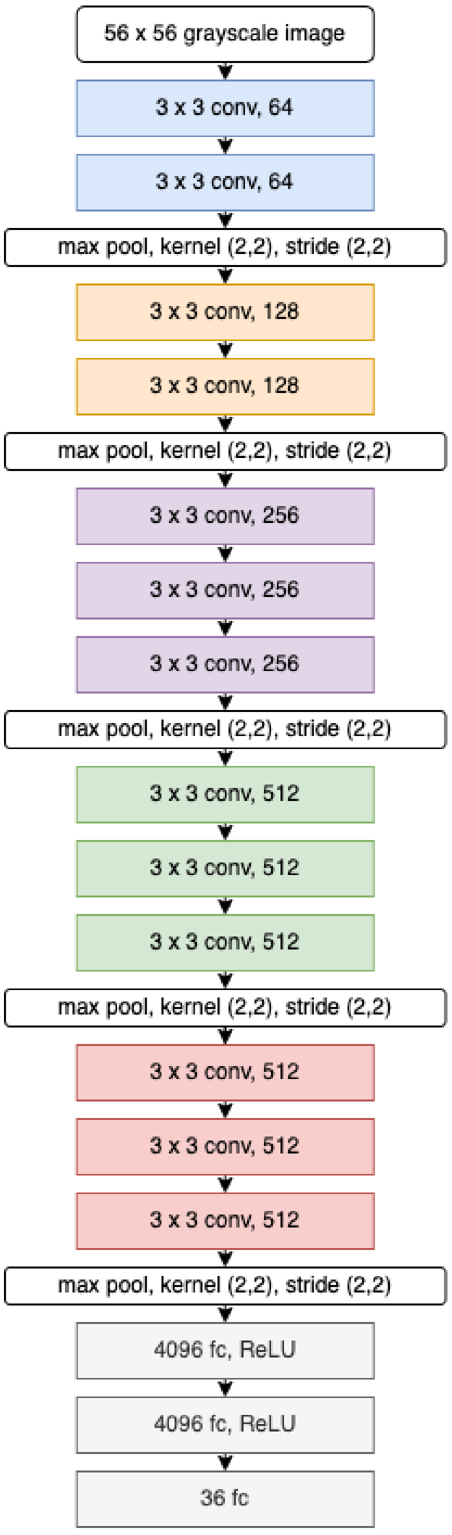

Figure 1: VGG16 Vanilla

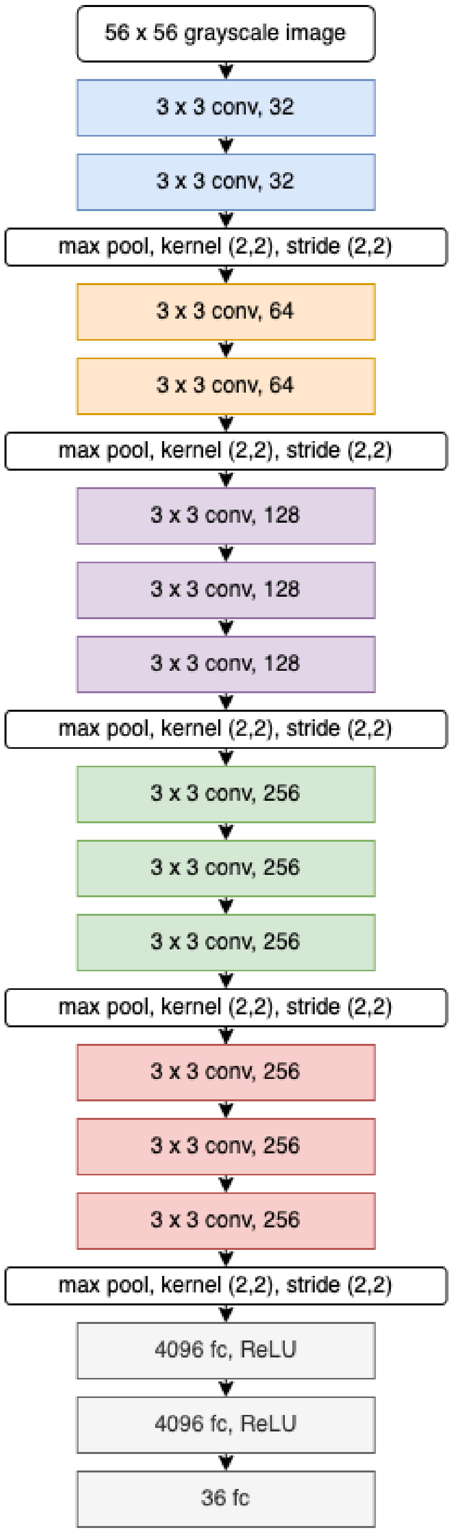

Figure 2: VGG16 Half Width

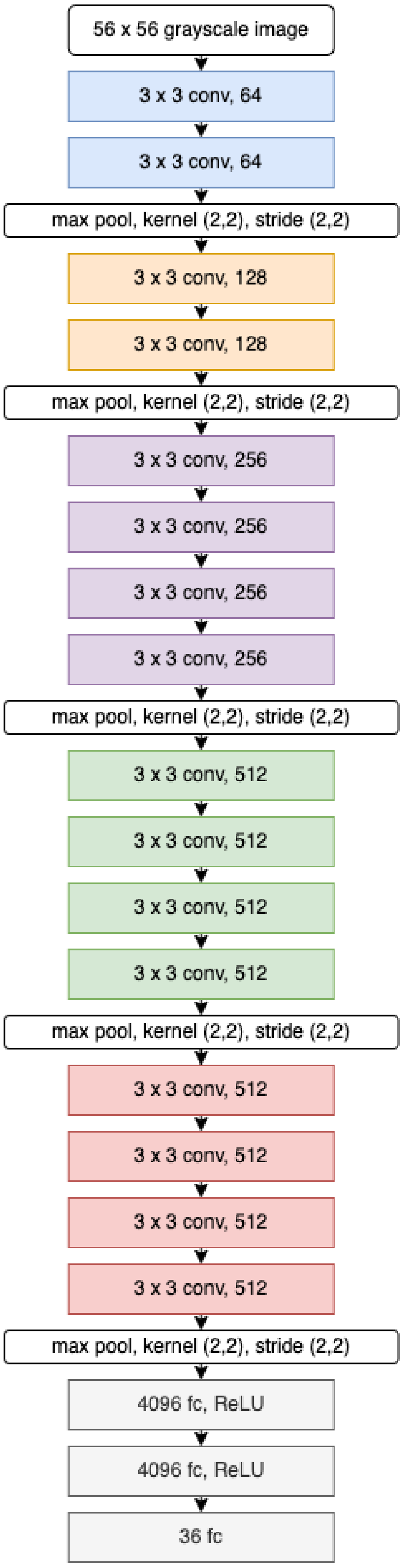

Figure 3: VGG16 Deeper

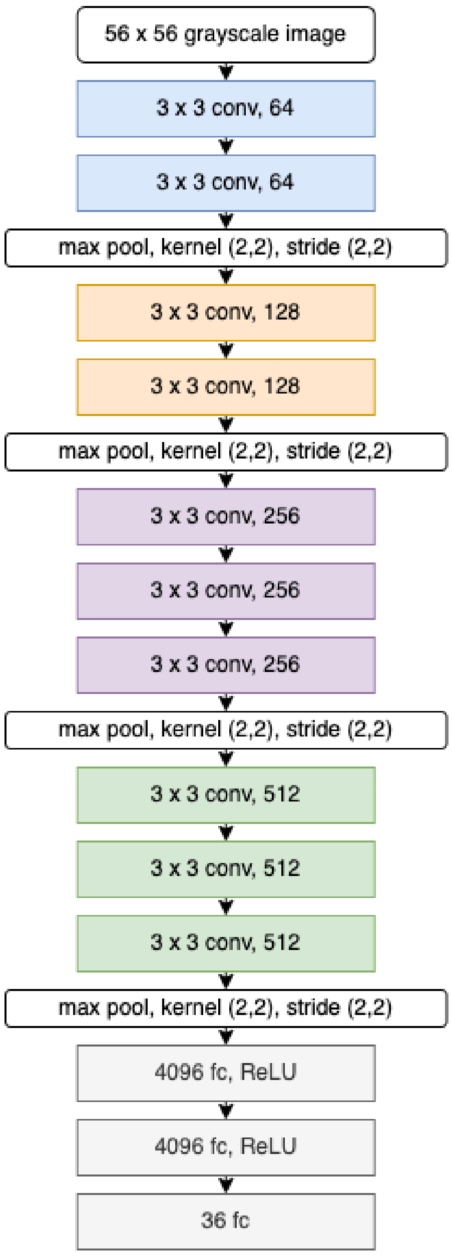

Figure 4: VGG16 Delete Tail

