Batch pruning, a robust distinctiveness pruning method for fine-grained image classification

John Yoo

School Of Computing, Australian National University, Canberra, Australia
John.Yoo@anu.edu.au

Abstract. Fine-grained image classification is an important task with applications in tasks such as object detection and species identification. Networks trained for fine-grained image classification are large and consume exorbitant amounts of computational resources due to the nature of the task. In this paper we expand upon the usage of a feed forward neural network for this classification task using the synthetic dataset Vehicle-X. We apply network reduction and pruning methods to explore the potential for such networks to be limited in their computational expense. In this paper we propose a method as an indicator for the robustness of pruning methods and a batch pruning method to achieve more robust pruning when compared to previous research. We find that with batch pruning we can reduce the computational complexity of our model to 33.2% of its original complexity with a sacrifice of 10% accuracy.

Keywords: Neural Networks · Pruning · Fine-grained Classification · Network Reduction.

1 Background and Introduction

Artificial neural networks and deep learning have proven to be able to produce solutions to many problems in various domains due to their ability to learn high-level abstractions in data through the use of hierarchical architectures. When training such networks, often times they are over-parametrised with large numbers of redundant neurons far larger than the optimal amount. The optimal number of neurons in a neural network are difficult to determine and thus oversized networks are often used. Oversized networks are undesirable as they consume more space and require more computational resources to run. A common approach to this issue is to use methods such as Dropout [2] during training to reduce the number of neurons that influence the model's predictive ability thus aiding in preventing overfitting. Such regularisation methods, however, do not reduce the computation time during evaluation and are thus only useful during the training process.

In this paper we develop a fine-grained classification task to classify samples from the synthetic dataset Vehicle-X [1] into their 11 different vehicle types based on features extracted from a ResNet [3] model pretrained on ImageNet. Using a

feed forward neural network we utilise this fine-grained image classification task to expand upon the ideology of utilising the weight matrix for pruning in [4]. The paper concludes that the static weight matrix is insufficient to determine the functionality of hidden neurons for pruning, and so we conduct experiments while fine-tuning the network after each pruning iteration. In particular, we explore the effects of the distinctiveness pruning method as described in [4] and [5] on the task and propose a method to determine the robustness of a pruning method as well as an a method to achieve more robust pruning.

1.1 Fine-grained Classification

Fine-grained image classification focuses on differentiating between objects that have small differences and potentially a large number of similarities, e.g., species of insects, animals and car models. A key factor in fine-grained classification is for the model to learn this relevant small difference in order to successfully predict the correct classes. In order to achieve this, CNNs with many layers are often used to extract the fine details in the input images and thus, successful models are large and expensive to run. In many popular CNN architectures often included as the final layer is a fully-connected layer consisting of many hundreds or thousands of neurons for e.g. ResNet uses 1000 hidden neurons in its fully-connected layer.

2 Methodology

2.1 Data

Vehicle-X is a large-scale synthetic dataset that contains 1362 different vehicles of 3D models with an expansive list of editable attributes. The nature of the dataset allows for an unbounded number of samples to be generated. In the original paper [1] a re-id task was explored with the use of Vehicle-X synthetic data and real world data. In our task we will only consider a subset of the synthetic data and explore classification on the higher level of 11 different types of vehicles. A total of 75,516 images generated using the Vehicle-X engine are fed through a ResNet model that has been pre-trained on ImageNet. We use features extracted from the output of the global average pooling layer which total to 2048 features per image. e.g. a single row of data looks like: [0.233, 0.124, 0.394 .. ,0.542] As the data is directly extracted from the ResNet model there is no need to pre-process it further before we use it for our feed forward neural network. We split the data into Train, Validation and Test sets as described in Table 1.

Table 2 illustrates the split of class data in train, validation and test datasets. We can see that all classes are split with similar portions in each of the sub datasets.

Table 1. Split of dataset into train, validation and testing sets

Split	**# of samples**	**% of dataset**
Train	45438	60.17%
Validation	15142	20.05%
Test	14936	19.78%

Fig. 1. Samples of Vehicle-X generated images [7], note the differing vehicle types, camera angles and lighting.

2.2 Evaluation Metrics

To measure the performance of the pruning method and the pruned network we use the classification accuracy on the test set of 15,142 images as well as the number of FLOPs (Floating Point Operations) from one evaluation of the network. FLOPs give us an indicator of the pruned networks computational performance that transcends computer hardware.

