Network Pruning on Pretrained AlexNet Model and SFEW Dataset with Different Number of Finetuned Layers

Abhijit Adhikary

Research School of Computer Science,
Australian National University
U7035746@anu.edu.au

Abstract. Transfer learning provide a robust starting point for training Neural Networks where the target has very few samples. As most Deep Convolutional Neural Network's initial layers learn similar patterns, these can be used for a wide variety of applications. But the number of layers to fine tune when using a pretrained model on a different dataset should be chosen carefully. In this paper we observe the effect of fine tuning different number of layers using transfer learning using the AlexNet model and report results on the SFEW face emotion dataset. Furthermore, we analyze the effect of pruning hidden neurons based on vector angles on the AlexNet model trained using transfer learning.

Keywords: Network Reduction, Pruning, AlexNet, Transfer Learning, Finetune.

1 Introduction

Artificial Neural Networks (ANN) have successfully been used to solve complex problems where a hand-crafted solution is hard to obtain. They are universal function approximators which can learn from data and can gain insights about their distribution. Although ANNs are being for about half a century, it saw a breakthrough during the 2012 ImageNet [1] Challenge. The deep Convolutional Neural Network (CNN) designed by Alex Krizhevsky won the competition and set a groundbreaking record. The network named AlexNet [2] was 10.8 percentage points clear from their nearest contender and started the neural network boom [3]. Since then researchers have focused on training deeper neural networks and hence the term Deep Learning arose. Unlike shallow machine learning models, the performance of ANNs continue to increase as the model gets deeper and the amount of training data increases [3]. The development of Graphics Processing Units (GPU) have played a vital role as they facilitate efficient ANN training through parallel computing [4]. But in absence of large amounts of training data these powerful models result in high variance. The model overfits to the training data and does not perform well on unseen real world scenarios.

Large amounts of data cannot be obtained for most real life applications. Even when possible, it often a lengthy time consuming process and very expensive. This counteracts the capability of these models. To overcome this problem Transfer Learning provides a promising alternative where the model is pretrained on a similar but different dataset [5] and later on the task specific dataset. By convention, only the weights of the final few layers are finetuned while the earlier ones are kept frozen. This results in better performance than a random initialization of weights. But Yosinski reports that when using a pretrained model, the performance of the network increases as the number of fine-tuned layers increase [6]. This can be useful for tasks such as facial expression analysis.

The Static Facial Expressions in the Wild (SFEW) [7] is a dataset which consists of movie scenes containing faces. This was originally extracted from the Acted Facial Expressions in the Wild (AFEW) [8]. AFEW is a temporal data which was extracted from movie scenes. The original SFEW has 700 natural occurring face images in varied poses and different lighting conditions. This makes categorization hard on this dataset compared to other carefully constructed datasets like JAFFE [9] and Multi-PIE [10]. Training a deep neural network on a dataset of this size can lead to overfitting. In our experiments we used a pretrained AlexNet model trained on the ImageNet dataset for transfer learning.

Even in the successful scenario when the network does not overfit, it requires a huge amount of computation. Most often the network requires more hidden units than necessary. Studies have shown [11] that a number of these excess hidden neurons can be removed after training without decreasing the performance. Although many methods are available for distinguishing between useful and redundant hidden units, Gedeon suggests that this process can be automated [11]. The angular separation between neurons in vector space can be used as the separation criterion. Neurons which have similar behavior have low angles between them. Conversely, neurons which exhibit opposite behavior to each other have high angles between them, possibly close to 180 degrees. Both these categories of neurons can be considered for removal. This is performed by calculating the vector angle a neuron with all the other neurons of the same layer. But as the number of hidden units in a layer increases this becomes intractable. To speed up the process, we only considered the calculation of the vector angle between two neurons if each of their norms are more than one standard deviation away from the mean of the layer.

In this paper we analyze the performance of a pretrained AlexNet model trained on the ImageNet dataset when different number of layers were finetuned. To reduce the network size and training cost we then pruned neurons on the 7th layer using vector angular separation. We analyze different separation angles and standard deviations for pruning and report the results.

2 Method

2.1 Network Architecture

AlexNet is one of the most popular deep CNN architectures [3]. It has eight layers of processing units. An AlexNet model from PyTorch's torchvision module [12] with pretrained weights was used for training. To be compatible with the SFEW dataset we changed the final fully connected layer to have 7 output neurons.

