Neighboring Collaborative Convolutional Neural Networks

Can Alper, Marissa Bennett, Ramesh Doddaiah, Rasika Karkare, Xinbo Niu

Multiple object detection is a well studied area of Computer Vision. The state-of-the-art models are based on a Machine learning technique known as R-CNN (Regions with CNN features)[7]. R-CNNs utilize Convolutional Neural Networks to predict an object in a given region. Some better working models which are the state-of-the-art equivalent are still building up on the core method of R-CNNs. However, even the best models cannot achieve a performance better than 65% on the Google Open Images Dataset V4[1]. Our team is trying to create a model to compete the current state of the art.

Our method will classify the objects in a given image. The object and the given image can be of varying shapes and sizes. It will do so by drawing the corresponding bounding box around the objects that have been detected. The main field that needs multiple object detection is robotics. Robotics extends to subfields such as autonomous cars and as the usage of multiple object detection goes, it can be used for any physical body that needs to navigate and operate in the real world.

Some other possible use cases of this idea in a variety of fields:

- Remote server and data storage plot workload analysis
- Medical image analysis
- Social Network platforms for image context

Current state of the art frameworks are:

- YOLO (v3, 9000)[3, 8]
- Mask RCNN[4]
- Faster RCNN[2]
- Single Shot MultiBox Detector[5]

Best out of all: YOLO 9000 is a state-of-the-art object detection system that is based on R-CNN. Along with R-CNN, it also uses batch normalization to improve recall and localization while maintaining classification accuracy. It regularizes the layers and eliminates need for a dropout layer. It uses a high resolution classifier so as to switch to learning object detection and adjust to the new input resolution.

Our method will approach the problem from a different perspective than the state-of-the-art models. Ideally it will be more efficient, and more accurate in detecting the objects compared to the state-of-the-art models. We will utilize the regional information from the images, however, we plan to do so by using a structure that can pass along information to

neighboring regions. Aside from region based models, we also took some inspiration from a new method that extends upon CNNs called Dynamic Routing Between Capsules.[6]

We think that the images have continuous spatial information along with information that is unique to an area. So in order to classify objects correctly, regions should share processed data to their neighbors, then a more precise prediction can be made for a region.

We propose to do this with following steps (more details will be provided in the website):

Step 1: Separate image into regions, fill the region with the label data.

Input:

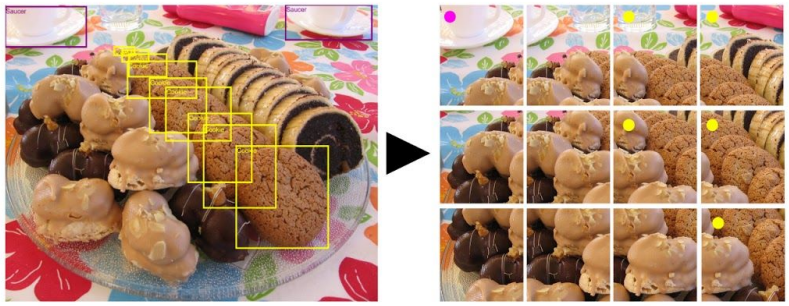
Image [width, height, 3(RGB)]

Output:

Ω: Region size
β: Stride
Δ: Class count

Data: [width/β, height/β, Ω, Ω, 3]

Label: [width/β, height/β, Δ]



Step 2: Train with the generated regions

Model 1 (encode)

θ: Encode size

Input: [width/β, height/β, Ω, Ω, 3]

output: [width/β, height/β, θ, θ, features]

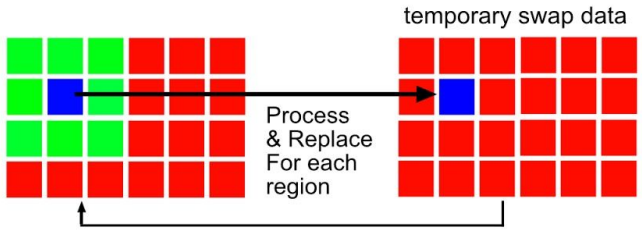
Model 3 (predict)

θ: Encode size

Input: [width/β, height/β, θ, θ, features]

output: [width/β, height/β, Δ]

Model 2 (share)



Input: [9, θ, θ, features]

output: [1, θ, θ, features]

Repeated N times

Step 3: Reconstruct bounding boxes

This step is a modified stepping algorithm that will find neighboring regions with the same labels and generates the bounding box with that information.

Input: [width/β, height/β, Δ]

output: Bounding boxes with classes

Due to the hard nature of computer vision, we need significant amount of processing power. So, to implement and evaluate the described method, we will use GPUs available to our team. We already have a pool of RTX 2080Ti, GTX 1080Ti and GTX 1080 cards available on our personal machines, but we can also utilize other services like Google Cloud or WPI GPU cluster.

Our evaluation method will be inline with the state-of-the-art models. We will also use the intersection over union(IoU) based mean Average Precision (AP) method to evaluate our success. However, due to the time limitation of the class time, we are planning on using only a subset of Open Images Dataset.

Resources:

1. Kaggle competition on google object detection (https://www.kaggle.com/c/google-ai-open-images-object-detection-track/leaderboard)
2. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks (https://arxiv.org/abs/1506.01497)
3. YOLOv3: An Incremental Improvement (https://pjreddie.com/media/files/papers/YOLOv3.pdf)
4. Mask RCNN (https://arxiv.org/abs/1703.06870)
5. SSD: Single Shot MultiBox Detector (https://arxiv.org/abs/1512.02325)
6. Dynamic Routing Between Capsules (http://www.cs.toronto.edu/~hinton/absps/DynamicRouting.pdf)
7. Rich feature hierarchies for accurate object detection and semantic segmentation (https://arxiv.org/pdf/1311.2524.pdf)
8. YOLO9000: Better, Faster, Stronger (https://arxiv.org/abs/1612.08242.pdf)