2.3 Baseline Model

We first develop a base feed forward neural network to solve the classification task. From a selection of parameters we tune the model on the validation set of 14,936 images. Table 1 details the final set of hyperparameters used for the base model.

To determine convergence of the model we implement early stopping with patience = 20. For our experiments we continue pruning until there are no more excess neurons to prune from the network.

2.4 Fine-tuning

Fine tuning is a common method used in transfer learning to apply the 'knowledge' one model has attained to another model. In doing so, fine tuning can re-train the weights in the pre-trained model allowing for a quicker convergence and potential increase in accuracy. In our research we fine tune our model to re-train the weights after pruning as a way to allow the model to make up for the lost neurons and to expand upon the findings of [4].

Table 2. Split of the 11 vehicle types across the 3 sub datasets.

Class	% in Train	% in Validation	% in Test
Sedan	21.15%	20.875%	21.83%
SUV	10.95%	10.63%	11.39%
Van	4.34%	4.29%	4.22%
Hatchback	20%	19.32%	20.38%
Mpv	1.16%	0.97%	1.07%
Pickup	3.92%	3.58%	3.92%
Bus	5.60%	5.42%	5.15%
Truck	7.55%	7.87%	7.97%
Estate	4%	3.99%	4.13%
Sportscar	16.64%	17.03%	16.48%
RV	4.60%	4.64%	4.82%

Table 3. Representation of the network, the first 8 are hyper-parameters used to train our base model

Attribute	Value	Description
Learning Rate	0.01	Networks rate of learning
Hidden Neurons	500 Neurons and 1 layer	Number of layers and number of neurons in each layer
Dropout	0.2 in each layer	Percentage dropout of neurons after each layer
Epochs	500	Number of training epochs
Bias	True	Learn the bias in the network
Activation Function	ReLu	Activation function to use in the network
Batch Size	1024	Number of training examples used in one iteration
Early Stopping Patience	20	Number of epochs to wait for validation loss decrease
Input Size	2048	Input size of each feature extracted image
Output Size	11	Output size equal to the number of output classes

2.5 Pruning

Pruning is a concept to reduce the size of a neural network through compression. The importance of each neuron is determined through some ranking method and the least important neurons are removed. This method of network compression was first explored by LeCun et al. in [6] where network compression resulted in better generalisation, fewer training examples required and improved speed of classification.

Distinctiveness pruning is a method whereby the least distinct redundant neurons are pruned. The conceptual idea is that we can calculate the angle between two output vectors of neurons and determine their degree of similarity. In the original paper [4][5] neurons with output vectors 15 degrees are considered similar, one is added to the other and removed, at the same time where the angle is over 165 degrees the 2 neurons are regarded as complementary and so both are removed. As the model loses redundant neurons the model is compressed and results in a smaller model and improved computational speed for post-hoc evaluation.

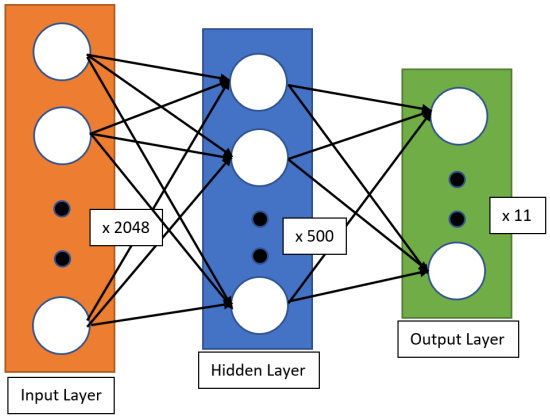

Fig. 2. Illustration of our base model containing an input layer of (1,2048) 1 hidden layer with 500 neurons and an output layer of (1,11)

To compute the distinctiveness of neurons we first normalise neuron output vectors to fit the range of 0-180 (1) and calculate the angle between each neuron pair as in equation (2).

$$\mathbf{x} = \mathbf{x} - mean(\mathbf{x}) \tag{1}$$

$$\cos(\mathbf{x}, \mathbf{y}) = \frac{\mathbf{xy}}{\|\mathbf{x}\|\|\mathbf{y}\|} = \frac{\sum_{i=1}^{n} \mathbf{x}_i \mathbf{y}_i}{\sqrt{\sum_{i=1}^{n} (\mathbf{x}_i)^2} \sqrt{\sum_{i=1}^{n} (\mathbf{y}_i)^2}} \tag{2}$$

For our research purposes as we are exploring the task of fine-grained image classification the feature space is magnitudes larger than of those tasks explored in [4][5]. As a result, the number of hidden neurons required to train a base model to learn the representations in the data are also high. Upon initial experimentation of the base model, we find that there are no neurons with angles less than 15 or more than 165. Thus, we consider the following approaches:

1. Iterative removal of distinctively similar and complementary neurons with no bounds. We remove one of the similar neurons and one of the complementary neurons as we have modified the bounds.
2. Iterative removal of only 1 distinctively similar neurons with no bounds.
3. Batch removal of distinctively similar neurons, whereby we select X number of similar neurons and remove all but 1 and add the total of the removed neurons to the remaining one.

