CS230

Pruning for Efficient Road Segmentation

Anjali Roychowdhury
Department of Mechanical Engineering
Stanford University
aroyc@stanford.edu

Justin Dieter
Department of Mathematics
Stanford University
jdieter@stanford.edu

Chandler Watson
Department of Computer Science
Stanford University
watsonc@stanford.edu

Abstract

In this project, we explored applications of structured and unstructured pruning on a road segmentation network for fast and lightweight autonomous vehicle applications. While road detection is a recurrent challenge for many autonomous vehicle applications with a high potential for impact, significant computational complexity and a lack of sufficient labeled data have made it one of the more challenging problems in the field. For our project we extended Marvin Teichmann's KittiSeg, based on a FCN8-VGG16 model, to allow for two styles of network pruning prominent in the literature. In addition, an attempt at layer-wise pruning was made, using a genetic algorithm on a set of binary vectors to evolve proper layer drops. Despite the resilience of neural networks demonstrated by Mittal et al. to random pruning, this appeared to cause learning to stagnate. Unstructured pruning, however, worked well, and a well in-progress structured pruning extension was created as well. Both are currently implemented at `https://github.com/watsoncm/PruneSeg`.

1 Introduction

Our group applied deep learning to road image segmentation to detect driving lanes, as if we were writing the software to process the data taken by a camera on top of an autonomous vehicle. Despite the prevalence of the need for road segmentation in the autonomous vehicle industry and the high-potential for impact in improving autonomous performance and safety, there are yet to be highly effective, efficient, and reliable solutions. Furthermore, in addition to significant computational complexity and difficulty in generating labeled data, this application is further complicated by the number of conditions under which it must perform well, including partial occlusions, shadows, and a variety of weather conditions.

For this project, our group extended Marvin Teichmann's KittiSeg open-source framework using the KITTI Road Detection dataset [4] presented with the benchmarks given by the paper [2]. Random patch sampling is performed to obtain 256x256 pixel inputs to the network. An FCN8-VGG16 model then encodes it and upsamples it via transpose convolutions into a block with one softmax output per original pixel. Thus, we output a classification of "road" or "not road" on a pixel by pixel basis for the entire image and calculate our evaluation metrics – specifically average precision and maximum F1 score – from these pixel classifications.

CS230: Deep Learning, Winter 2018, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

2 Related work

Efficient deep models for monocular road segmentation [6]

This paper explores possible modifications to existing road segmentation networks to use less parameters and operate more efficiently while using innovative techniques to reduce the number of filters in the later layers of the network. This ensured that it operated more efficiently and required less compute time to perform the segmentation.

Structural Compression of Conv. Neural Networks Based on Greedy Filter Pruning [1]

This paper attempts to deduce an algorithm to eliminate filters within the convolutional network to damage performance the least. It assigns a rank to each filter in their importance in the network and the selectively prunes the filters that are least necessary. This approach causes significant boosts in efficiency since it prunes entire kernels in the model but accordingly degrades accuracy significantly.

To prune, or not to prune: exploring the efficacy of pruning for model compression [7]

This was the paper that gave the method we applied to our model for selective pruning. It surveyed the possibility of using pruning of neurons to compress arbitrary neural networks. It gave precise formulas for how and when pruning should be applied to ensure that the model learns around the pruning. Since this approach only prunes individual neurons it should give only minor reductions in accuracy but should give a slight performance boost with fast sparse matrix multiplication and greatly improve memory efficiency.

Pruning Convolutional Neural Networks for Resource Efficient Inference [5]

This approach is similar to [1] but it uses a Taylor expansion with loss gradients to approximate the importance of each convolutional kernel. Other methods only used first-order approximations, so this method is able to outperform them. Since this method still prunes entire kernels it also yields significant boosts in performance at the cost of non-negligible deductions in accuracy.

3 Dataset and Features

This project worked from the KITTI Road Detection Dataset, which is comprised of 579 1242x375 pixel color images of roads in a variety of locations, lighting conditions, and weather, which are labeled with ground truth color maps (Figure 1). The original dataset contains 289 train images and 290 test images, the former of which was pre-divided by KittiSeg into a 241/48 train/validation split.

As we were working with a remarkably small dataset of under 300 total images and were attempting to achieve ambitious F1 scores, pre-processing and data augmentation were crucial to generate good results. As provided by the framework, pre-processing consisted of random patch training on 256x256 pixels with random cropping probability of 0.8.

4 Methods

4.1 FCN8-VGG16 Model

The main model we worked with and attempted to modify was the FCN8-VGG16 model implemented by KittiSeg. A typical VGG16 model is a network with convolutional and max pooling layers followed by fully connected layers as shown in Figure 1. However, in the FCN8-VGG16 model, the fully connected layers are replaced with convolutional layers to avoid losing spatial data. Finally, upsampling with transpose convolutional layers is used to obtain a final layer with the input dimensions of the input image. Each pixel is then processed through a softmax to output their respective segmentation class probabilities. This model is trained with a standard pixel-wise cross-entropy loss function.