2.1.1 Loss Function

To account for the 7 output categories, the Cross Entropy [13] loss function was used. Given an input it calculates probability scores for each of the categories.

2.1.2 Optimization Function

To optimize the learnable parameters of the network the Adaptive Moment Estimation (Adam) [14] optimizer was used. The use of this optimization function helps the model to converge faster.

2.2 Dataset Details

To train the network a subset of the SFEW dataset was used which has 675 images in total. The images belong to seven categories representing seven facial expressions, namely: `'Emotion', 'Angry', 'Disgust', 'Fear', 'Happy', 'Neutral', 'Sad'` and `'Surprise'`. Each of the categories have 100 samples except the `'Disgust'`, which has 75 samples.

2.2.1 Data Preprocessing

As the dataset contains naturally occurring facial images in different lighting conditions, some preprocessing was done. Images in each training batch were randomly rotated between a 30 degree offset. Then random crops were taken of the image and resized to 224 x 224 x 3 as required by AlexNet. Here, 224 represents both height and width and 3 represents the number of color channels. In case of test and validation images, instead of taking crops of the images they were directly resized to 224 x 224x 3. Furthermore, all data were normalized by the corresponding mean and standard deviation.

2.2.2 Splitting the dataset

The dataset was randomly split to have 60% data for training, 20% data for validation and 20% data for testing. This resulted in 392 for training, 149 for validation and 134 for testing. A seed of 1 was used to produce the same split every time for consistency.

2.3 Pruning Neurons

After training completion, the model with the highest test accuracy (50.00%) was used to detect and remove unnecessary hidden units using an automatic approach [11]. Neurons were picked for removal in the following way:

i. The output activation vector of the 7th hidden layer was extracted for all the training samples. This resulted in a vector of shape (number of training samples x number of hidden units) or (392 x 4096)
ii. The values were then normalized between [-05, 0.5]
iii. The vector angle was calculated between pairs of neurons in the activation vector

This vector angle was used to measure the functionality of the neurons which ranges from [0, 180]. To remove excess neurons, two cutoff degree values were used namely: `degree_low` and `degree_high`. The neurons were pruned in the following way:

i. Any pair of neurons which had a vector angle less than `degree_low` or more than `degree_high` were considered for removal.
ii. All the weights of the second weight vector were added to the first weight vector.
iii. The weights of the neuron corresponding to the second weight vector was set to zero. This is analogous to the direct removal of the neuron from the network [11].

But there is a limitation to this method. As there are 4096 hidden neurons in the selected layer, calculating the vector angle between each pair of neurons and would result in 4096 x 4096 calculations, which is very expensive if not intractable. To avoid this, a threshold value was used to consider which pair of neurons should be considered for the degree calculation and eventually pruning. For this, the mean norm and standard deviation of the hidden layer outputs was calculated. If both of the neuron's norm were more than a standard deviation away from the mean, only then they were considered for the degree calculation. This resulted in much less computation without compromising too much performance.

2.4 Hyperparameter Search

The network was trained in a loop to search for the optimal hyperparameters. Effects of the following hyperparameters were observed:

- Learning Rate: [0.01, 0.05, 0.001, 0.005]
- Total training epochs: Ranging from 20 to 100 with an interval of 10

It was found that the optimal learning rate of the network was 0.001 and the optimal number of epochs to train was 50. The validation loss started after the 30th epoch and any training after 50 epochs resulted in overfitting.

2.5 SPI Baseline

The Strictly Person Independent (SPI) [7] is an automated evaluation process. The baseline is calculated as follows:

$$\begin{aligned} \text{Overall Accuracy} &= \frac{tp+tn}{tp+fp+fn+tn} \\ \text{Class Wise Precision} &= \frac{tp}{tp+fp} \\ \text{Class Wise Recall} &= \frac{tp}{tp+fn} \\ \text{Class Wise Specificity} &= \frac{tn}{tn+fp} \end{aligned}$$

Where *tp* = true positive, *fp* = false positive, *fn* = false, negative, and *tn* = true negative. The top test result was reported using this SPI baseline.

3 Results and Discussion

3.1 Network Accuracy prior to pruning

Initially, we ran a grid search to find the optimal learning rate and the total number of epochs to train. Afterwards, we trained the AlexNet model without using the pre-trained weights. As the final fully-connected layer has different number of output units, their weights were reinitialized by default. We observed an average test accuracy of 21.35 % over five runs.

