Analysis of the effectiveness of Neural Network Reduction techniques on Static Facial Expression In the Wild Dataset

Tanya Dixit

The Australian National University, Canberra ACT 0200, Australia
Tanya.Dixit@anu.edu.au

Abstract. This paper talks about using Neural Network Reduction techniques on the SFEW (Static Facial Expression in the Wild) dataset. The dataset contains images of expressions depicting seven emotions. The PCA (Principal Component Analysis) vectors of image features are used to train a feed-forward neural network for classification. Then, to improve on this, a pretrained Convolutional Neural Network (CNN) Resnet-18 is used on the pre-processed RGB images to get better accuracy than feedforward networks. The pruning technique of "distinctiveness" is compared to percentile pruning and it is proved that percentile pruning works better. Further investigations are carried to prove the Lottery Ticket Hypothesis as given in Frankle and Carbin (2018) for pretrained networks.

Keywords: Neural Networks · Network Reduction · Distinctiveness

1 Introduction

Deep Neural Network architectures offer high accuracies on difficult datasets and problems but come with high storage and computational costs. The millions of parameters in such architectures, while necessary sometimes lead to redundancies and can be removed without affecting the accuracies negatively. The techniques of neural network reduction have been around for a long time [1] and allow practitioners to reduce the parameter counts of a huge network. This facilitates faster inference and lighter networks that provide commensurate accuracies as the original bigger networks. Several techniques have been proposed in this area like global magnitude pruning, layerwise magnitude pruning, global gradient magnitude pruning, layerwise gradient magnitude pruning, random pruning, and more [2], [3]. We study one such technique based on Gedeon and Harris (1999) [8] that removes nodes based on their similarity with other nodes in the same layer. We use the SFEW dataset for our study. This dataset is particularly difficult because it models facial expressions realistically rather than posed expressions. Facial expression detection is a vital area of research and application, but one where real data is not very common. Most of the datasets contain images that are posed and [4], [5], [6] and most of the state of the art models trained on such datasets would experience reduced performance on in-the-wild datasets [7]. Datasets like LFW and Pubfig database contain natural movements, different illumination conditions, age, ethnicity, etc.

The original paper by Dhall et al. [7] focused on the creation of a similar dataset called SFEW. SFEW was constructed by taking static frames from AFEW which is a dynamic temporal facial expression corpus constituting real-world expressions. The dataset is taken from movies where specific static frames are labeled with the emotion the actor is portraying. Movies mimick real-world situations and are not posed, and that is why this dataset models real-world situations much better than datasets like JAFFE and Multi-PIE.

This paper extends the study done by Dhall et al. [7], and shows that Neural Networks can handle and give a better accuracy on such a dataset in difficult conditions. The investigations include various pre-processing techniques that were tried along with hyperparameter experiments. A major contribution is to analyze the effect of pruning hidden nodes on the effectiveness of the Neural Network model in the proposed scenario. In Gedeon and Harris [8], neural network pruning was implemented to measure the distinctiveness of hidden units. The units that were too similar or complementary were removed using techniques discussed in Section 2.5. We apply these to a feedforward network trained on the PCA components of the images and report the accuracies. We extend our experiments to a finetuned Resnet-18 model on the RGB images. Apart from using distinctiveness to prune the Resnet-18 weights, we also use percentile pruning to remove a certain percentage of the nodes with lowest magnitudes and re-train the Resnet for the same number of epochs as used to fine-tune. The final accuracies of the re-trained networks are compared and percentile pruning stands out as a better technique to remove a higher percentage of nodes. We further analyze that percentile pruning is a superior technique and leads to the creation of Lottery Ticket winning architectures [9]. We follow the methodology given in [9] to prove that percentile pruned Resnet-18 network reaches commensurate accuracies faster than a random pruned network, thus proving that the Lottery Ticket Hypothesis works even for pre-trained networks. The metric used throughout is accuracy as the classes are balanced and it is a metric used by the original paper as well.

