R-CNN

Introduction

RCNN (Region-based Convolutional Neural Networks) is a popular object detection algorithm that was introduced in a paper by Girshick et al. in 2014.

The main idea behind RCNN is to propose regions in an image that are likely to contain an object. Each region is then passed through a convolutional neural network (CNN) to extract features, which are fed into a set of class-specific linear support vector machines (SVMs) to determine the presence or absence of each object class.

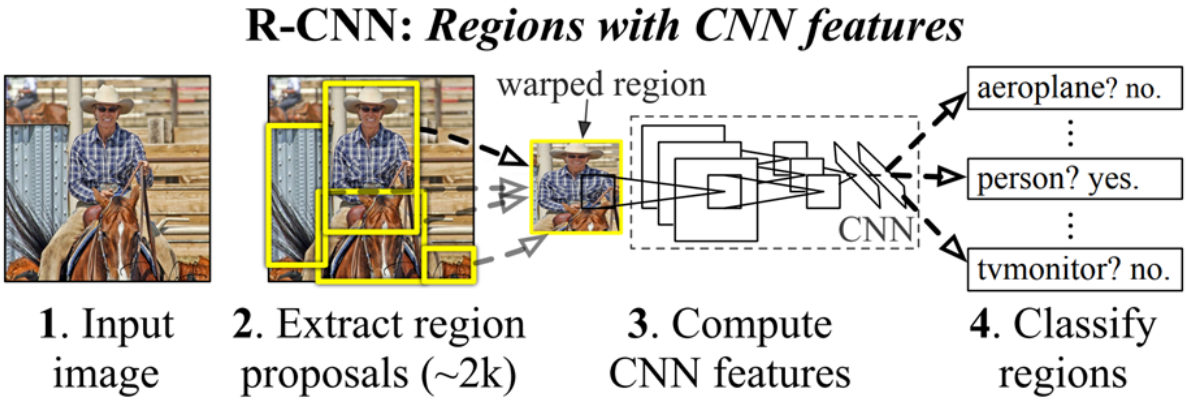

Fig 1. Object detection system overview (Source: R-CNN paper)

One of the main advantages of RCNN is that it can detect objects in a wide range of sizes and aspect ratios, since the region proposal algorithm generates region proposals of varying sizes and shapes. Additionally, because RCNN uses a CNN to extract features from each region, it is able to capture high-level semantic information about the objects in the image.

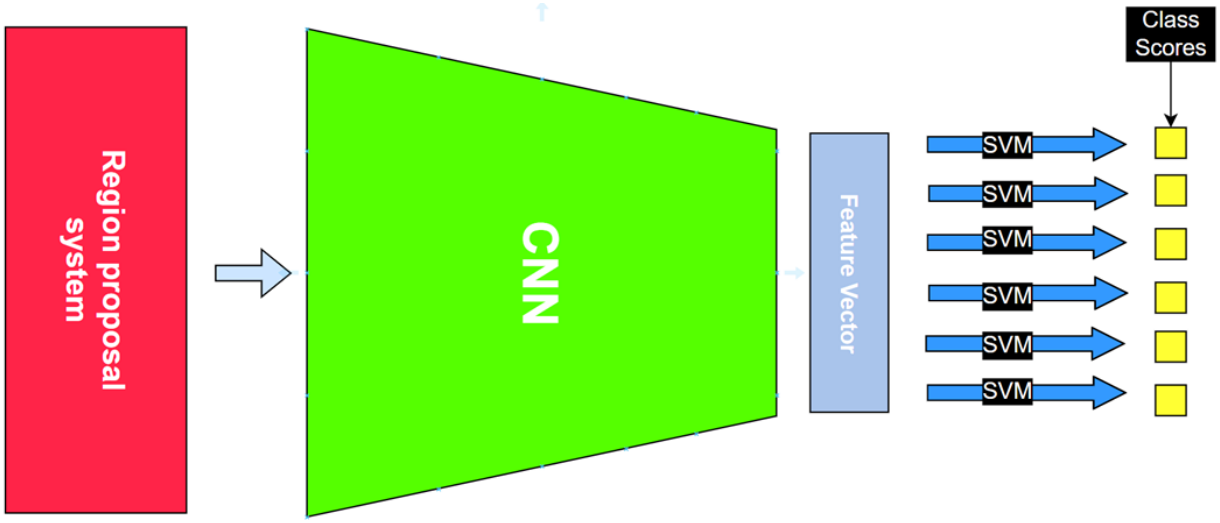

Fig 2. Simplified block diagram of the R-CNN architecture

However, RCNN also has some limitations. It can be quite slow, since each region-proposal must be passed through the CNN and the set of SVMs separately. This led to the development of faster RCNN, which uses a region proposal network (RPN) to generate region proposals and shares the convolutional features across proposals to speed up the detection process.

Architecture

The whole object detection system is divided into 3 main modules:

1. The first generates category independent region proposals
2. The second module is a large CNN that produces a fixed length feature vector for each of these regions.
3. The third module is a set of class specific linear SVMs.

The region proposal module generates around 2000 region proposals that are not dependent on class labels. Each proposed region is then transformed into a fixed-length feature vector using the CNN, which is then input into class-specific SVMs.

R-CNN is agnostic to a particular region proposal method, ‘Selective Search’ is used in the R-CNN paper to enable a controlled comparison.

Fig 3. Different object proposal transformations. (A) the original object proposal at its actual scale relative to the transformed CNN inputs; (B) tightest square with context; (C) tightest square without context; (D) warp. Within each column and example proposal, the top row corresponds to p = 0 pixels of context padding while the bottom row has p = 16 pixels of context padding. (Source: R-CNN paper)

Now, the proposed regions do not have a fixed aspect ratio. So, to make it compatible with the CNN input the authors propose a few methods, as illustrated in the figure above. Out of them, they adopt warping the image (D) with a context padding of p=16 pixels.

