Applying Pruning Filters In Convolutional Neural Network

Vincent Martineau

Department of Computer Science and Software Engineering, Université Laval

Introduction

It was shown in P.Molchanov et al. that it was possible to prune a convolution neural network and still have decent results. However the proposed solution might require a lot of iterations. In the case we need more complex model to run on smaller device(Jetson) and prototype quickly there are some properties that seems consistent and that could improve workflow.

Related work :

- P.Molchanov et al. (2017) : Pruning Convolutional Neural Networks for Resource Efficient Inference.
- H.Li et al. (2017) : Pruning Filters for Efficient ConvNets.

Goals :

- Reduce training time to produce sufficient network.
- Provide a module that could handle multiple models.
- Explore the effect of pruning for speed and size.

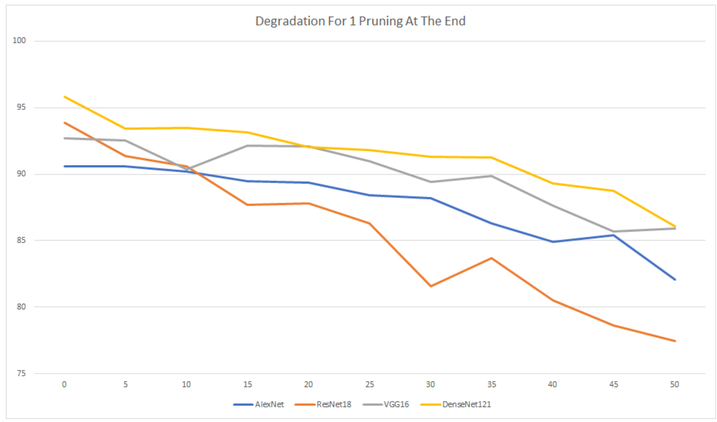
Algorithm

- Do initial training (not complete)
- **Prepare Pruning**
  - Convert model to ONNX
  - Extract execution graph and Determine which layer can be pruned
- **Prune network (iterative process)**
  - Find number of filters to prune
  - Rank filters and remove filter with the lowest rank
  - Apply pruning effect to next layers
  - Reset optimizer
- Finalize training

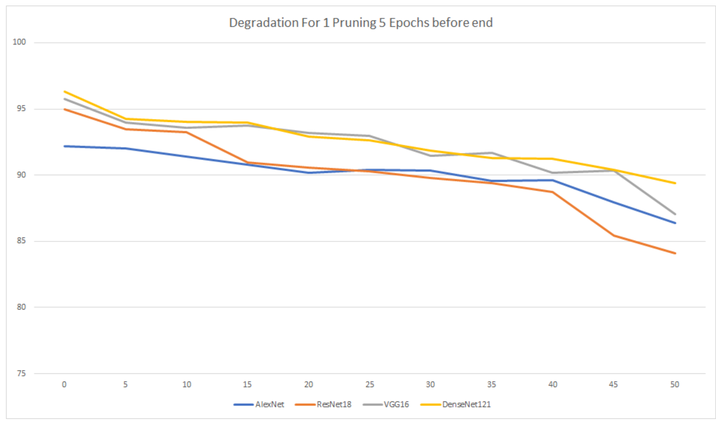
Settings

- **Dataset** : Cifar10
- **Optimizer** : Stochastic Gradient Descent
- **Learning Rate** : 0.01
- **Momentum** : 0.0
- **Nesterov** : False
- **Batch Size** : 64
- **Use GPU** : Of course!

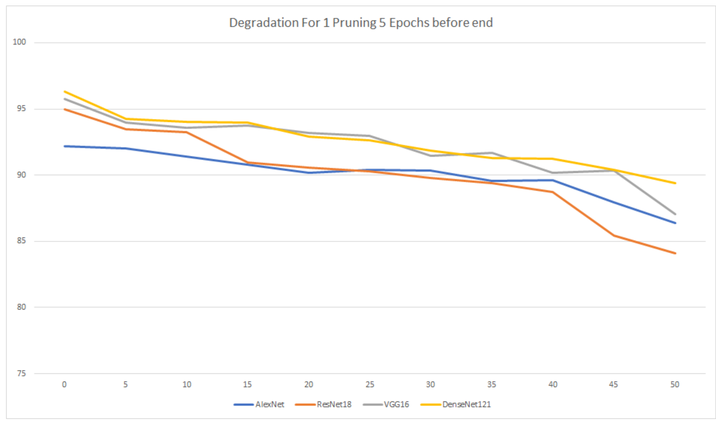
Compare Timing Strategy For Pruning



The first thing we realize is that if we wait until the end to start our pruning we may expect a greater degradation of our results. Also we see that the result contains some spikes. This seems to



When doing 1 step of pruning once the model start to stabilize will produce consistently higher results. Also the difference between model is a lot less important across the range.



This is a very long line of text

Observations

- Not all convolutional layer can be pruned. Pruning layers before a residual connection is dangerous because both side of the residual connection must have the same side.
- When pruning in a convolution layer it is important to propagate to the following layers so the next layers have the right input size. This apply to convolution, linear and batchnorm layers..
- It is been seen that algorith leave only one filter on a layer. When this happen it reduce the quality of the results.
- When pruning it is important to reset optimizer.

Conclusion

Discussion :

- It is a **possible** to support multiple model type using the same module.
- The reduction on some model is **impressive** and could be run on lower trier hardware.

Future Works :

- Pruning proved to be a valid form of regulation.
- Would be possible to use different criteria to sort filters.
- Experiments were made on Cifar10. It would be interesting to try on various datasets.