2 Method

2.1 Dataset

The SFEW contains 700 images that have been labeled for 7 different classes - angry, disgust, fear, happy, neutral, sad, and surprise for 6 emotions and 1 neutral emotion (675 in this paper as 25 samples of the emotion 'disgust' haven't been included). We deal with two versions of the dataset - the first version contains the first two principal components (each dimension 5) of the LPQ (Local Phase Quantization) and PHOG (Pyramid of Histogram Oriented Gradients) features of each image. Therefore, each image is characterized by 10 features. We also follow the Strictly Personal Independent protocol in this data i.e. the images are person independent. However, the second version of the dataset contains the RGB images itself of the 7 emotions. The width and height of each image are 720 and 576 pixels respectively.

2.2 Data Exploration and Preprocessing

For version 1 of the dataset, the data was processed to eliminate any nan values. Since, the dataset contains two principal components of the images, the order of magnitude of the first five inputs (the first principal component) differs from the second five inputs. The normalization technique used took the mean and standard deviation of all the inputs of the first five features and normalized the first principal component, and similarly, the next five inputs were normalized.

For version 2 of the dataset, it was observed that the images contain not just the face but other scene information as well. Since the scenes were not adding any distinguishing information, the faces were cropped out using a pre-trained MTCNN (Multi-task Cascaded Convolutional Neural Networks). After cropping, the dataset was split into test and train sets (10 percent split was finalized after experimenting with different splits). The images were all resized to 224x224. Since it's a small dataset, data augmentation techniques were applied including flipping the images horizontally, applying shift, scale and rotate, blurring the images, and random brightness contrast to the images. As a result, the dataset grew to three times its original size giving 1821 train images and 204 test images.

2.3 Model Design

The model for version 1 of the dataset is a one-hidden layer neural network with 10 input nodes, a hidden layer with 30 nodes, and 7 output nodes for 7 classes. The hyperparameters of the model were selected based on extensive experiments by varying the learning rate, batch size, number of epochs, and the number of hidden nodes. Cross-Entropy was used as the loss metric since it is the most widely used for classification problems. Also, a Sigmoid function was used as activation for the hidden nodes based on experimental results with ReLU, Tanh, and Sigmoid. The average accuracy reported was 28.3 percent with a standard deviation of 5 percent using hyperparameters - batch size=1, learning rate=0.001, and epochs=1200.

The model for version 2 is a pre-trained Resnet-18 that has been trained on the ImageNet dataset. It is 18 layers deep and takes an image input of 224 by 224. The model was fine-tuned on our dataset with learning rate=0.001, batch size=8, and epochs = 25 and reported a 50 percent average accuracy on the dataset with a standard deviation of 3 percent. The fine-tuning experiments were performed with different hyperparameters and the above setting gave the best test accuracy.

2.4 Network Reduction Technique and Improvements

After zeroing in on a model, network reduction techniques were performed to verify the hypothesis. In the following sections, we use different techniques to prune the neural network and measure the effects on test accuracy.

On the feedforward neural network, the measurement of similarity between hidden units was determined from the unit output activation from the aforementioned hidden units. We first normalize these output activations to get their values from -0.5 to 0.5. The second step is to find the angle between the hidden unit vectors in the training data space (using cosine similarity) and start pruning. Hidden unit activations with smaller angles than an upper bound between them are potentially similar and can be handled by removing one of the units and adding the weights of the removed one to the hidden unit that is similar to it. Whereas, units with larger angles than a lower bound between them are canceling each other out and can both be removed. After pruning, the test accuracy is calculated without re-training and is compared to the test accuracy before pruning.

For the Resnet model, it is computationally expensive to get the outputs from each convolution layer to calculate angles. Note that we are only pruning conv2d layers here. We propose that distinctiveness can also be measured between weights directly and hence prune similar nodes based on the weight vectors of each convolution layer. Each hidden weight is a vector of size (input layer size) x (filter size) x (filter size). Pruning

is not applied to the first convolution layer as it is getting information from the input and pruning it will lead to insufficient information being passed to successive layers. We vary the lower bund on the angle and observe the re-training behavior when trained for 25 epochs (same as the unpruned model).

