Multi-SpooNN: A Lightweight Neural Network for Multiple Object Detection

Madelyn Gatchel
Brown University
Summer 2019

Abstract

Real-time object detection is essential for autonomous robots to perform tasks in human environments. As a result, many autonomous robots rely on small, efficient neural networks. SpooNN is a lightweight convolutional neural network (CNN) for object detection that is optimized for FPGA implementation [1]. However, the network only supports single object detection without classification. In this project, we extend the network capability to detect and classify multiple objects and then evaluate network performance on various datasets appropriate for autonomous robots.

1 Introduction

1.1 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are a special type of neural network that contain at least one convolution layer, and they are often used in computer vision. CNNs are especially useful for object detection and image classification because the convolution layers allow the network to better identify similar features or patterns in an image by "zooming in" and looking at smaller regions of the image (see Figure 1); in contrast, fully connected layers try to identify features by looking at the entire image. Other common layer types include RELU (activation layer), average pooling, and max pooling.

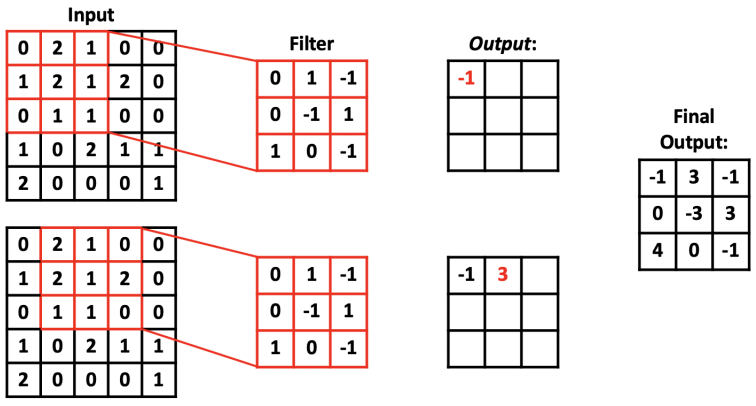

Figure 1: A simple convolution layer using a 3x3 filter and stride equal to 1.

Like most machine learning algorithms, CNNs first train on a large set of images to "learn" the best weights. This training stage is divided into three steps, which are repeated many times for each image in the training set. In the forward step, the input image data is transformed through a series of computations (layers) using the current weights. The results are the bounding box and label guesses for each object in the current image. Intermediate results that will be needed for gradient computation are then stored in memory. In the loss step, the guess is compared with the ground truth (for labels and bounding boxes) using a loss function like softmax or SVM. In the backpropagation step, the chain rule is applied to compute the gradient of the loss function with respect to the weights. These gradients are then used to determine the new weights. After the training stage is complete, the learned weights from the final iteration of the training stage are used in the inference stage. In this stage, each image in the testing set goes through the same set of transformations from the forward step. The resulting label guess for each object in the current image represents the object's classification; the resulting bounding box should encompass the object.

1.2 SpooNN

SpooNN is a network created by the Systems Group at ETH Zurich. The network is appealing because it is lightweight and optimized for FPGA implementation. Additionally, it achieved second place in the 2018 Design Automation Conference. The network is a simplified version of SqueezeNet [3]; instead of using SqueezeNet's fire module, SpooNN uses a half-fire module that uses one fewer convolution layer. Figure 2 compares SpooNN and SqueezeNet's architectures. SpooNN only supports single object detection without classification. In this project, we extend the network capability to detect and classify multiple objects and then evaluate network performance on various datasets appropriate for autonomous robots.

