UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH

Facultat d'Informàtica de Barcelona

Master in Innovation and Research in Informatics

Algorithm Data Mining

Comparing performance of two pruning on Fashion- MNIST data set

Third Assignment

Mohana Fathollahi

June 1, 2022

1 Introduction

In this assignment, two approaches of pruning neural network have been applied on Fashion-MNIST data set. In the following, brief description about two type of pruning has been provided and at the end implementation of these techniques on data set has been discribed.

2 Prunning

The goal of neural network pruning is to convert a large network to a smaller network but without considerable effect on accuracy. It helps to have a faster neural network and make it more efficient [1]. To prune a network we can delete some parameters to enhance efficiency while we are trying to keep accuracy. At the end of pruning we will have not fully connected neural network like figure 1.

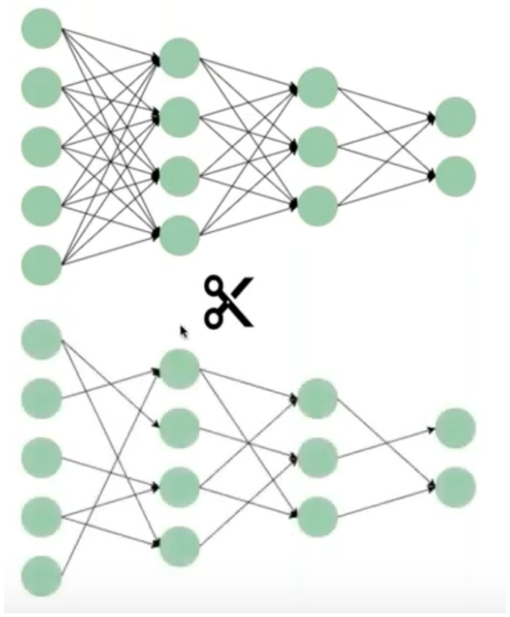


Figure 1: result of pruning

The question can be raise is that why pruning is useful in real cases? Efficient model is a model that optimizes memory usage and performance at the inference time[1] Therefore we need to decrees computing cost to save memory and time and one possible solution is using pruning.
Steps that we should do to prune neural network:

- Determining the significance of neurons or weights.
- Prioritizing the neurons based on their value.

[1]Inference time is the amount of time that we need to process test data and make a prediction, but it excludes training time.

- Removing the neurons that are the least significant.
- Determine whether to prune further based on different conditions (defined by the user).

Two groups of pruning described in below and in the figure 2 we can see clear difference between these approaches;

- Unstructured: In this approche we zeroing out the indivitual weights, but the problem of this approach is that; it does not speed up the process
- Structured: pruning group of weights, for example; deleting entire neurons, filters, or channels.[1]