The "distinctiveness" technique on Resnet is compared to Percentile Pruning where we prune a percentage of weights with lowest absolute magnitudes in a layer (we exclude zero magnitude weights in the calculation of the percentage). We vary the pruning percentage and observe the re-training behavior when trained for 25 epochs.

3 Results and Discussion

3.1 Comparison among models

First, let's compare the neural network model and Resnet-18 with the original paper[7]. The model used by the previous paper was an SVM with a radial basis function. The accuracy of SVM with 5-fold cross-validation on the dataset (SIP protocol) was 19 percent. The feedforward neural network model performs better with an average accuracy of 28.3 percent and the Resnet performs even better with an average test accuracy of 50 percent.

3.2 Accuracy analysis on the feedforward network after applying Pruning Techniques

We try experiments with a different number of hidden units (30, 50, and 100). According to Gedeon and Harris [8], the bounds were 15 and 165 degrees, but in these experiments, no pairs were found in these ranges. Therefore, we perform two sets of experiments separately and with modified bounds.

Accuracy Analysis when removing similar nodes In this section, results are presented in Table 5,6 and 7 for hidden nodes 30, 50 and 100 respectively.

Runs	Pairs and Degree measures	Test Accuracy Before removal	Test accuracy after removal
Run 1	2 pairs less than 45 degrees	20.59	20.59
Run 2	2 pairs less than 45 degrees	25	8.82
Run 3	1 pair less than 50 degrees	27	25

Table 1: Accuracy changes after removing similar nodes for 30 hidden layer size

Runs	Pairs and Degree measures	Test Accuracy Before removal	Test accuracy after removal
Run 1	7 pairs less than 45 degrees	11.76	17.65
Run 2	4 pairs less than 45 degrees	16.18	20.59
Run 3	1 pair less than 45 degrees	25	23.53

Table 2: Accuracy changes after removing similar nodes for 50 hidden layer size

Runs	Pairs and Degree measures	Test Accuracy Before removal	Test accuracy after removal
Run 1	12 pairs less than 45 degrees	19.12	19.12
Run 2	7 pairs less than 45 degrees	23.53	22.06
Run 3	1 pair less than 35 degrees	23.53	25

Table 3: Accuracy changes after removing similar nodes for 100 hidden layer size

From the analysis presented above, it can be seen that pruning similar nodes as per instructions discussed in section 2.5 leads to different results for different hidden sizes. For 30 hidden nodes, pruning almost always

results in lowering accuracy. This can be attributed to the fact that lower the number of hidden units, the more valuable each node is because there aren't enough nodes to be redundant and that each node contributes something. On the other hand, when the hidden layer size increases, removal doesn't affect the test accuracy by much as a model with large hidden units might be overfitting on the training set, hence pruning might offer some sort of regularization and makes the network more general and give a better accuracy on the test set.

Accuracy Analysis when removing complementary nodes In this section, results are presented in Tables 7 and 8 for hidden nodes 50 and 100 respectively. We have skipped results for 30 hidden nodes because we didn't find any pairs with angles greater than 140 degrees.

Runs	Pairs and Degree measures	Test Accuracy Before removal	Test accuracy after removal
Run 1	2 pairs greater than 140	13.24	20.59
Run 2	1 pair greater than 140	22.06	20.59
Run 3	2 pairs greater than 140	19.12	20.59

Table 4: Accuracy changes after removing complementary nodes for 50 hidden layer size

Runs	Pairs and Degree measures	Test Accuracy Before removal	Test accuracy after removal
Run 1	4 pairs greater than 140	19.12	20.59
Run 2	8 pairs greater than 140	11.76	19.12
Run 3	3 pairs greater than 140	25	23

Table 5: Accuracy changes after removing complementary nodes for 100 hidden layer size

From the above tables, it is seen that removing nodes that have greater than 140 degrees angle between their output activation vectors doesn't lead to a steep fall in accuracy.