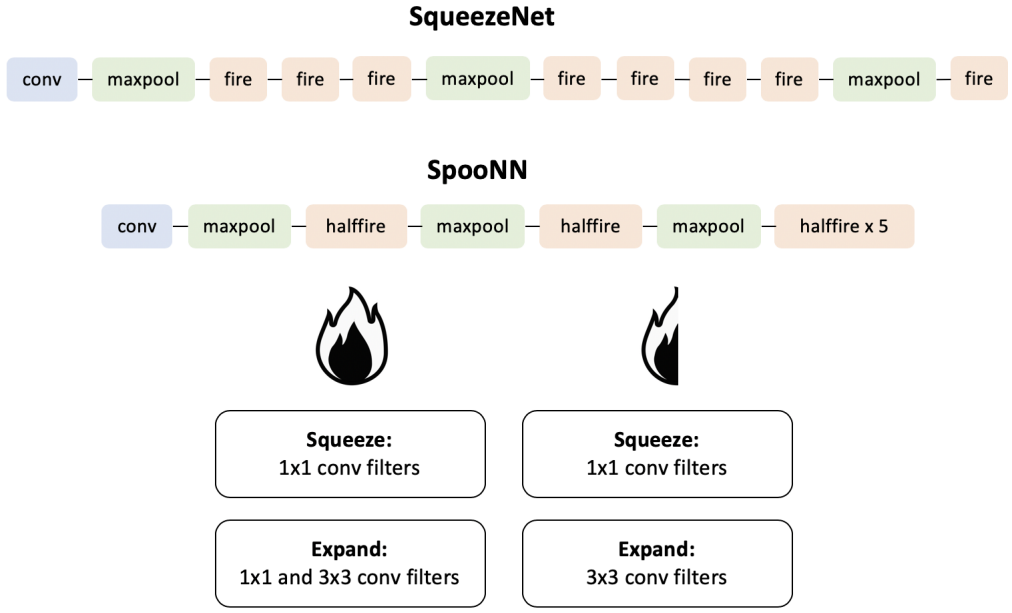

Figure 2: SpooNN and SqueezeNet architectures.

2 Related Work

2.1 SqueezeNet

SqueezeNet is a small CNN that requires few parameters and achieves AlexNet-level accuracy. The network uses a fire module, which is comprised of squeeze layers and expand layers with smaller filter sizes (1x1 and 3x3); the smaller filters require fewer parameters. SqueezeNet is sufficiently small enough to be implemented on an FPGA. As previously discussed, SpooNN is a simplified version of this already-small network.

2.2 YOLOv2

YOLOv2 is a CNN for multiple object detection in real time. The network gets greater than 0.75 mAP on multiple datasets [4]. YOLOv2 uses anchor boxes to predict bounding boxes; instead of using hand-picked anchor box sizes, the network uses k-means clustering on the ground truth bounding boxes using IOU as the distance metric. This method allows the network to focus on adjusting whichever anchor box is closest to the ground truth bounding box in the training stage. While it is certainly fast, efficient, and accurate, the network is still significantly larger than SpooNN.

3 Class-SpooNN: Methodology and Results

While SpooNN does not classify the objects it detects, it only took a few modifications to classify the objects. First, we assigned the ground truth label to the image in the training phase. In the network architecture, we added a softmax layer after the global average pooling layer. Now for the image, we have a probability for each class. In the inference stage, once we find the grid cell with the highest object detection probability, we classify the object by finding the class with the highest probability. The network performed well with a mean IOU of 0.83. Figure 3 shows an example of the progression from SpooNN to Class-SpooNN.

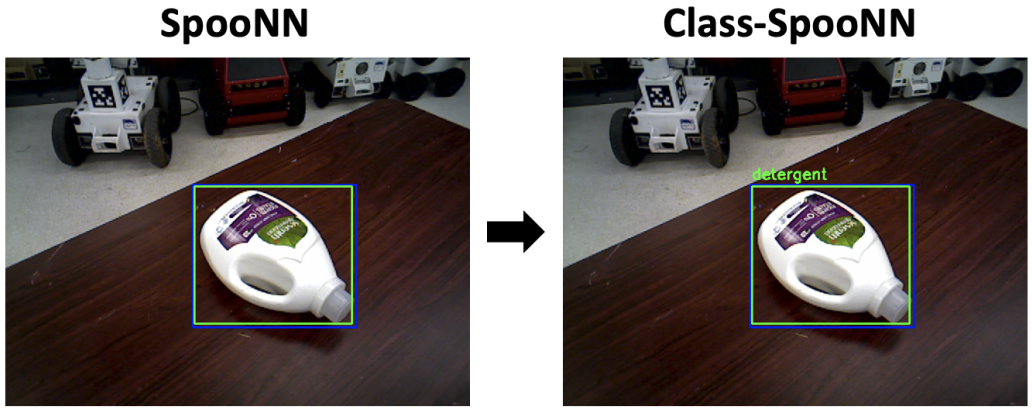

Figure 3: SpooNN and Class-SpooNN detecting a detergent bottle.