Afterwards, we ran the AlexNet model with pretrained weights from the ImageNet dataset. We sequentially froze the weights of 0 to all 7 of the hidden layers to stop gradients from backpropagating and weights being updated. For each variant we trained the model five times and observed the results.

The results are presented below:

Table 1. Test ccuracy over 5 runs when different numbers of layers are frozen

	Frozen 0	Frozen 1	Frozen 2	Frozen 3	Frozen 4	Frozen 5	Frozen 6	Frozen 7
Run 1	17.16	17.16	19.4	46.27	46.27	47.01	47.01	44.03
Run 2	17.16	17.91	26.12	44.03	46.27	45.52	47.76	38.81
Run 3	23.13	17.16	23.88	39.55	46.27	45.52	45.52	44.78
Run 4	14.18	17.16	23.13	35.82	47.01	45.52	50	41.04
Run 5	17.16	17.16	27.61	48.51	48.51	49.25	44.03	42.54
Mean	**17.758**	**17.31**	**24.028**	**42.836**	**46.866**	**46.564**	**46.864**	**42.24**

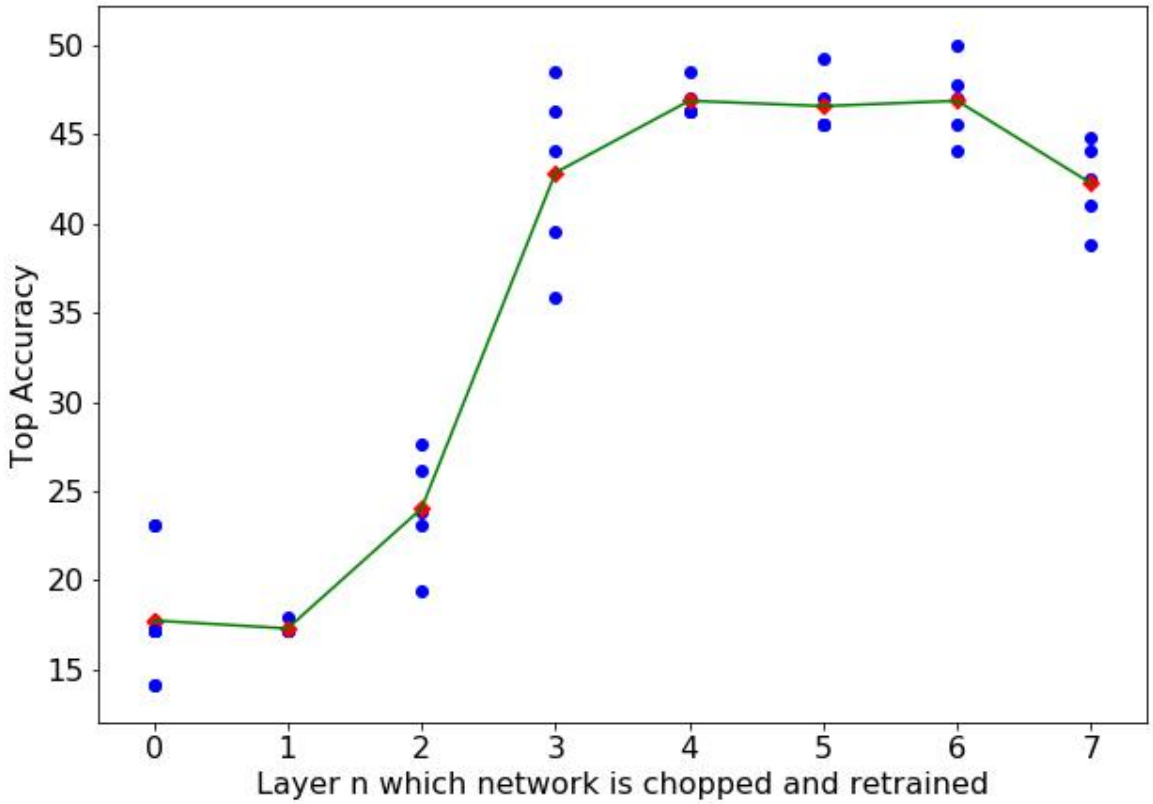

Fig. 1. Test ccuracy of the network as different number of layers are frozen. (Blue – Actual values, Red – Mean values)

This indicates that if all the layers are finetuned after loading the pretrained model it results in a worse performance (17.458% test accuracy) than training the network from scratch using random weights. This could indicate that the initial features learned on the AlexNet model is specific to finding general types of objects. But not specific enough to capture fine features like facial expressions. Although the size of the dataset might play a factor here.