3.3 Accuracy analysis on the Resnet-18 network after applying Pruning Techniques and Re-training

For experiments in this section, we use a saved model every time so that the testing conditions do not vary and results are reliable. We compare the two pruning techniques in Fig. 1. For distinctiveness pruning, we try different angles between the convolutional layer weights of Resnet-18 and see the percentage of weights pruned from each layer. For comparison with percentile pruning, we take the highest percentage pruned among all the convolutional layers and perform percentile pruning with that percentage. In Fig. 1, we compare their performance when pruning is 21.875 percent (angle 45 degrees) and 50 percent (angle 50 degrees). We also tried with angles lesser than 45 degrees for the distinctiveness technique, but the pruned percentage was way too small for comparison.

We observe that when the percentage pruned is less, the pruned networks can reach commensurate accuracy as unpruned in both cases. But when percentage pruned becomes higher, Percentile pruning is more effective than distinctiveness pruning in regaining the accuracy levels. In fact, at 50 percent pruning, the network never reaches an accuracy higher than 20 percent establishing that it has become oversimplified. This may be because, in distinctiveness pruning, we are removing complete filters and not just individual nodes. While some filters may be similar to others, they still might perform an important function and have high magnitude weights. Removing such high magnitude weights might be leading to decreased learning capacity. Another reason could be that whereas a filter might be redundant at one layer, it might be propagating information that is essential for the next layer.

3.4 Lottery Ticket Hypothesis on pre-trained Resnet-18

The Lottery Ticket Hypothesis formulated by [9] states that "A randomly initialized, dense neural network contains a sub-network that is initialized such that - when trained in isolation - it can match the test accuracy of the original network after training for at most the same number of iterations". The experiments in [9] were based on an iterative pruning technique by training different network architectures from scratch. Here,

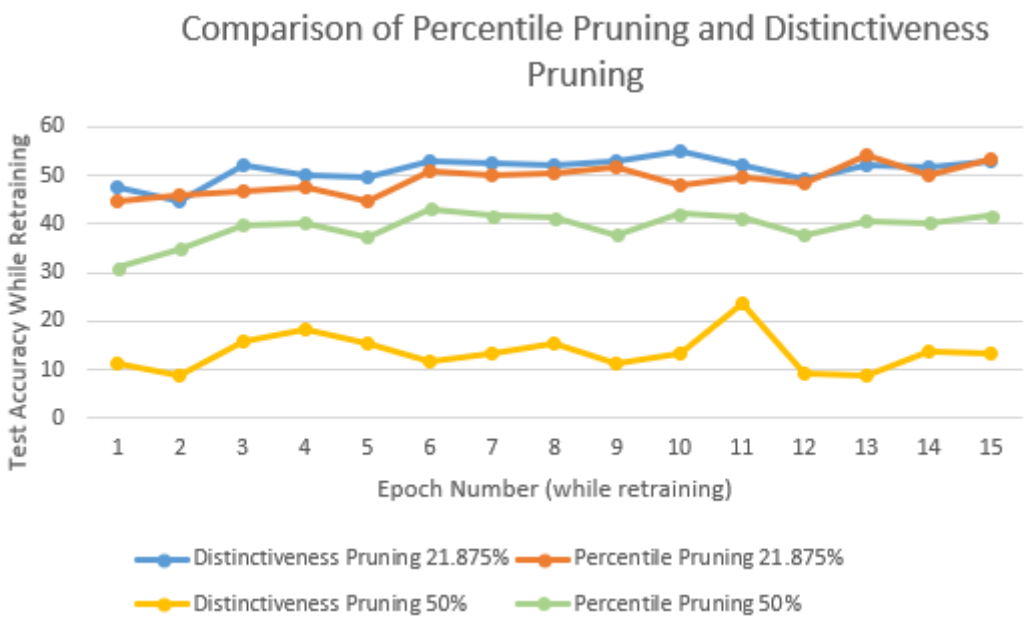

Fig. 1: We compare percentile pruning and distinctiveness pruning. Observe that for 50 percent pruning, percentile pruning is still able to reach commensurate test accuracy as unpruned model, whereas distinctiveness pruning is not.