4.2 Genetic Algorithm for Layer Trimming

The goal of our project is to extend the FCN8-VGG16 model to operate more quickly. We hypothesized that layers could potentially be removed from the model and near equal performance could be

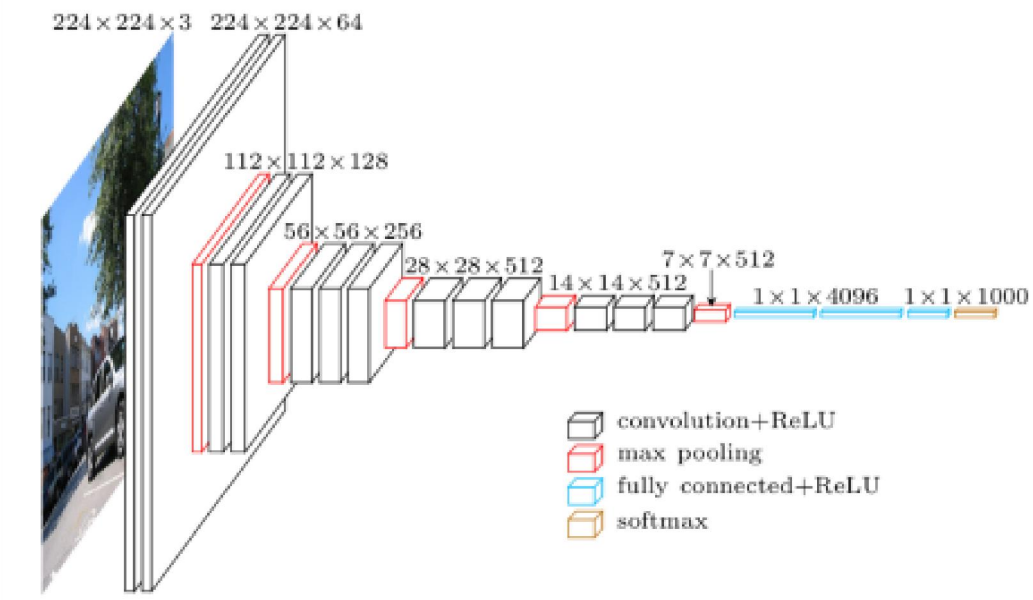

Figure 1: The structure of a standard VGG16 model [3].

achieved as the layer after the pruned one learned how to use the higher-level features. To account for shape changes, we manually spliced in tf.Tile operations wherever a change of shape necessitated it.

The challenge is then figuring out which are the optimal layers to removed so that performance can be boosted while still maintaining a high F1 score. To do this we used a simple genetic algorithm with crossover and mutation on a set of five binary vectors describing which layers to keep/drop. Crossover was modeled as a set union with a given drop probability, and mutation was given by randomly toggling 1-5 bits. The fitness function was given by the F1 score if the model outperformed the baseline, and negative infinity if it didn't.

4.3 Model Pruning

Our final approach was utilizing a more precise pruning method given by [7], as described in the related work section. We took the model pruning functionality provided in `tf.contrib` and extended KittiSeg to use it, allowing for sparser kernels and thus less storage-intensive networks. We train for 1500 steps with a target sparsity for each convolutional layer of 0.5 and 0.75, along with a baseline of 0.0. All other hyperparameters were kept the same. An RoC curve was generated using a custom tweak to the KittiSeg code to dump precision and recall values at various thresholds.

Additionally, the framework given by [7] in `tf.contrib` was leveraged to create a structured pruning setup as in [5]. Although this wasn't finished, it provided a strong baseline to work off of and could quickly result in a pruning framework for KittiSeg that would more readily lend itself to runtime efficiency (rather than just storage efficiency).

5 Experiments/Results

In our project, we implemented three different models: Model 1, our baseline model; Model 2, a model with dropped layers selected by a genetic algorithm; and Model 3, a model with pruned neurons. The second and third models were hoping to match or improve the accuracy of the first model while improving the eventual runtime.

Figure 2 gives an example of output from our pruned segmentation model and Figure 3 gives an RoC curve for our final pruning model. Qualitatively, it can be seen to identify the road almost perfectly despite the neurons pruned from the model, with the exception of some error on the right side of the

Figure 2: Output of our pruned segmentation model.