It was also noticed that as the number of frozen layers increased, the accuracy increased. By freezing layers between 4 to 6, the average test accuracy went up to 46.86%. The highest test accuracy (50.0%) was recorded by freezing 6 layers. Finally, the accuracy decreased again on the 7^{th} layer. This can be interpreted as an effect of using a distant dataset of different distribution.

The confusion matrix and SPI parameters of the top accuracy (50.0%) are presented below:

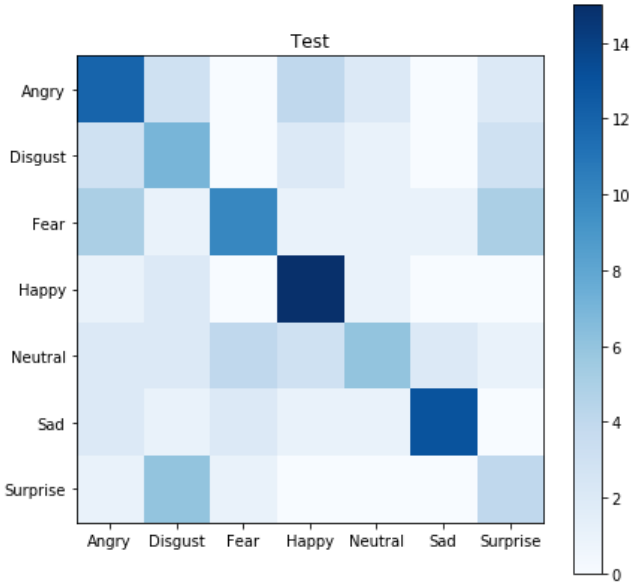

Fig. 2. Confusion matrix where test accuracy was 50.0%

Table 2. SPI scores where the test accuracy was 50.0%

Emotion	Angry	Disgust	Fear	Happy	Neutral	Sad	Surprise
Precision	0.46	0.32	0.59	0.58	0.50	0.81	0.27
Recall	0.52	0.44	0.42	0.79	0.30	0.65	0.33
Specificity	0.87	0.87	0.94	0.90	0.95	0.97	0.91

The confusion matrix suggests that most of the classes were classified properly except the '`surprise`' and the '`disgust`' class. This can be because some other context might be necessary rather than facial expressions to recognize these expressions. Also, the SPI scores suggests a significant improvement from the experiments done by authors of [7].

3.2 Observing the Behavior of using different vector angles

Finally, we observed the network's accuracy and how many neurons were pruned using different vector angles and standard deviations for pruning. Here too we used a grid search method to observe the performance using angles of 7.5 to 17.5 at interval of 2.5 and standard deviations of 1 to 5 at an interval of 0.5. For pruning, the model with 50% test accuracy was used. The results are presented below:

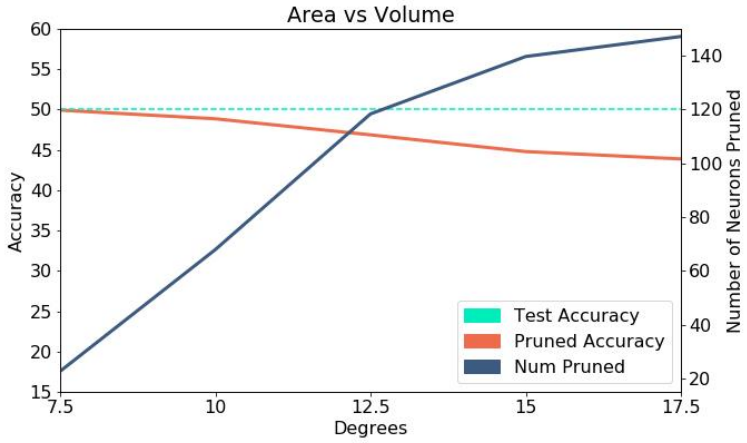

Fig. 3. Figure represents the networks performance when different angular separation degrees are used

The following was observed by using different vector angles:

