CSE616: Neural Networks and Applications

Paper Discussion and Presentation

Ayman Wagih Mohsen (2000728)

Robeir Samir George (1901823)

Fast R-CNN

1. Introduction

Fast R-CNN is an object detection model that improves in its predecessor the R-CNN which is called regional-based convolutional neural network, in a number of ways. Instead of extracting CNN features independently for each region of interest, Fast R-CNN aggregates them into a single forward pass over the image; i.e. regions of interest from the same image share computation and memory in the forward and backward passes.

1.1 R-CNN

The region-based CNNs has a good accuracy in object detection, but it has some disadvantages such as: the training is a multi-stage pipeline, as well as the problem that the training is expensive in space and time, and finally the detection at test time is very slow (reaches about 47 seconds per image).

Like R-CNN we have SPPnet, which also has a great drawback, similar to R-CNN, training is a multi-stage pipeline that involves extracting features, fine-tuning a network with log loss, training SVMs, and fitting boundary-box regressors.

1.2 Contributions

The paper introduces a new training algorithm, trying to fix the drawbacks of both the R-CNN and SPPnet.

The algorithm is called Fast R-CNN, with advantages such as: the training is a single-stage, training updates all layers of the network, and no disk storage is required for feature caching.

2. Fast R-CNN architecture and training

The paper uses Selective Search to generate a number of regions of interest (RoIs) (the specific method for generating RoIs is not the focus of this paper). Each RoI is then evaluated for whether there's an object there, and if so what its bounding box is. The innovation here is to share expensive early-layer computation with the help of an RoI pooling layer, which also means that the heavy use of caching features to disk is no longer necessary.

The architecture takes in an image and a set of RoIs. For each RoI we output softmax probabilities over K classes and x, y, w, h adjustments to the region that determine a bounding box prediction — the exact parameterization of the bounding box numbers is somewhat complicated and is laid out in the earlier R-CNN paper. For the loss function, the paper follows the now-standard strategy of putting as much as possible into one network and combining multiple terms into a single loss. In this case there are two terms: a class loss (a log loss) and a location loss (a smoothed L1 loss summed over the 4 outputs — they went with this because L2 loss caused exploding gradients).

Regions that are classified as background don't include a location loss, which makes sense considering that backgrounds don't really have bounding boxes.

$$L(p, u, t^u, v) = L_{\text{cls}}(p, u) + \lambda[u \geq 1] L_{\text{loc}}(t^u, v)$$

RoI Pooling:

This is what the paper is all about, but it's actually quite simple and is equivalent to what is now referred to as adaptive pooling. The idea is to have a max pooling layer that takes an arbitrary input region and maps it down to a fixed *H*×*W* feature map (e.g. 7×7 — this is a hyperparameter). You just divide the region into *H*×*W* subregions and do max pooling.

Scale Invariance:

In the object detection problems, it is common to try achieving scale invariant detection, here we have two approaches: 1. Via the brute force learning and 2. Via using image pyramids.

In the brute force approach, each image is processed at a pre-defined pixel size during both training and testing. The other approach provides approximate scale-invariance to the network through an image pyramid, where at the testing time, the image pyramid is used to approximately scale-normalize each object proposal.

3. Fast R-CNN Detection

- The network takes an image as an input, and a list of R objects proposals to score. R is around 2000 at testing time.

- When using an image pyramid, each RoI is assigned to the scale such that the scaled RoI is closer to 224^2 pixels in area.

- For each test RoI, the forward pass outputs a class and a set of predicted bounding-box offsets relative to the RoI.

3.1 Truncated SVD for faster detection

For whole-image classification, the time spent computing the fully connected layers is small compared to the conv layers. On the contrary, for detection the number of RoIs to process is large and nearly half of the forward pass time is spent computing the fully connected layers