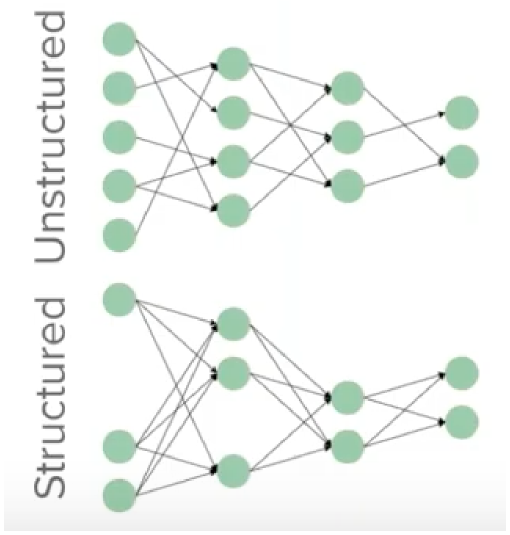


Figure 2: Structured vs Unstructured pruning

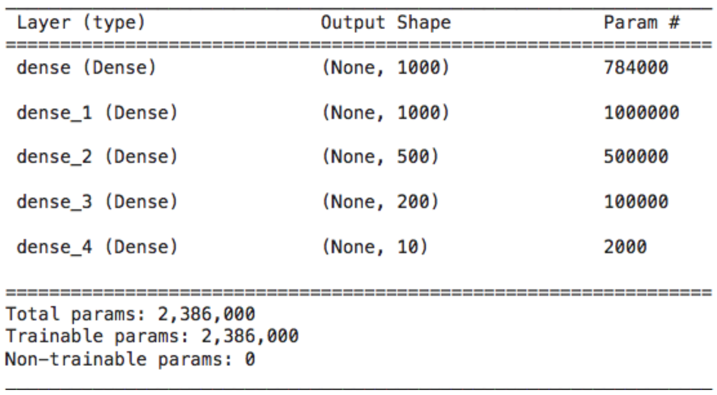
3 Dataset

In this study, Fashion-MNIST has been used. In this dataset, In this data set we have 10 classes, such as; 'T-shirt', 'Trouser', 'Pullover' and so on. Our train data set has 60000 items and our test set has 10000 items. Items refer to images of T-shirt, Trouser, Pullover and so on that we should classify them. In the first step we should reshape images from 28*28 to 784.

3.1 Building Neural Network

Five dense layers have been used to train the model, the summary of model has been provided in figure 3. The first dense layer has 1000 nodes and because we have 784 features in input layer total parameter of this layer will be 1000*784=784000, for other dense layers number of neurons have been shown and number of parameters have calculated like above. Additionally, relu applied as an activation function for hidden layers and in the last layer softmax as an activation function used, because we should categorized images in 10 groups.

After building the network, we need to save model while it trains. By using ’Callback’ and ’ModelCheckpoint’utilities of Keras, we can save the model with the best weights. It checks if the performance of model with updated weights after every epoch is better than the performance of the saved model.[2]



```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 1000)              784000

 dense_1 (Dense)             (None, 1000)              1000000

 dense_2 (Dense)             (None, 500)               500000

 dense_3 (Dense)             (None, 200)               100000

 dense_4 (Dense)             (None, 10)                2000

=================================================================
Total params: 2,386,000
Trainable params: 2,386,000
Non-trainable params: 0
_________________________________________________________________
```

Figure 3: Network Structured

Based on figure 4, we can compare performance of model in train and validation set. When number of epochs increased training accuracy started to increase while validation accuracy started to decrease, that there is a sign of overfiting and our model does not learn the data, it memorizes the data.[3] Therefore increasing epochs more than this will not be a good idea.

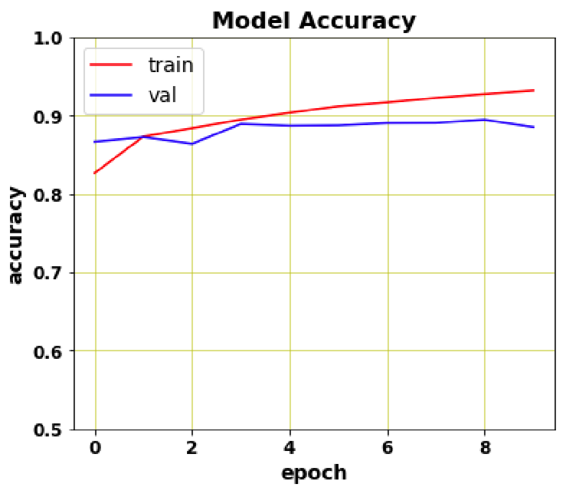


Figure 4: Comparing validation and test accuracy

3.2 Pruning techniques

In this study, wight pruning and neuron pruning respectively as structured and unstructured pruning techniques have been applied on data set and their performance compared together.

3.2.1 Weight Pruning

For weight pruning, a dictionary of the weights have been created and key of this dictionary has a three valued tuple like this; (layer_number, row of weight matrix, column of weight matrix) and its value is weight.
Row number is the neuron number from previous layer, and each column number is the neuron number of that particular layer. For example in figure 5 we can see that number of rows represent number of nodes in input layers and number of columns represent number of number of neurons in hidden layer.[4]