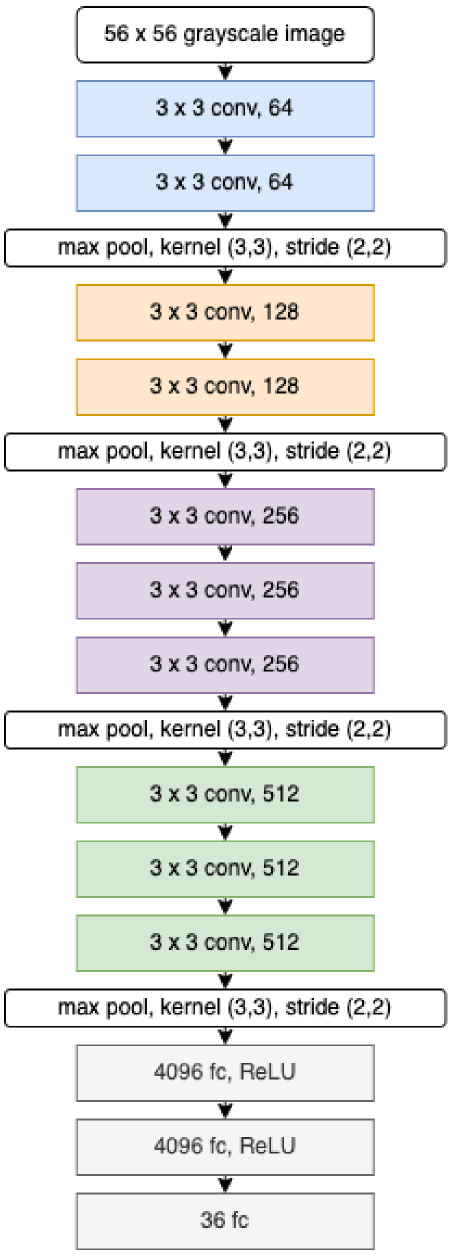

Figure 5: VGG16 Higher MaxPooling

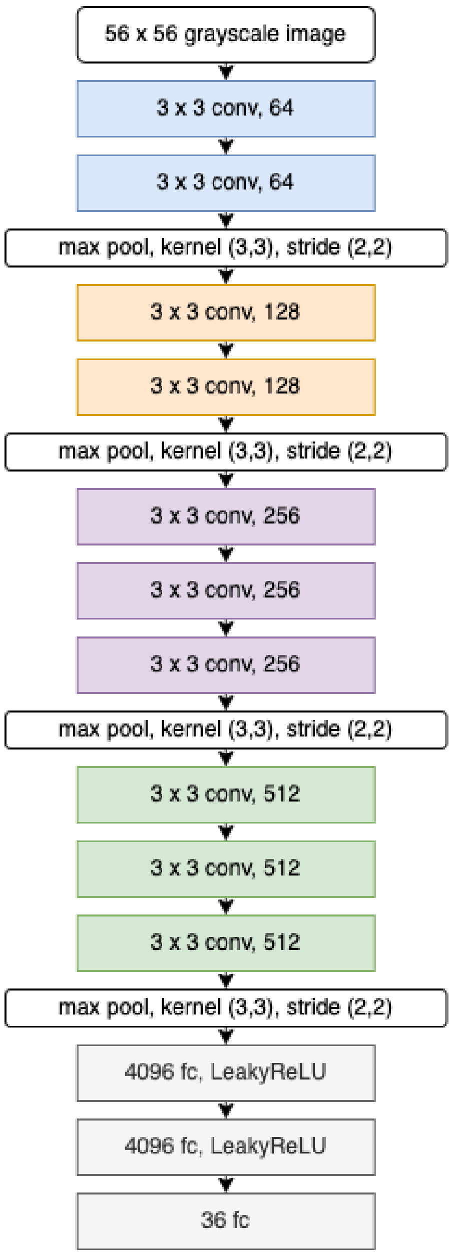

Figure 6: VGG16 Higher MaxPooling Leaky

Appendix B

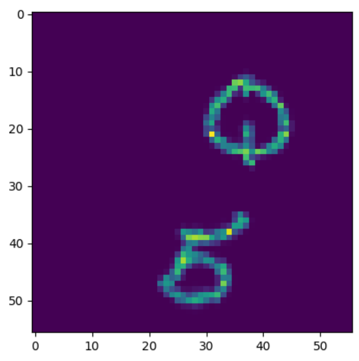

Figure 7: Original Image

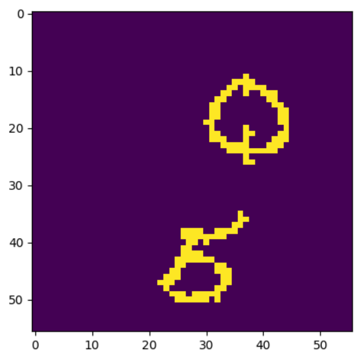