Following the basic outline of the flowchart in Fig.3 we:

1. Train the initial base line feed forward neural network with tuned hyper-parameters until convergence.
2. Apply pruning operations to the network.
3. Fine-tune the network with the updated weight matrix until convergence.
4. Repeat steps 2 and 3 until we are done pruning.

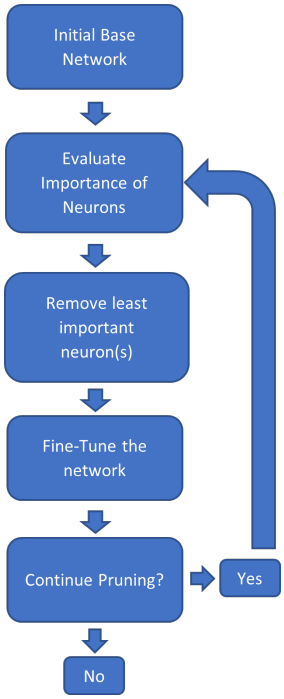

Fig. 3. Flowchart of iterative pruning process

3 Results and Discussion

We obtain a final test accuracy of 46.988% using our base model, comparing this to the 1362 class classification problem in the original Vehicle-X paper which achieves 94.34 we can see a significant difference. The low accuracy of our model can be explained by the significantly reduced size of the dataset used to train the model whereby we use 45,438 images as opposed to the over 300000 in Vehicle-X.

We apply each pruning method following the overall methodology in Fig.3 and iteratively prune until there are no more extra neurons to prune. The accuracy against the testing set is plotted against the number of pruning iterations.

In Fig.4 we observe a common trend of extreme decrease at some point in testing accuracy in all 3 methods which is expected as the process would eventually remove important neurons. After this inflection the models start to seemingly randomly guess, resulting in a somewhat flat line. An interesting observation can be seen at which point in the pruning process each method experiences the inflection. This point of inflection can be noted as the point at which the model loses the majority of the neurons required to successfully predict. As such we can use this as an indicator for the robustness of each pruning method. Note that single similarity pruning removes 1 neuron in each pruning cycle as opposed to the 2 in the methods.