we perform one-shot percentile pruning on the fine-tuned network, initialize the unpruned weights back to the pretrained weights, and then train back again for the same number of epochs and try to find if such sub-networks exist. We take the same fine-tuned model (49.5 percent acuracy) for all the experiments in this section. The steps are:

1. Load the state dictionary of the fine-tuned model (this was saved in advance when fine-tuning was done)
2. Prune a percentage of weights using percentile pruning
3. Initialize the unpruned weights to the initial weights (pre-trained weights)
4. Train the model again for the same number of epochs as used for fine-tuning
5. Observe the highest accuracy reached and the iterations it takes to reach there

In Fig. 2, we report the retraining observations with different percentages of the network using the technique mentioned above. We can prune up to 50 percent of the network and it is still able to regain accuracy withing a few epochs as seen in Fig. 2 even after being re-initialized to previous weights. For higher pruning like 70 and 90 percent, the network doesn't reach the previous accuracy. The reason might be that the network has become too simple or maybe with such heavy pruning, Lottery Ticket winning subnetworks are not formed with this particular combination of dataset and architecture. We need more experiments with other architectures to confirm this.

3.5 Proving the Lottery Ticket Hypothesis on pre-trained Resnet-18

In the previous section, it was established that subnetworks at least 50 percent the size of the unpruned network exists in Resnet-18 that give commensurate accuracy in at most the same number of epochs. But it remains to be proven that these subnetworks aren't a random occurrence, but are a result of the pruning strategy employed. Hence, we repeat the experiments performed in Section 3.4, but with a small difference. Instead of using percentile pruning, we use random pruning on each convolutional layer.We find that randomly pruning networks lead to slower re-training. Also, the randomly pruned network doesn't reach the same accuracy as the percentile pruned network, especially when the pruned weight percentage is high. We show the results in the Appendix for this.

3.6 Final Words on the Results

For feedforward networks, the pruning techniques lead to low accuracies when the number of hidden units is just right, but when hidden units are large (larger than required), pruning leads to the same or higher test accuracies. This can be attributed to not only overfitting of large networks on the training set, but also to the