Figure 8: Denoised Image

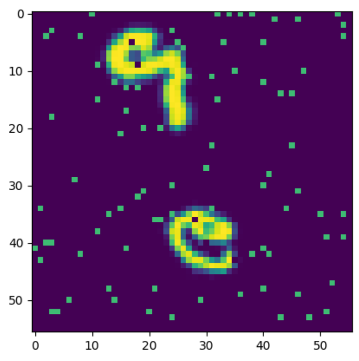

Figure 9: Original Image

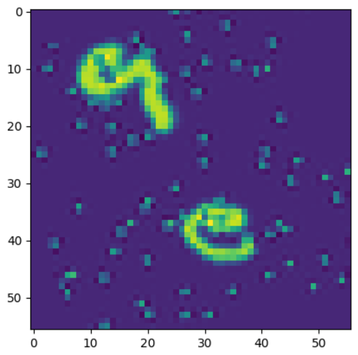

Figure 10: Rotated Image (15 degrees)

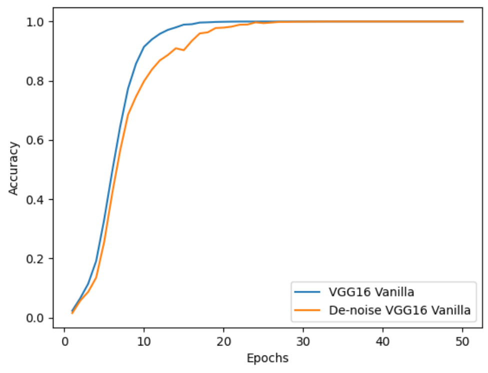

Figure 13: Training Accuracy of VGG16 trained with original and denoised images