Training

The training is a 2-step process. The first step is supervised pre-training, where a CNN is trained on a large dataset using image-level annotations only. The second step is domain-specific fine-tuning, where the pre-trained CNN is further trained using warped region proposals for object detection.

The CNN's architecture remains unchanged, except for replacing the classification layer with a new layer that includes an additional class for background. In fine-tuning, region proposals with ≥ 0.5 IoU overlap with a ground-truth box are treated as positive and others as negative.

The training is performed using stochastic gradient descent, and each mini-batch includes 32 positive and 96 background windows. Positive windows are sampled more often to address their rarity compared to background.

Annotating training data: Consider training a binary classifier to detect cars. It's clear that an image region tightly enclosing a car should be a positive example. Similarly, it's clear that a background region, which has nothing to do with cars, should be a negative example.

Less clear is how to label a region that partially overlaps a car. For partially overlapping regions, the authors use an IoU overlap threshold of 0.3, determined by a grid search, to label them as negatives or ignored. Choosing the threshold carefully is crucial, as setting it to 0.5 or 0 results in decreased mean average precision by 5 or 4 points, respectively. Ground-truth bounding boxes are used as positive examples for each class.

Put simply:

a) Ground truth boxes for a class – *Positive Examples*

b) Proposals with less than 0.3 IoU overlap with all instances of a class – *Negative Examples*

c) Proposals that fall into the gray zone (More than 0.3 IoU overlap, but are not ground truth) – *Ignored*

Bounding-Box regression: The R-CNN paper uses a simple bounding-box regression stage to improve localization performance. After scoring each selective search proposal with a class-specific detection SVM,

a new bounding box is predicted for the detection using a class-specific bounding-box regressor.

Test-time Detection

At test time, selective search is run on each test image (in fast mode), proposing around 2000 proposals per image. These proposals are then warped using the technique mentioned above and passed through the CNN to compute a fixed length feature vector. Then for each class, this feature vector is scored using the SVMs trained for each class. Given all scored regions, ‘Greedy Non-Maximum Suppression’ is performed for each class separately for all regions.

Non-Max Suppression:

Input: A list of Proposal boxes B, corresponding confidence scores S and overlap threshold N.

Output: A list of filtered proposals D.

Algorithm:

1. Select the proposal with highest confidence score, remove it from B and add it to the final proposal list D. (Initially D is empty).
2. Now compare this proposal with all the proposals — calculate the IOU (Intersection over Union) of this proposal with every other proposal. If the IOU is greater than the threshold N, remove that proposal from B.
3. Again, take the proposal with the highest confidence from the remaining proposals in B and remove it from B and add it to D.
4. Once again calculate the IOU of this proposal with all the proposals in B and eliminate the boxes which have a higher IOU than threshold.
5. This process is repeated until there are no more proposals left in B.

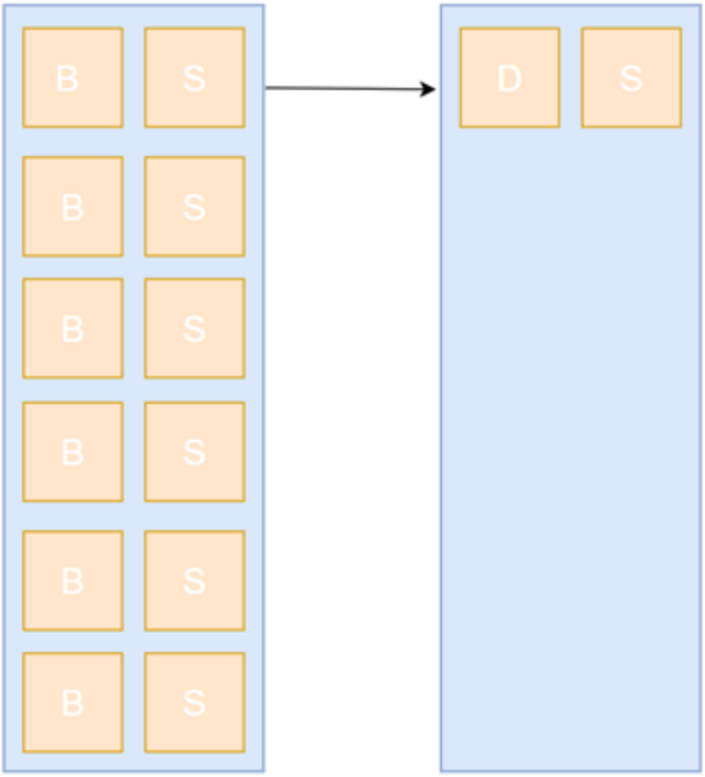

Fig 4. Non-max Suppression algorithm

Conclusion

Two properties that make this system efficient. First, all CNN parameters are shared across all classes; and second, the feature vector is relatively low dimensional when compared to other approaches.

With stagnation in object detection performances, where the best performing models relied on complex ensembles of combining multiple low-level features with high level context; R-CNN presented a simple, scalable algorithm that achieved 30% relative improvement over PASCAL VOC 2012.

This was achieved mainly via 2 insights. The first is to apply CNNs to bottom-up region proposals. The second is a paradigm for training large CNNs when labeled training data is scarce.

They conjectured that the “supervised pre-training/domain-specific fine-tuning” paradigm is still highly effective for a variety of data-scarce vision problems today.