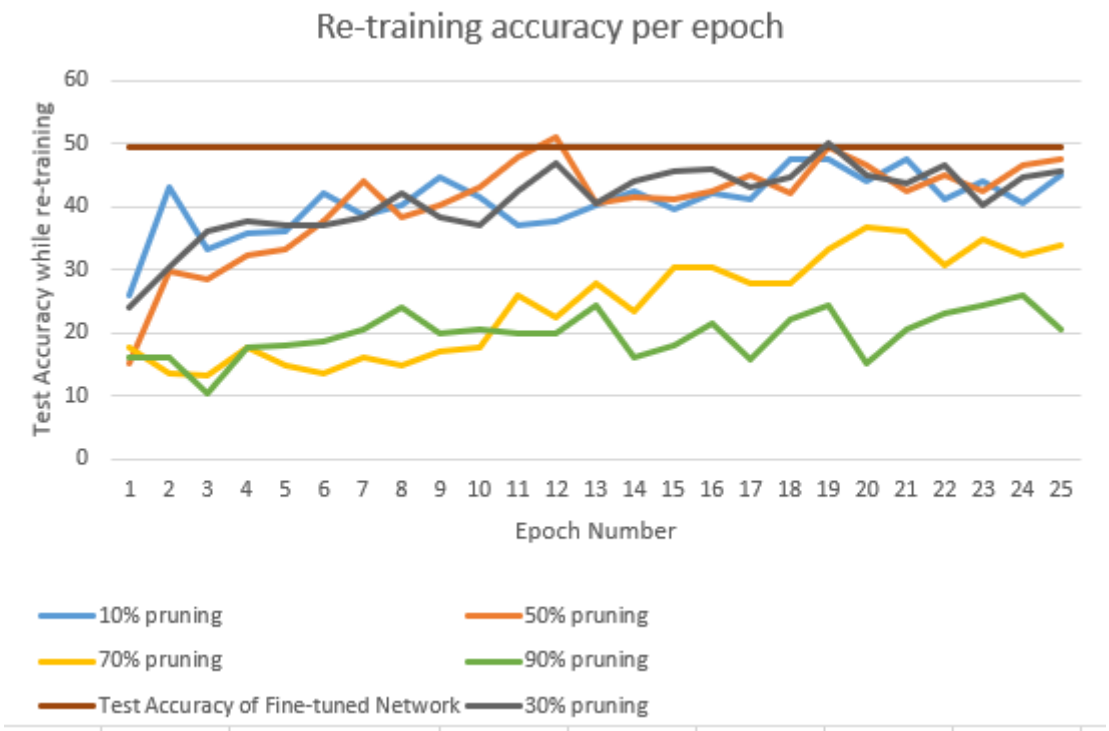

Fig. 2: Here we are re-training after initializing to the original pre-trained weights of Resnet-18 (weights before fine-tuning). This experiment clearly shows that a network even 50 percent the size of the original network can learn similarly and reach an accuracy that the unpruned network reaches in the same number of epochs.

fact that SFEW dataset is of a smaller size and involves a slight correlation between inputs. When using CNN on the augmented dataset, much better accuracy is observed. Pruning CNN weights using different techniques lead to quite different results. Distinctiveness removes filter weights whereas Percentile Pruning removes weights one by one, and that could be the reason why Percentile Pruned networks can recover their accuracy via re-training. To find subnetworks that lead to commensurate accuracies when re-trained, percentile pruning is better than random pruning. We can prune 50 percent of the network using percentile pruning and can re-train it from the initial weights of the unpruned network and get commensurate test accuracy.

4 Conclusion and Future Work

This paper has highlighted that Neural Network Reduction techniques are effective in cases where the model size is large. The model accuracy reported is better than what was reported by [7] and goes to show how neural networks might be effective when the data is non-linearly separable. It also shows that Deep Learning techniques might give good accuracies even with less data when combined with proper pre-processing and data augmentation[10], [11]. However, there is still scope for further improvement. As an extension of this work, Variational Autoencoders or Generative-Adversarial Networks can be used to create new images and try to augment such small datasets[12].

Different pruning techniques give wildly different results and if an effective way is discovered to find the subnetworks and initial weights that give the same accuracies as the unpruned network, a lot of computational power can be saved[1]. The work done in this paper can be extended to find out exactly the amount of compression and computational speedup achieved on bigger datasets using Distinctiveness and Percentile Pruning techniques. Also, more research needs to be done concerning the Lottery Hypothesis in Pre-trained networks with other datasets and architectures. A more theoretical understanding is required to leverage this finding to reduce networks without having to train them all the way. Future work includes comparing the Lottery Tickets architecture on the same dataset and same neural network architecture, but between pre-trained weights (on a standard dataset like ImageNet) and weights trained from scratch on the same dataset. Here, since the dataset is small, such an analysis, though was performed, did not lead to any significant results. Pruning techniques can be improved and applied for different neural network architectures by modeling the minimization of correlation of activation outputs as an optimization problem as well [13].