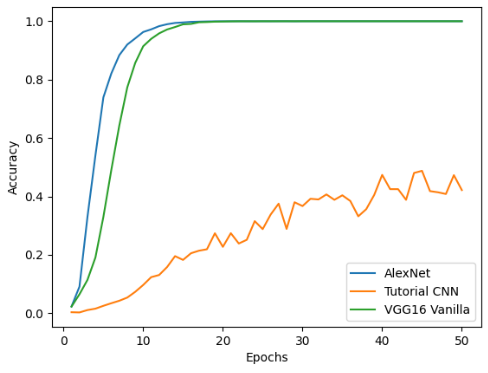

Figure 11: Training Accuracy of Tutorial CNN, AlexNet, and VGG16

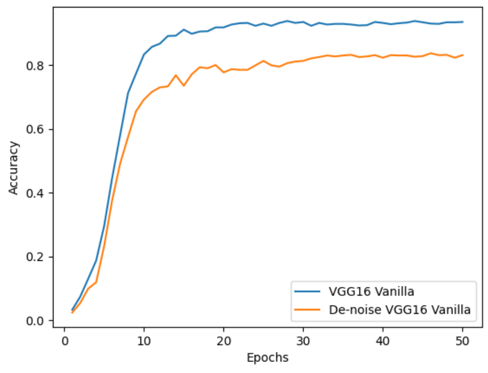

Figure 14: Validation Accuracy of VGG16 trained with original and denoised images