Table 3. Accuracy and number of pruned neurons after pruning by using different vector angles

	Degree 7.5	**Degree 10**	**Degree 12.5**	**Degree 15**	**Degree 17.5**
Accuracy	49.92%	48.84%	46.85%	44.78%	43.86%
Num Pruned	22.66666667	67.88888889	118.3333333	139.6666667	147.111111

The accuracy for each degree category was calculated by taking the mean across all the standard deviation thresholds. Both Table 3 and graph 3 suggest that as the degree angles increased, more neurons were pruned and the accuracy decreased. But these are the average of multiple runs. In our experiments there were a few cases found where a large number of neurons were pruned without reducing the network's performance. One example is using 12.5 degrees 292 neurons were pruned while still maintaining 50.0% accuracy another case was using 15 degrees 175 neurons were removed without affecting performance.

4 Future Work and Conclusion

Although transfer learning can provide a starting point for training deep neural networks the number of layers to fine tune plays a crucial part in the performance of the model. The results in this paper suggest that for a dataset with limited samples like the SFEW dataset, transfer learning can provide a boost to performance. But fine tuning all the layers results in worse performance that training with random weights. Also freezing all the layers except the last one does not result in the optimal result. The ideal number of layers for the AlexNet model on the SFEW would be recommended between 4 to 6.

For future work, the effect of pruning can be analyzed by reducing the number of neurons in the final fully-connected layer. Also, effect of fine tuning can be analyzed using more complex generalization methods.

5 References

[1] O. Russakovsky *et al.*, "ImageNet Large Scale Visual Recognition Challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, 2015, doi: 10.1007/s11263-015-0816-y.

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 6, pp. 84–90, 2017, doi: 10.1145/3065386.

[3] M. Z. Alom *et al.*, "The History Began from AlexNet: A Comprehensive Survey on Deep Learning Approaches," Mar. 2018, Accessed: May 31, 2020. [Online]. Available: http://arxiv.org/abs/1803.01164.

[4] NVIDIA Corporation, "Deep Learning | NVIDIA Developer," 2019. https://developer.nvidia.com/deep-learning (accessed May 31, 2020).

[5] L. Torrey, J. S.-H. of research on machine learning, and undefined 2010, "Transfer learning," *igi-global.com*, Accessed: May 31, 2020. [Online]. Available: https://www.igi-global.com/chapter/transfer-learning/36988.

[6] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?," *Adv. Neural Inf. Process. Syst.*, vol. 4, no. January, pp. 3320–3328, 2014.

[7] A. Dhall, R. Goecke, S. Lucey, and T. Gedeon, "Static facial expression analysis in tough conditions: Data, evaluation protocol and benchmark," *Proc. IEEE Int. Conf. Comput. Vis.*, pp. 2106–2112, 2011, doi: 10.1109/ICCVW.2011.6130508.

[8] A. Dhall, R. Goecke, S. Lucey, and T. Gedeon, "Acted Facial Expressions In The Wild Database," 2011, Accessed: May 31, 2020. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.224.7755&rep=rep1&type=pdf.

[9] M. Lyons, S. Akamatsu, M. Kamachi, and J. Gyoba, "Coding facial expressions with Gabor wavelets," in *Proceedings - 3rd IEEE International Conference on Automatic Face and Gesture Recognition, FG 1998*, 1998, pp. 200–205, doi: 10.1109/AFGR.1998.670949.

[10] R. Gross, I. Matthews, J. Cohn, T. Kanade, and S. Baker, "Multi-PIE," *Image Vis. Comput.*, vol. 28, no. 5, pp. 807–813, 2010,

doi: 10.1016/j.imavis.2009.08.002.

[11] T. D. Gedeon and D. Harris, “Network Reduction Techniques,” in *Proceedings International Conference on Neural Networks Methodologies and Applications*, 1995, pp. 119–126.

[12] Pytorch, “PyTorch.” https://pytorch.org/ (accessed May 31, 2020).

[13] G. E. Nasr, E. A. Badr, and C. Joun, “Cross Entropy Error Function in Neural Networks: Forecasting Gasoline Demand,” 2002. Accessed: May 31, 2020. [Online]. Available: www.aaai.org.

[14] D. P. Kingma and J. L. Ba, “Adam: A method for stochastic optimization,” in *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, Dec. 2015.