Implementing and Evaluating SqueezeNet

Caitlin McElwee
August 2018
Brown University and Distributed Research Experience for Undergraduates

Abstract

The project was part of a larger encompassing project in computer vision. The purpose was to conduct a study to evaluate the accuracy of SqueezeNet, a neural network architecture, for drawing and labeling bounding boxes on photos of cluttered surfaces. The end goal was to determine if SqueezeNet would perform on par with or better than other tested architectures (AlexNet, VGG, ResNet). After pursuing multiple routes, SqueezeNet was not able to be implemented and therefore could not be evaluated against the other networks. Future plans include finding other routes to implement SqueezeNet and eventually evaluating it.

1. Introduction

For a robot to be able to look at a table and pick up an object, several stages of analysis need to happen. We first focus on the difficulty of identifying said object. One cannot tell a robot "this is a Tide bottle" and expect it to understand: it must learn what a Tide bottle is through training a mathematical model through machine learning. The lab's previous work involves training neural networks based on the Faster R-CNN architecture (e.g AlexNet, VGG16, ResNet50, etc) to be able to draw bounding boxes around desired objects in photographs of a cluttered table, as shown in Fig 1.

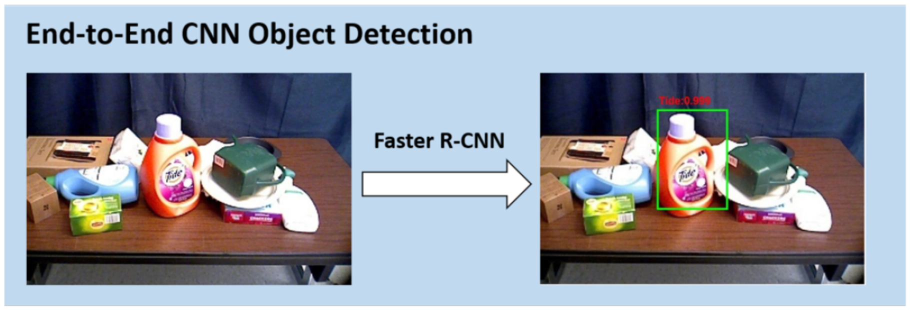

Fig 1: The neural network takes an image as input and gives as output bounding boxes for detected objects; details include coordinates for two opposing corners, the detected object, and certainness of the guess. Picture from [æ].

They all performed admirably, but more networks can always be evaluated. This project focused on implementing SqueezeNet, a neural network architecture that is smaller and less complex than other network architectures (and therefore more suitable for mobile applications). My goal was to evaluate its performance against the other networks.

SqueezeNet is notable for its small size: at 10MB of storage needed for a model, it is 50x smaller than AlexNet [1]. Despite its much smaller size, SqueezeNet is known to perform on par with AlexNet, hence the comparison. If SqueezeNet could be implemented with the lab's setup, it would potentially allow running the network onboard the robot instead of an offboard computer which could not be done with other networks due to size constraints.

2. Related Work

Object Detection in Photographs of Cluttered Surfaces

As previously mentioned, this project was part of an overarching project of the lab to evaluate neural networks on object detection for the lab's needs. As presented in [æ], the networks AlexNet; VGG11, -13, -16, and -19; ResNet18, -34, -50, -101, and -152; and MobileNet were evaluated for accuracy on the lab's dataset of photographs of a table cluttered with up to fifteen unique objects. This study extends the work from this paper by evaluating another network (SqueezeNet).

3. Approaches

The lab's previous work was done in the Python neural network library PyTorch, so this is where the attempt to implement SqueezeNet started. The lab used models of the networks that were pre-trained on ImageNet, a dataset for object detection with 500,000 labeled images. The lab's computer did not have a pre-trained model for SqueezeNet, nor could one be found in PyTorch's collection of pre-trained models. Pre-trained models were, however, discovered for the Caffe library, and the decision to move over to Caffe was made.

The process for installing Caffe went awry: multiple dependencies were failing to install properly and the computer's setup was changed in a way that broke the PyTorch scripts. Caffe changes were rolled back and the project turned back to using PyTorch.

It was at this time the lab discovered PyTorch does have pre-trained models; they were not being accessed properly before. Once that was addressed, and the lab's scripts edited so that they would run with SqueezeNet, SqueezeNet was run. Multiple sessions of training SqueezeNet resulted in a maximum accuracy of 1%: worse than randomly guessing. It was reasoned that the pre-trained model may be not as desired for the lab's work, and so another model had to be made.

The lab decided to train a fresh model of SqueezeNet on the ImageNet dataset. A script was found to train the model, however it was made for a newer version of PyTorch than the lab uses. The script was adjusted to work on the lab's version, and it was run for 87 epochs. After the training, the script validates the model; it was shown that the model had acceptable accuracy while training but close to 0% accuracy while validating. It was discovered the model guessed object #0 always as its first guess, which works for training since the training set has a high presence of scattered 0s as the correct answer. Dissimilarly, the validation answer set goes ascendingly in order with no 0s. Together, this meant the training function thought the model was learning well, but the validating function revealed the model to be ineffective. After 87 epochs the best accuracy for a model on validation was 1.24%, which is approximately where it started after epoch 1. Even with poor results, this model must be evaluated on the lab's data.

Attempts to run the model pre-trained by the lab on ImageNet failed due to the lab's scripts requiring models inheriting from FasterRCNN while models from the pre-training script did not do so, meaning they were incompatible.

4. Results, Conclusions, Future Work

The best accuracy measured on the lab's data on a given pre-trained model was ~1%. The in-lab pre-trained models could not be evaluated.

To conclude, this project did not succeed in implementing SqueezeNet, on the grounds that the closest its work came to implementation came in well below an acceptable threshold of "poor accuracy". Since this project did not succeed in implementing SqueezeNet, it could not evaluate SqueezeNet.

This is not to say nothing was gained for the primary researcher from undertaking this project. The researcher gained a strong appreciation and fluency in Python. They now have an understanding of neural networks and machine learning as a whole. The researcher discovered a love for the location of their work, Providence RI, and intends to return, perhaps for their doctorate. The project did not accomplish all of its initial goals, but that is not to say it was a wasted effort.

Future work includes successfully implementing SqueezeNet and evaluating it against previously-used networks. It also includes finding other architectures and evaluating those as well.

References

[1] Iandola, Forrest N., et al., 2017. *Squeezenet: Alexnet-Level Accuracy With 50x Fewer Parameters And <0.5MB Model Size*. arXiv:1602.07360.