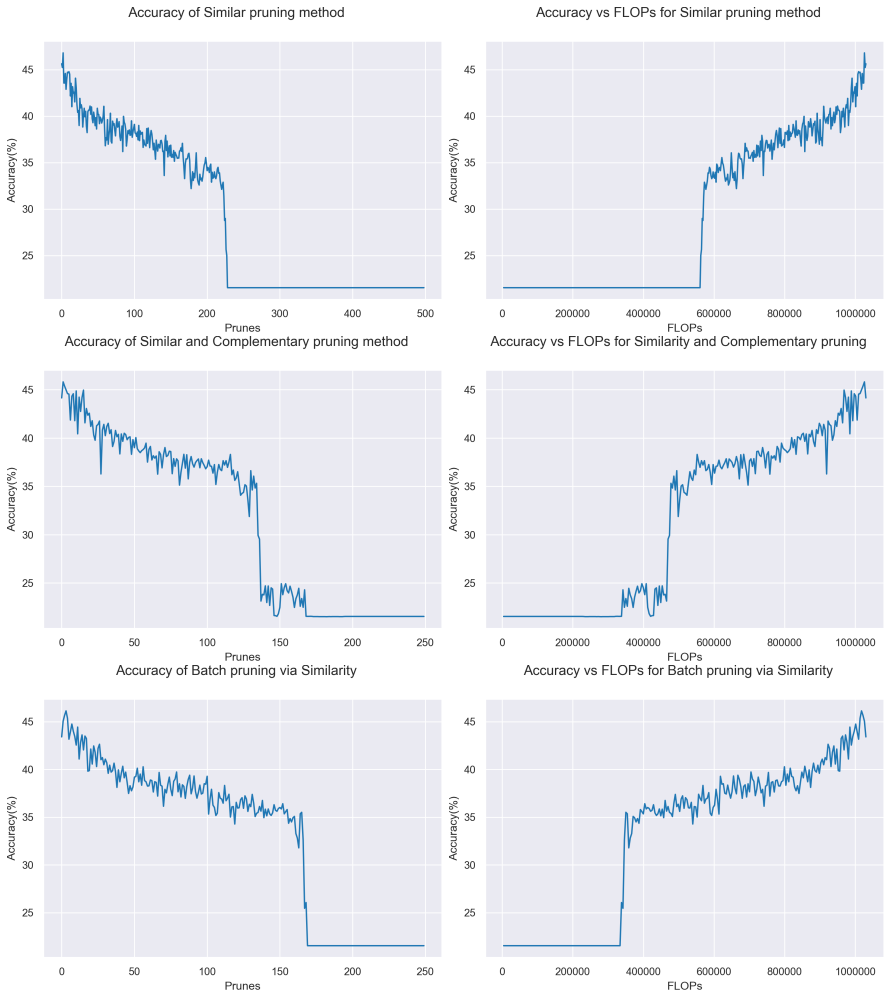

Fig. 4. Plots of each method against both accuracy over neurons pruned and total FLOPs

Table 4 shows the exact stage at which each model experiences the inflection and also shows how many FLOPs can be reduced before the model reaches the inflection point. We note that the batch pruning of 2 similar neurons from a set of 3 is the most robust with 688374 FLOPs able to be reduced from the 1030500 from the base model before the inflection point. In this respect using the batch pruning method with a loss of 10% testing accuracy we can reduce the network to 33.2% of its original computational expense at evaluation.

Table 4. Number of neurons pruned and maximum FLOPs reduced before the inflection point for each method

Method	**# of neurons pruned**	**FLOPs**
Base Model	0	1030500
Most similar	224	461664
Most Similar and Complementary	270	556470
2 Most Similar	334	688374

4 Conclusion and Future Work

In this paper we have explored a fine-grained image classification problem using features extracted from ResNet on the Vehicle-X dataset. As opposed to the 1362 class classification task in the original Vehicle-X paper we reproduce a higher level 11 class vehicle type classification problem as we use a smaller subset of data to train our feed forward network. We implement an iterative pruning method to explore the effects of different pruning approaches on the testing accuracy and computational resources required. In particular, we explore the distinctiveness pruning method [4][5] and apply the removal of least distinct neurons via 3 methods. Observations reveal an inflection point that we propose can be used to indicate the robustness of a pruning method. We find that batch pruning removes the most neurons before the inflection point as compared to traditional methods of similarity and complementary angle removal. We also find that the test accuracy of the model can be maintained to a similar degree while reducing the number of FLOPs. In particular, with the batch pruning method with a sacrifice of 10% accuracy we can reduce the evaluation computational expense to 33.2% of its original size. In the future we will apply this method onto different fine-grained image classification datasets to verify and observe further effects. An extension to the pruning methods used will also be implemented including: badness and sensitivity measures to further explore the effects on our task. In this paper we have explored batch pruning for a set of 3 neurons with the pruning of 2, further experimentation with larger batch values will need to be conducted as well. As we have only used a subset of the Vehicle-X dataset in this paper we will utilise a similar subset of data as the original paper to produce comparable testing accuracy results. On top of this, we plan on using different

CNN architectures other than ResNet to explore implementations of the final convolutional layer that feeds into the fully connected layer.

References

1. Yao, Y., Zheng, L., Yang, X., Naphade, M., & Gedeon, T. (2019). Simulating content consistent vehicle datasets with attribute descent. arXiv preprint arXiv:1912.08855.
2. G. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. http://arxiv.org/abs/1207.0580, 2012.
3. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. arXiv preprint arXiv:1512.03385, 2015a.
4. Gedeon, T.D.: Indicators of hidden neuron functionality: the weight matrix versus neuron behaviour. In: Proceedings 1995 Second New Zealand International Two-Stream Conference on Artificial Neural Networks and Expert Systems. pp. 26-29. IEEE (1995).
5. Gedeon, T., Harris, D.: Network reduction techniques. In: Proceedings International Conference on Neural Networks Methodologies and Applications. vol. 1, pp. 119-126 (1991)
6. LeCun, Y., Denker, J. S., & Solla, S. A. (1990). Optimal brain damage. In Advances in neural information processing systems (pp. 598-605).
7. Yao, Y., Zheng, L., Yang, X., Naphade, M., & Gedeon, T. (2019). https://github.com/yorkeyao/VehicleX