References

1. Blalock, D., Ortiz, J. J. G., Frankle, J., Guttag, J.,(2020). What is the state of neural network pruning?:arXiv preprint.
2. Huang, Q., Zhou, K., You, S. and Neumann, U., (2018), March. Learning to prune filters in convolutional neural networks. In 2018 IEEE Winter Conference on Applications of Computer Vision (WACV) (pp. 709-718). IEEE.
3. Li, H., Kadav, A., Durdanovic, I., Samet, H. and Graf, H.P., (2016). Pruning filters for efficient convnets. arXiv preprint arXiv:1608.08710.
4. Kanade, T., Cohn, J., Tian, Y.: Comprehensive Database for Facial Expression Analysis. Proceedings of the Fourth IEEE International Conference on Automatic Face and Gesture Recognition 2000, pp. 484–490. USA (2000). https://doi.org/10.5555/795661.796155
5. Kamachi, M., Lyons, M., Gyoba, J.: The Japanese Female Facial Expression (JAFFE) Database [Data set]. Zenodo (2000). https://doi.org/10.5281/zenodo.3451524
6. The MMI Facial Expression Database: www.mmifacedb.com
7. Dhall, A., Goecke, R., Lucey, S., Gedeon, T. (2011, November). Static facial expressions in tough conditions: Data, evaluation protocol and benchmark. In 1st IEEE International Workshop on Benchmarking Facial Image Analysis Technologies BeFIT, ICCV2011.
8. Gedeon, T. D., Harris, D. (1991). Network reduction techniques : In Proceedings International Conference on Neural Networks Methodologies and Applications (Vol. 1, pp. 119-126).
9. Frankle, J., Carbin, M.,(2018).The lottery ticket hypothesis: Finding sparse, trainable neural networks.:arXiv preprint.
10. Hussain, M., Bird, J.J. and Faria, D.R., (2018), September. A study on cnn transfer learning for image classification. In UK Workshop on Computational Intelligence (pp. 191-202). Springer, Cham.
11. Huan, E.Y. and Wen, G.H., 2020. Transfer learning with deep convolutional neural network for constitution classification with face image. Multimedia Tools and Applications, pp.1-15.
12. Frid-Adar, M., Klang, E., Amitai, M., Goldberger, J. and Greenspan, H., 2018, April. Synthetic data augmentation using GAN for improved liver lesion classification. In 2018 IEEE 15th international symposium on biomedical imaging (ISBI 2018) (pp. 289-293). IEEE.
13. Cogswell, M., Ahmed, F., Girshick, R., Zitnick, L., Batra, D., (2016). Reducing Overfitting in Deep Networks by Decorrelating Representations: International Conference on Learning Representations 2016.

5 Appendix

5.1 Comparison of Re-training accuracies between Percentile pruning and Random Pruning

In this section we present the results that we talked about in Section 3.5 and attempt to prove the Lottery Ticket Hypothesis. We present epoch-wise re-training accuracies and compare percentile vs random pruning using method outlined in Section 3.5. It is observed that in most cases, randomly pruned subnetworks re-train slowly and reach lower accuracies than percentile-pruned networks, hence proving that some weights matter more than others and the subnetworks that consist of those weights are the ones that have "won the lottery" and that it's not a random occurence.

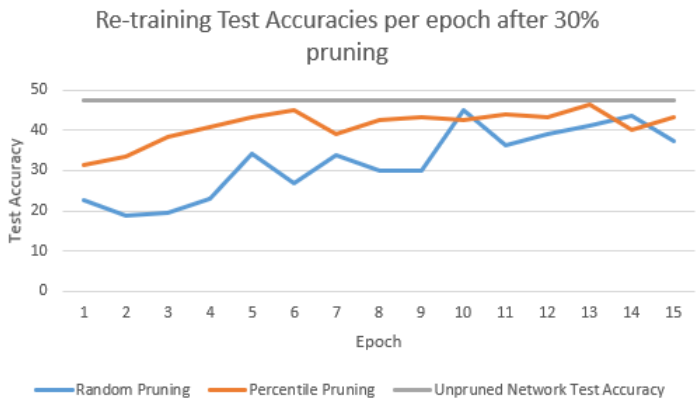

Fig. 3: Both random and percentile pruning reach the test accuracy of the unpruned network, but it takes more epochs for random pruning and the network struggles with local minimas initially

