Pruning Fine-tuned models on Static Facial Expression In the Wild Dataset

Nathan Elazar and Tanya Dixit

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

Abstract. Facial expression recognition (FER) models can be used to automatically detect a person's emotions based on an image of their face. In this work we create a FER model by fine-tuning an ImageNet-pretrained ResNet-18 on the Static Facial Expression in the Wild (SFEW) dataset. We aim to investigate how well network pruning methods can function on this fine-tuned FER model. We compare the performance of two different pruning algorithms, magnitude pruning and distinctiveness pruning, and find that magnitude pruning performs significantly better. In-fact, we find that after pruning away 50% of the weights in the network, the network can continue to be trained and reach an even higher test accuracy (66.14%) compared to the original full model (63.36%). In addition, in order to confirm that the lottery hypothesis still holds for fine-tuned networks, we perform experiments where un-pruned weights are re-initialized to their original pre-trained value after pruning. Our experiments successfully demonstrate that there exist subnetworks in the pre-trained ResNet with as little at 5% of the total weights in the full network which can be successfully trained to the same performance as the full model.

Keywords: Neural Networks · Network Reduction · Distinctiveness · Pruning · Lottery Ticket Hypothesis · Facial Expression Recognition · Transfer Learning

1 Introduction

Deep Neural Network (DNN) architectures offer high accuracies on difficult datasets and problems but come with high storage and computational costs. This can pose a problem when deploying DNN in constrained computing environments. Numerous methods have been developed for reducing the size of DNN models while maintining high performance. Pruning based methods seek to remove redundant weights or neurons from an already trained model. Different kinds of pruning, such as global magnitude pruning, layerwise magnitude pruning, global gradient magnitude pruning, layerwise gradient magnitude pruning, random pruning, and more have been successfully employed [2], [3]. For image classification tasks, it has generally been observed that increasing the size of models has lead to improved performance, however often these large models can be effectively pruned down to a small fraction of weights without suffering any decrease in performance [9]. However, the networks must start off large when training and then be pruned; if a small network is trained from scratch it will have significantly worse performance [9]. Recently, simple magnitude based pruning, in which weights with the smallest absolute value are pruned first, in combination with weight re-initialization has shown to be very effective at pruning networks [9]. This finding, dubbed the 'lottery hypothesis' posits that within a randomly initialised network there exist small subsets of connections, called a 'winning ticket', that can be trained to reach the same performance as the entire network. Hence, larger models have a higher chance of containing better tickets at initialization [9].

Our primary aim is to investigate whether or not pruning methods are effective for DNN trained on the Static Facial Expression in the Wild (SFEW) dataset. Facial expression detection is a vital area of research and has numerous applications, but often real-world data is difficult to come by. Most of the facial expression datasets contain images that are posed (e.g [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]. SFEW is particularly difficult because it models facial expressions realistically in a range of different lighting conditions and camera angles. While most recent research on network pruning has been conducted on large datasets, such as CIFAR or ImageNet [9], SFEW only contains 675 images. We are therefore interested in knowing if DNN can be pruned when trained on such a small dataset, and to what extent. As we will be using transfer learning of pre-trained models, we will also investigate how pruning interacts with the fine-tuning process.

Early pruning techniques did not operate at the weight level, but rather pruned entire neurons [8]. As far as we are aware, current experiments with re-initialization based pruning have only considered weight-based pruning, but not neuron-based pruning. We therefore aim to study how one such neuron-based pruning technique, distinctiveness pruning [8], performs in combination with re-initialization to determine if winning tickets can be made from entire neurons.

This paper extends the study done by Dhall et al. [7], and shows that DNN can give a better accuracy on the SFEW dataset. In order to improve the performance of our models as much as possible, various pre-processing techniques and hyperparameter settings were experimented with. Once the optimal model was established, we applied distinctiveness pruning and magnitude pruning with different levels of reduction to see how much the optimal model could be compressed while mainting its performance.

2 Pruning Methods

In this work we consider two different pruning methods: distinctiveness pruning and magnitude pruning. Distinctiveness pruning [8] performs pruning of entire neurons from a layer. For each layer in the network, an activation vector is computed for every neuron. Each neuron n has a $|D|$ dimensional activation vector a_n, which is computed as

$$a_n(i) = n(x_i)$$

, where $|D|$ is the size of the training dataset, x_i is the i'th training example and $n(x_i)$ is the output of neuron n when x_i is input to the network. The neurons activation vector contains its outputs for every one of the input data points. Since our model is a convolutional architecture, the output a neuron is averaged over the spatial dimensions of the input image. Then for every pair of neurons in a layer, the angle between their activation vectors is computed and if the angle is lower than some threshold, the two neurons are considered as functionally identical so one of them is removed from the network and its weights added to the remaining one.