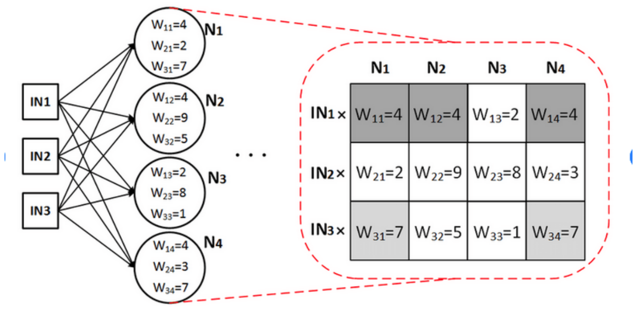
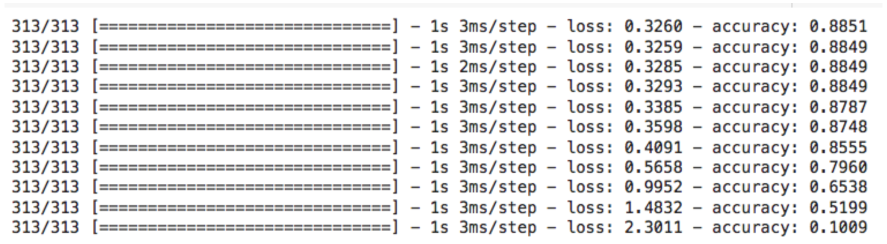

IN1
IN2
IN3
$W_{11}=4$ $W_{21}=2$ $W_{31}=7$
N1
$W_{12}=4$ $W_{22}=9$ $W_{32}=5$
N2
$W_{13}=2$ $W_{23}=8$ $W_{33}=1$
N3
$W_{14}=4$ $W_{24}=3$ $W_{34}=7$
N4
N1 N2 N3 N4
IN1 × $W_{11}=4$ $W_{12}=4$ $W_{13}=2$ $W_{14}=4$
IN2 × $W_{21}=2$ $W_{22}=9$ $W_{23}=8$ $W_{24}=3$
IN3 × $W_{31}=7$ $W_{32}=5$ $W_{33}=1$ $W_{34}=7$

Figure 5: Simple weight matrix

All weights that we have in this network is 2,384,000 and after creating this dictionary, we should sort values or weights in ascending order. In the next step some of these weights that have less values should be removed. To know how many of weights should be zero out, different values have been used that are gathered in list $k : [20, 30, 40, 50, 60, 70, 80, 90, 99]$. After pruning with each of these values, we have a new model with new weights and we need to analyze performance of this model on test set for different percentage of pruning. Based on figure 6 when percentage of pruning increases accuracy will decrease, until 30% pruning, accuracy did not change a lot so we can say that 30% of weights are useless.



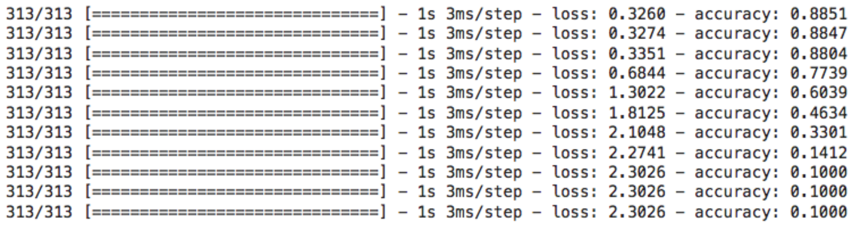
```
313/313 [==============================] - 1s 3ms/step - loss: 0.3260 - accuracy: 0.8851
313/313 [==============================] - 1s 3ms/step - loss: 0.3259 - accuracy: 0.8849
313/313 [==============================] - 1s 2ms/step - loss: 0.3285 - accuracy: 0.8849
313/313 [==============================] - 1s 3ms/step - loss: 0.3293 - accuracy: 0.8849
313/313 [==============================] - 1s 3ms/step - loss: 0.3385 - accuracy: 0.8787
313/313 [==============================] - 1s 3ms/step - loss: 0.3598 - accuracy: 0.8748
313/313 [==============================] - 1s 3ms/step - loss: 0.4091 - accuracy: 0.8555
313/313 [==============================] - 1s 3ms/step - loss: 0.5658 - accuracy: 0.7960
313/313 [==============================] - 1s 3ms/step - loss: 0.9952 - accuracy: 0.6538
313/313 [==============================] - 1s 3ms/step - loss: 1.4832 - accuracy: 0.5199
313/313 [==============================] - 1s 3ms/step - loss: 2.3011 - accuracy: 0.1009
```

Figure 6: Accuracy of test set after Weight Pruning

3.2.2 Neuron Pruning

In this type of pruning we focus on entire neurons. Like weight pruning, we should create a dictionary but key of dictionary is different compare to previous case. It has two values in tuple like this; (layer_number, column of weight matrix). Why column number? because as mentioned before column number in weigh matrix is the neuron number of that particular layer and we should consider weights that point out to these neurons.