4 Multi-SpooNN: Methodology

4.1 Overview

To detect multiple objects in the image, we first remove the global average pooling layer. Now the network views each image as a 14x14 grid. Each grid cell is responsible for determining whether there is an object in the cell, predicting a bounding box around the object, and classifying the object. In the inference stage, we select whichever grid cells have the highest confidence scores and use the corresponding bounding boxes and classifications.

4.2 IOCA vs. IOU

How do we determine whether there is an object in a given grid cell in the training stage? We say there is an object in a given grid cell if there is a ground truth bounding box that intersects the grid cell. But what if the intersection area is really small? What if multiple ground truth bounding boxes intersect the grid cell? In these cases, we can look at a standardization of the intersection area using Intersection Over Cell Area (IOCA) or Intersection Over Union. In the case of multiple objects, we can select whichever object that has the higher IOU or IOCA value. Also, we can use an IOCA or IOU threshold to ensure that grid cells are only responsible for predicting bounding boxes and classification probabilities if the object intersects the grid cell by a significant amount. Figure 4(a) demonstrates how IOCA and IOU are calculated.

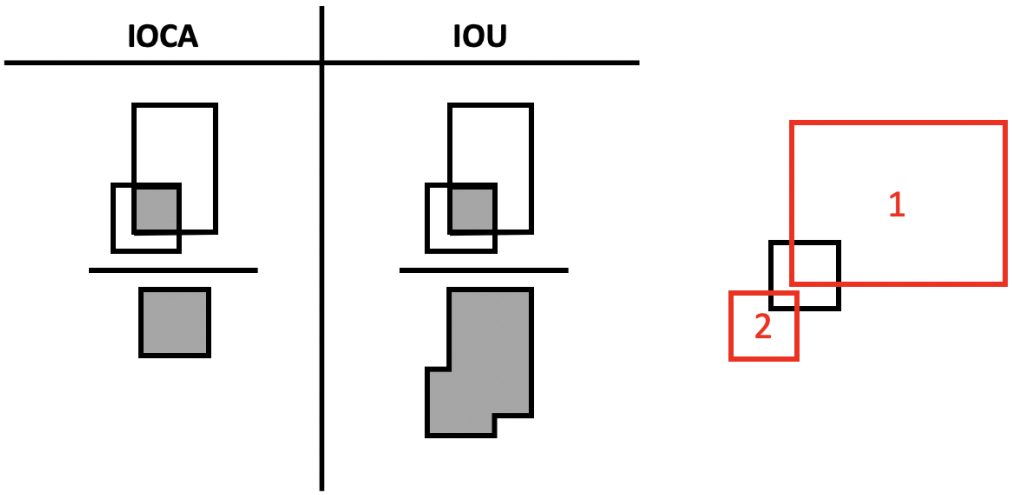

Figure 4: (a) Visual representation of IOCA and IOU (b) IOCA vs IOU and varying object size

The benefit of using IOU as opposed to IOCA is that it prevents larger objects from overpowering or dominating smaller objects just because of their larger area. For example, consider the case in Figure 4(b) above where objects 1 and 2 intersect the black grid cell. With IOCA as the distance metric, object 2 clearly has a higher value; with IOU as the distance metric, object 1 has a higher value. In certain cases, small objects like object 2 could be completely missed with IOCA if they always have to compete with larger objects. As a result, we decided to use IOU for Multi-SpooNN instead.

4.3 Anchor Bounding Boxes

Clustering analysis has shown that many bounding boxes have similar height-width ratios [4]. We used k-means clustering with IOU as the distance metric to find the centroids or "anchor boxes" with the most common height-width ratios. Instead of using a grid cell as the starting bounding box prediction, we used k anchor boxes and kept track of k bounding boxes per grid cell. In the inference stage, we select whichever bounding box has the highest confidence score for the given grid cell. The model makes fewer adjustments to fit an anchor box to the ground truth bounding box than to fit a tiny grid cell to the ground truth bounding box. For each anchor box, we predict its object detection probability and its associated bounding box. Since the model is keeping track of more information each iteration, the training time does increase.