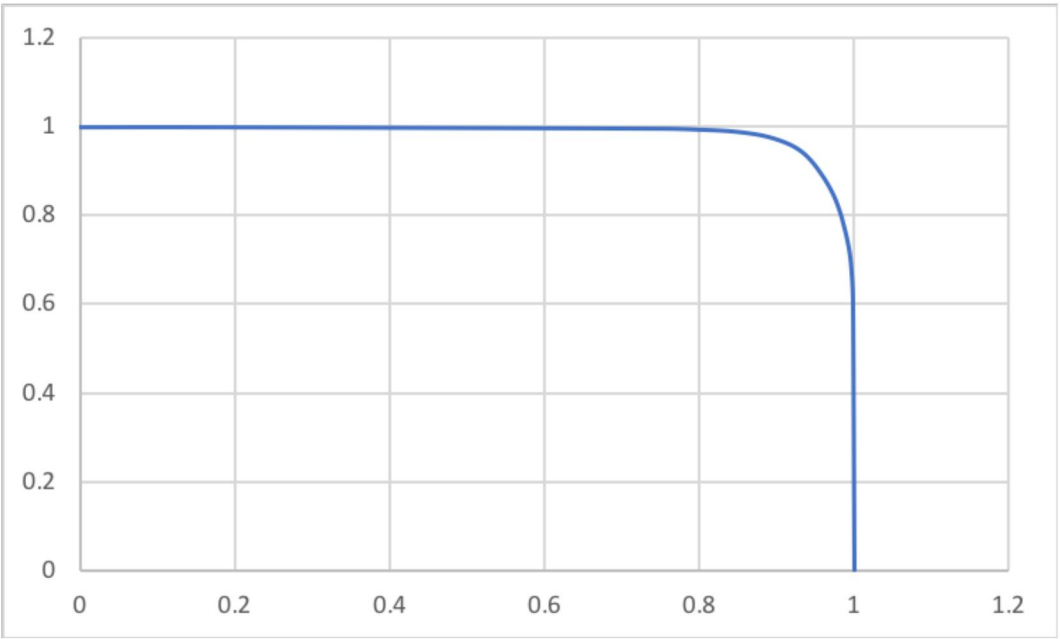

Figure 3: An RoC curve for our final pruning model.

image. Below is a table of the metrics recorded for each pruned model, with 0% sparsity being the baseline:

Sparsity	Average Precision	F1 Score
0%	92.1684	94.5185
50%	91.8006	93.8772
75%	91.8063	92.8530

The model produced from our genetic algorithm failed to produce a good F1 score, yielding an F1 score of only 33.3214. We believe this is because the model requires every convolutional layer and is unable to function with layers trimmed. However, the genetic algorithm approach may still be applicable in future work when applied to deciding which filters or neurons to prune but entire layers cannot be pruned.

6 Conclusion/Future Work

Ultimately, performing pruning as in [7] proved to be the most successful. However, to achieve speed with unstructured pruning, we would need to implement efficient sparse convolution. Nonetheless,

we did get a huge benefit in model size and memory efficiency which is very beneficial for building embedded systems that could actually be used in an autonomous vehicle.

Our approach using a genetic algorithm ultimately proved to be unsuccessful. Dropping layers from the model resulted in huge drops in performance, as later layers have no information on how to interpret the activations of earlier ones. However, the approach with a genetic algorithm could be interesting to apply to the pruning approach where a genetic algorithm is used to select which neurons to prune. Furthermore, random search approaches with stronger theoretical guarantees such as the cross entropy method could be explored instead of a genetic algorithm if random search methods are to be applied to the pruning approach.

Overall, we explored several methods and successfully found an approach to producing a pruned model that produces almost equal F1 score to the baseline model for road segmentation. With fast sparse matrix multiplication implemented this model should show a significant jump in performance to the original baseline and already shows significant improvements in memory efficiency. Beyond this, further progress on implementing structured pruning could result in massive improvements in both storage and runtime efficiency.

7 Contributions

Chandler Watson - Worked on model pruning and contributed widely to other areas of the code.
Anjali Roychowdhury - Contributed to the genetic algorithm and did a significant portion of the writing tasks.
Justin Dieter - Worked on evaluation of models and did a significant portion of the writing tasks.

References

[1] R. Abbasi-Asl and B. Yu. Structural compression of convolutional neural networks based on greedy filter pruning. *CoRR*, abs/1705.07356, 2017.

[2] J. Fritsch, T. Kuehnl, and A. Geiger. A new performance measure and evaluation benchmark for road detection algorithms. In *International Conference on Intelligent Transportation Systems (ITSC)*, 2013.

[3] D. Frossard. Vgg in tensorflow, 2016. [Online; accessed March 22, 2018 at https://www.cs.toronto.edu/ frossard/post/vgg16/vgg16.png].

[4] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The kitti dataset. *International Journal of Robotics Research (IJRR)*, 2013.

[5] P. Molchanov, S. Tyree, T. Karras, T. Aila, and J. Kautz. Pruning convolutional neural networks for resource efficient inference. 2016.

[6] G. L. Oliveira, W. Burgard, and T. Brox. Efficient deep models for monocular road segmentation. *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4885–4891, 2016.

[7] M. X. Zhu and S. Gupta. To prune, or not to prune: exploring the efficacy of pruning for model compression. *CoRR*, abs/1710.01878, 2017.