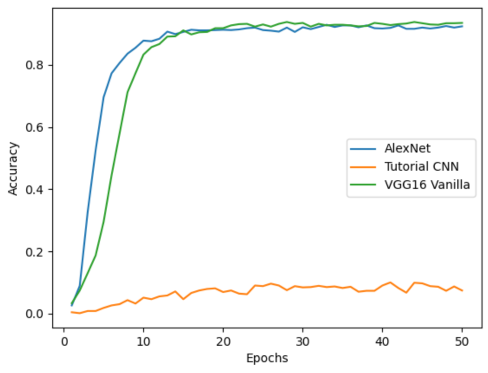

Figure 12: Validation Accuracy of Tutorial CNN, AlexNet, and VGG16

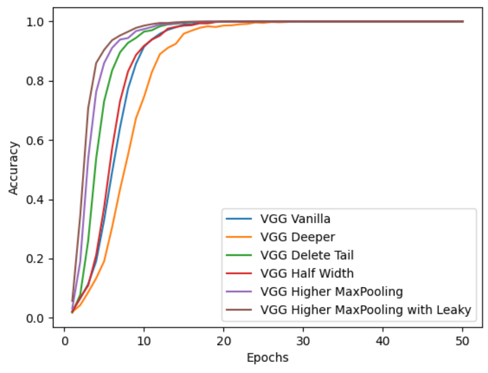

Figure 15: Training Accuracy of VGG16 variants

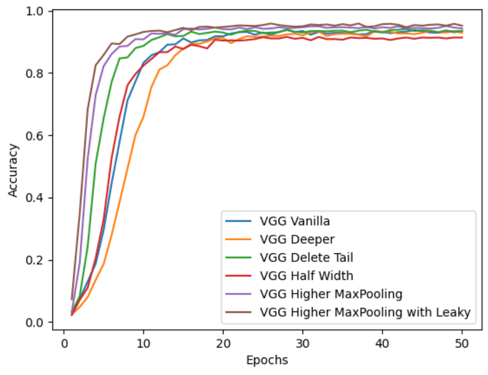

Figure 16: Validation Accuracy of VGG16 variants

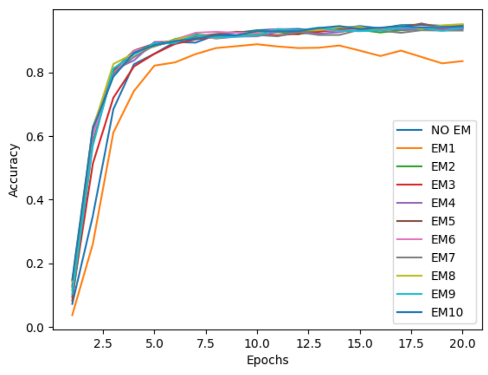

Figure 19: Validation Accuracy of VGG-HML for each EM Iteration

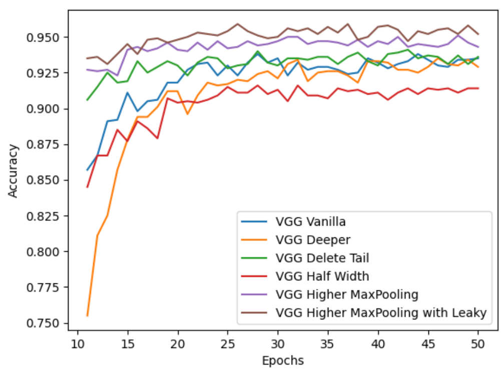

Figure 17: Validation Accuracy of VGG16 variants from epoch 10 to epoch 50

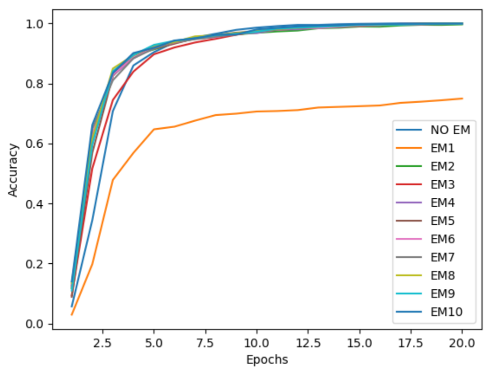

Figure 18: Training Accuracy of VGG-HML for each EM Iteration

Appendix C

Table 1: Highest Training and Validation Accuracy for Tutorial CNN, AlexNet, and VGG16

Model	Training	Validation
CNN	48.8	9.1
AlexNet	100.0	92.7
VGG16	100.0	93.8

Table 2: Highest Training and Validation Accuracy of VGG16 Variants

Model	Training	Validation
Vanilla	100.0	93.8
Deeper	100.0	93.5
Delete Tail	100.0	94.1
Half Width	100.0	91.4
Higher Max Pooling	100.0	94.8
HML	100.0	95.9

Table 3: Highest Training and Validation Accuracy of Each EM Iteration

Iteration	Training	Validation
1	74.9	88.8
2	99.7	93.8
3	99.9	94.1
4	99.8	93.9
5	99.9	95.3
6	99.9	94.2
7	99.9	93.4
8	99.9	95.1
9	99.9	94.0
10	99.9	94.5

Algorithm 1 EM Algorithm

```
Assign random labels for unlabled images
while $i \leq 10$ do
    Train models for 50 epochs
    Load the model with the lowest validation accuracy
    Use this model to predict labels for unlabeld images
    Update labels in the training set
end while
```