5 Multi-SpooNN: Results and Conclusion

We tested the network on the YCB dataset [2] which contains 21 object classes. This dataset is appropriate for autonomous robots because it contains images of common household objects for robot manipulation tasks. Examples of these common household objects can be found in Figure 5. On this dataset with 2 anchor boxes and an IOU threshold of 0.2, the network

Figure 5: Examples of objects from the YCB dataset.

achieved an mAP value of 0.24. Visually we can confirm how accurate the network is at predicting bounding boxes, even without any thresholding (see Figure 6). However, at this point the classification aspect still needs to be tweaked, and the low classification probabilities are causing the mAP values to be low.

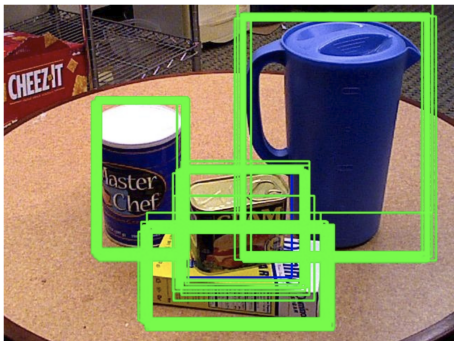

Figure 6: Example of bounding boxes generated from Multi-SpooNN without thresholding.

While at one point we considered that maybe the network is too small to accurately classify multiple objects, we were able to find a setting (grid cell as starting bounding box prediction and IOU threshold $= 0.1$) that produced acceptable heatmaps (i.e. the classification aspect worked), but unfortunately the bounding boxes were not accurate, so the mAP value dropped to 0.05. Figure 7 shows an example of these acceptable heatmaps.

Figure 7: Example of heatmaps generated from Multi-SpooNN without anchor boxes.

Multi-SpooNN shows promise, but still has room for improvement. We suspect that with multiple anchor boxes, the model spends more time training the anchor boxes and as a result spends less time on training on classification. That being said, we believe that using anchor boxes and trying to improve the classification is the next task to tackle.

6 Future Directions

Below is a list of questions I would like to answer in the future about Multi-SpooNN.

- How can we modify the network so it trains both the bounding boxes and the classification equally (i.e. so it produces both good heatmaps and good bounding boxes)? Will this increase the mAP value?
- How does the number of objects per image affect the number of clusters/anchor boxes needed to be effective?
- Is having more anchor boxes worth it given the resulting increase in training time?
- How does the size (area) of the anchor boxes affect what the IOU threshold should be? Should there be a different threshold for each anchor box or just one overall threshold?
- How do we determine which IOU threshold is the best?
- What factors affect the repeatability of Multi-SpooNN with anchor boxes?

- If the IOU values for all anchor boxes are greater than the IOU threshold, do we want to train on all anchor boxes or just the anchor box with the highest IOU? How does this affect the label we select for the grid cell?
- Should we keep track of classification probabilities for each anchor box or just one overall?
- Does changing the bounding box representation from $\{xmin, ymin, xmax, ymax\}$ to $\{width, height, xcntr, ycntr\}$ improve the mAP?
- How much data does the network need to train on to be effective?
- How does the network perform on datasets of images depicting objects in challenging environments?
- How does Multi-SpooNN actually perform when running on an FPGA as opposed to on a GPU?

References

[1] SpooNN. `https://github.com/fpgasystems/spooNN`.

[2] Berk Calli, Arjun Singh, Aaron Walsman, Siddhartha Srinivasa, Pieter Abbeel, and Aaron Dollar. The ycb object and model set: Towards common benchmarks for manipulation research. In IEEE, editor, *IEEE International Conference on Advanced Robotics (ICAR)*, July 2015.

[3] Forrest N. Iandola, Matthew W. Moskewicz, Khalid Ashraf, Song Han, William J. Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and <1mb model size. *CoRR*, abs/1602.07360, 2016.

[4] Joseph Redmon and Ali Farhadi. YOLO9000: better, faster, stronger. *CoRR*, abs/1612.08242, 2016.