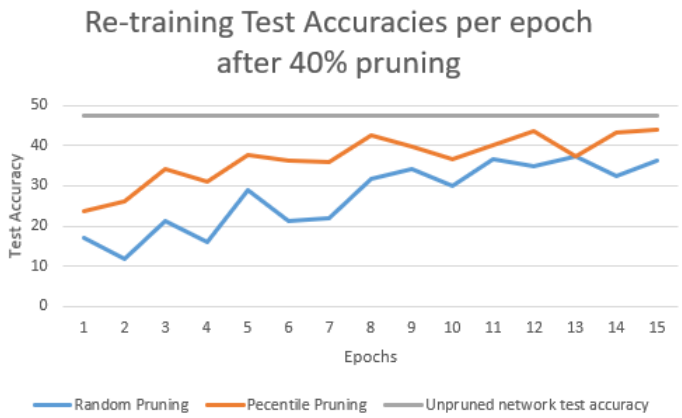

Fig. 4: As pruning percentage increases, now a significant difference is observed between random and perentile pruning. The final test accuracy reached also differs significantly.

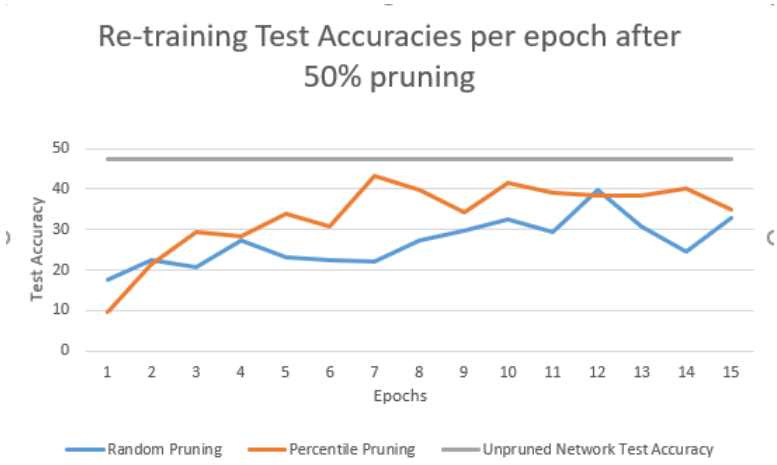

Fig. 5: At 50 percent pruning, we see that randomly pruned subnetwork struggles to get a good accuracy, while the percentile pruned network is able to reach a high accuracy.

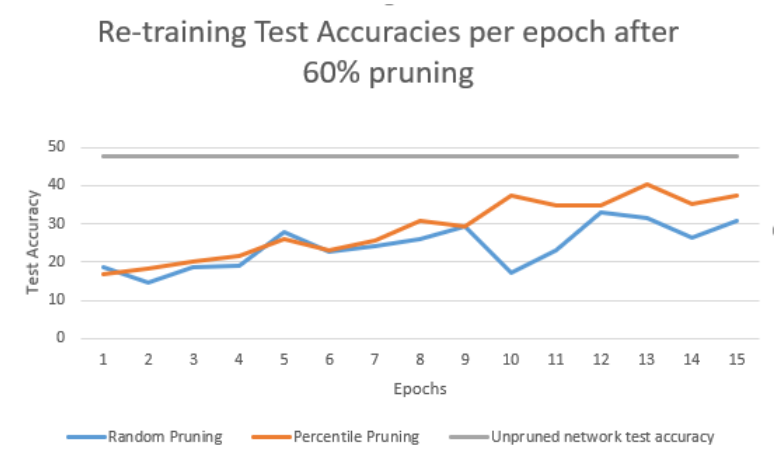

Fig. 6: If we further increase pruning, Lottery Ticket subnetworks are less likely to be formed but still percentile pruning reaches higher accuracy than random pruning