Based on network that has been built, we have 2,700 neurons, therefore we will have 2700 $(key, value)$ in dictionary. In first step, we sorted wights based on l2 norms and in ascending order. Then, like previous approach we need to apply different percentage of pruning on wights with less values. At the end, we will have new model with new weights that will have different performance on test set. In figure 7 various accuracy for different pruning percentage has been provided, and like previous case when pruning increased accuracy decreased.



```
313/313 [==============================] - 1s 3ms/step - loss: 0.3260 - accuracy: 0.8851
313/313 [==============================] - 1s 3ms/step - loss: 0.3274 - accuracy: 0.8847
313/313 [==============================] - 1s 3ms/step - loss: 0.3351 - accuracy: 0.8804
313/313 [==============================] - 1s 3ms/step - loss: 0.6844 - accuracy: 0.7739
313/313 [==============================] - 1s 3ms/step - loss: 1.3022 - accuracy: 0.6039
313/313 [==============================] - 1s 3ms/step - loss: 1.8125 - accuracy: 0.4634
313/313 [==============================] - 1s 3ms/step - loss: 2.1048 - accuracy: 0.3301
313/313 [==============================] - 1s 3ms/step - loss: 2.2741 - accuracy: 0.1412
313/313 [==============================] - 1s 3ms/step - loss: 2.3026 - accuracy: 0.1000
313/313 [==============================] - 1s 3ms/step - loss: 2.3026 - accuracy: 0.1000
313/313 [==============================] - 1s 3ms/step - loss: 2.3026 - accuracy: 0.1000
```

Figure 7: Accuracy of test set after Neuron Pruning

4 Result of model

In this part compassion between two pruning will provided, based on figure 8 with pruning 20% of neorons we can keep accuracy near to 90% while when we pruned weights until 50% we could keep accuracy near to 90%. Additionally, in neuron pruning after 20% pruning we have sharp decrease in accuracy, while this sharp decrease happen in weight pruning near 90% pruning.
Even though with high percentage of pruning, we could get smaller and more efficient network, but we have less accuracy. On the other side, we could prune with lower percentage and have high accuracy but not as efficient as previous case. Therefore, there is a trade off between model accuracy and efficiency.[5]

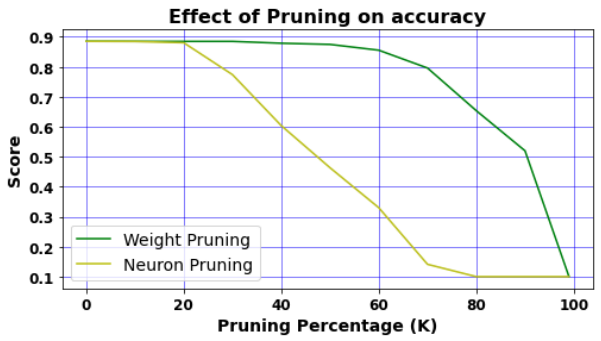


Figure 8: comparison between Neuron pruning and wight pruning

References

[1] *Pruning structure*. URL: https://analyticsindiamag.com/a-beginners-guide-to-neural-network-pruning/.

[2] *checkpoint*. URL: https://keras.io/api/callbacks/model_checkpoint/.

[3] *Overfiting*. URL: https://datascience.stackexchange.com/questions/27561/can-the-number-of-epochs-influence-overfitting.

[4] *weight matrix*. URL: https://www.researchgate.net/figure/A-simple-neural-network-and-the-mapping-of-the-first-hidden-layer-onto-a-43-Weight_fig2_292077006.

[5] *Pruning trade off*. URL: https://towardsdatascience.com/pruning-neural-networks-1bb3ab5791f9.