Magnitude pruning [9] performs pruning of individual weights from the entire network. All weights in the network are ordered by their absolute value, in increasing order. Then some proportion p of the weights are removed from the network. For example, $p = 20\%$ would remove the smallest 20% of weights in the network.

We also experiment with two different modes of pruning, re-initialisation and continuation. For re-initialisation pruning, after we have pruned the fine-tuned model we reset all of the remaining weights to have the same value of the pre-trained model, before any fine-tuning has taken place, as in [9]. After the remaining weights have been re-initialised, the network is again fine-tuned. For continuation pruning, after we have pruned the fine-tuned model we immediately continue fine-tuning it, with no re-initialisation taking place.

3 Implementation

3.1 Dataset

The SFEW contains 675 RGB images that have been manually labeled with one of seven different classes - angry, disgust, fear, happy, neutral, sad, and surprise for 6 emotions and neutral. Each class has 100 images labelled as such, except for disgut which has only 75.

Preprocessing It was observed that the images in the SFEW dataset contain not just the face but other background 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) [14]. The MTCNN successfully identified the faces in most images, but failed on 31 of them - these images were manually cropped to just include the face. All images were then resized to 224x224 pixels and normalised to have pixel intensities between 0 and 1.

Augmentation During training, images were randomly modified by applying the following augmentations: horizontally flipped with probability 0.5, randomly rotated +/- 15 degrees, a random square section with 90%-100% of the full image size is cropped out and used as input.

3.2 Evaluation

The dataset was randomly partitioned into separate training (75%), validation (15%) and test (15%) sets. When running experiments we train models on the training set for up to a maximum of 50 epochs and save the model and its performance on the validation set at the end of every epoch. The model from the epoch with the highest validation accuracy is then evaluated on the test set, and its performance reported.

3.3 Model

The model used is a Resnet-18 that has been pre-trained on the ImageNet dataset. It is 18 layers deep and designed to receive an image input of 224 by 224 RGB pixels. The model was fine-tuned on our training dataset using SGD with learning rate=0.02, momentum=0.9, batch size=64. These hyperparameters were chosen as they gave the best validation accuracy in our initial experiments. and reported a 63.36 percent average accuracy on the test data with a standard deviation of 3 percent.

4 Results and Discussion

4.1 Comparing Magnitude Pruning vs Distinctiveness Pruning

In this section, we compare the performance of the Magnitude Pruned network to the Distinctiveness pruned network in the continuation setting.

Table 1: Results of Distinctiveness Pruning with different angles in the continuation setting.

Angles	**Test Accuracies**	**Percentage Pruned**
1	64.55	0
5	61.98	2.1
10	31.68	20
15	22.38	47.8
30	20.99	68.3

The first thing we notice is the magnitude pruned network performs better (66.14 %) than the original unpruned network (63.36 %). We hypothesize that

Table 2: Results of Magnitude Pruning with different percentage of weights pruned in the continuation setting.

Percentage Pruned	**Test Accuracies**
50	66.14
90	63.17
95	65.15
99	51.48
99.9	25.94

this is due to the fact that the pruned network has smaller capacity and hence is less likely to overfit.

As per Table 2, the network pruned using Magnitude Pruning is able to reach commensurate accuracies as the original unpruned network even after 95 percent of hidden nodes have been pruned away. Whereas, when the network is pruned using Distintiveness Pruning, the network is not able to perform even after just 20 percent have been removed as per Table 1. We therefore conclude that for the SFEW dataset, Magnitude pruning is better.

4.2 Pruning with re-intialization to initial weights

In this section, we investigate the formation of winning tickets in both of these pruning methods. The pruning experiments are re-run in the re-initialization setting.

Table 3: Results of Distinctiveness Pruning in the re-initialization setting.

Angles	**Test Accuracies**	**Percentage Pruned**
1	59.21	0
5	57.82	1.94
10	34.26	19.6
15	25.94	47.63
30	22.18	68.32

As we can see from Table 4, the magnitude pruning is able to identify winning tickets of only 5 % of the original network size. However, the distinctiveness pruning shows about the same performance for both re-intialization and continuation. Since distinctiveness pruned network doesn't perform well even with 80 % of the neurons intact, we can conclude that this pruning method is not suitable for finding winning tickets. This could be due to the fact that winning tickets are not made out of entire neurons, but we will need to run further experiments in order to conclusively determine this.

Table 4: Results of Magnitude Pruning in the re-initialization setting.

Percentage Pruned	**Test Accuracies**
50	61.39
90	62.18
95	60.59
99	50.89
99.9	22.57

4.3 Comparing Random Pruned Networks to Magnitude Pruned Networks

In this section, we compare the performance of the Magnitude Pruned network to the Random pruned network, where a percentage of weights are removed randomly, in the re-initialization setting. The goal of this experiment is to demonstrate that it is the specific selection made by the pruning method that leads to winning tickets.

Table 5: Results of Random Pruning in the re-initializing setting.

Percentage Pruned	**Test Accuracies**
50	53.66
90	40.59
95	35.44
99	24.55
99.9	15.84

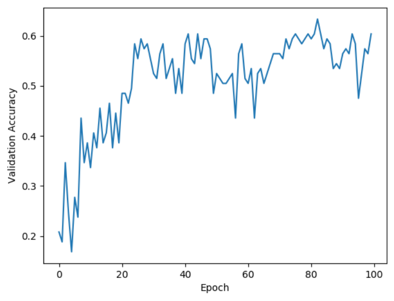
As we can see, the randomly pruned network performs consistently worse than the magnitude pruned network.

4.4 Speed of Convergence for Magnitude Pruned Networks in the re-initialization setting

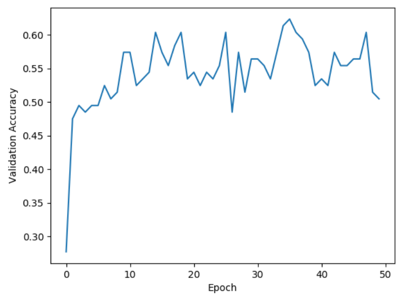
In this section, we investigate how fast the re-initialized pruned networks converge compared to the original network. From Figures 1.(a),1.(b) and 1.(c), we observe that the magnitude pruned networks are able to train faster than the original network with the same weight initialization as the original network.

5 Conclusion and Future Work

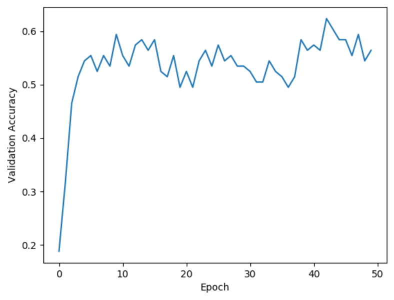
In this paper we have demonstrated that neural network pruning methods are still very effective on pre-trained and fine-tuned networks. By applying simple



(a) Validation accuracies per epoch of the original unpruned network



(b) Validation accuracies per epoch of 50 % magnitude pruned network.



(c) Validation accuracies per epoch of 90 % magnitude pruned network.

Fig. 1: Speed of convergence of original network compared with magnitude pruned networks.

magnitude pruning to a fine-tuned model and continuing to train it we were able to remove 50% of the weights in the network and significantly *increase* its test accuracy. We were able to remove as much as 95% of the weights in the network with only a marginal decrease in test accuracy. We showed that winning tickets can be found even in pre-trained models by re-running some of the crucial experiments of the original lottery hypothesis paper [9]. We found that distictiveness based pruning was by far inferior to magnitude based pruning, both in terms of resulting network performance and in identifying winning tickets.

For future work, we would like to perform a more detailed investigation of how the lottery hypothesis works in the pretrain and fine-tune setting. We will run experiments to check if magnitude pruning of a pre-trained network prunes the same weights compared to pruning the fine-tuned the network. This could provide insight on whether or not a single winning ticket is useful for multiple different tasks, or if there is a separate ticket for each possible fine-tuning task. We would also like to compare the effect of re-initialising to random values and re-running both the pre-training and fine-tuning processes. In this paper, we only experimented with re-initialising to pre-trained weights due to time constraints. It is also necessary to find out the exact reason why distictiveness pruning fails in this setting. We hypothesise it is because distictiveness pruning removes entire neurons at a time. Hence we will run experiments where entire neurons are pruned based on magnitude, to determine if it is possible to find winning tickets while only pruning entire neurons. Overall, having a better understanding of what makes a ticket winning may allow us to identify winning tickets without having to train a model to convergence. This would allow us to vastly reduce the computational burden of training networks, since we could perform pruning *before* training the model.

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.
14. Zhang, K., Zhang, Z., Li, Z., and Qiao, Y. (2016). Joint face detection and alignment using multitask cascaded convolutional networks. IEEE Signal Processing Letters, 23(10):1499–1503.