Mask R-CNN

Computer Vision Framework by Facebook Research

Olga Agapova
Higher School of Economics (National Research University)
Moscow, Russia

***Abstract*—Over the last few years, research in computer vision is progressively moving towards solving four problems: classification, semantic segmentation, object detection and instance segmentation. Not long ago the Facebook AI Research group proposed new approach to instance segmentation which is called Mask R-CNN. This approach is a logical extension of sequence of some other frameworks, which solve different problems of mentioned above. This paper contains a brief survey of the walk-through stages and reviews the design of Mask R-CNN framework in particular.**

Keywords—neural network; object detection; instance segmentation

1. Introduction

Recent research in computer vision proposes several approaches to following problems: classification, semantic segmentation, object detection and instance segmentation, which allow to solve these problems with mixed success. Mask R-CNN can be represented as a logically extended Faster R-CNN framework, which, in turn, is an extension of Fast R-CNN.

In short, Mask R-CNN is a deep neural network architecture which aims to solve the problem of instance segmentation. This means that Mask R-CNN can separate different objects in an image or a videoclip. Mask R-CNN can be divided into the two stages. First, it generates the proposal about location of objects, based on the original image. Second, it predicts the class of every object, and then generates the mask for every object, based on the result of the first stage. Mask is a rectangular matrix with binary elements, where "1" means that corresponding pixel belongs to the object, and zero means that it doesn't.

This paper is organized as follows: Section 2 describes previous neural network designs and problems related with each of them. Section 3 describes Mask R-CNN architecture and compares this approach to the previous ones.

2. Region-based, Fast, Faster

We assume that the reader is already familiar with such concepts as "multi-layer neural network", "backpropagation", as well as with how multi-layer neural networks are organized in general terms (or is referred to [1]).

Before proceeding to describing R-CNN, we would like to define a convolutional layer. Convolutional layer allows to unite values in neighbor pixels in order to extract the value of the generalized feature. For this aim there's a square window of size 3x3, or 7x7 pixels, etc. This window is called a kernel. The elements of the kernel have weights, which, multiplied by value of the corresponding pixel, is summarized. This result is a value of corresponding feature and an element of the next layer, which is called the convolutional layer.

2.1 R-CNN

Convolutional neural networks (CNN) successfully solve the classification problem. The US Berkeley group [2] has set a goal to modify CNN in order to use it for object detection. The Region-based CNN (R-CNN) is a neural network which is supposed to do this.

The main idea is that R-CNN takes previously prepared segments instead of the whole image. The image is segmented by a specific method. There are many methods to do this, but the authors chose Selective Search for this purpose.

Selective Search returns about 2 thousand regions of different size and aspect ratio, which is then edited and given to a convolutional neural network (authors used CaffeNet for this purpose). Then CaffeNet network extracts 4096-dimensional feature vector, which is then used to classify regions. For every object class, SVM is used in order to determine if the object of this class is in this region or not. For the case that the object is not fully in the region, Intersection over Union (IoU) is used. IoU calculates the area of object bounding box intersection and the region, then

compares it to the area of region and bounding box united, and then decides if the object is in the region or not.

During the error analysis the authors developed the method which allows to minimize the error in bounding box estimation. This method was called a bounding box regression (since it uses linear regression). After all the objects in the candidate region are classified, the following parameters are calculated: dx, dy, dw, dh. These parameters describe how the center of the bounding box should be moved for a better estimation.

Thus, the procedure of object detection by R-CNN can be divided into several steps as follows:

1. Proposing candidate regions and edit them to fit the network;
2. Extracting the feature vector;
3. Using SVM for each object class;
4. Adjusting the bounding box by linear regression.

The authors have also noticed that the developed framework works good for semantic segmentation, whis means that R-CNN can rather successfully detect all the objects of certain class in an image.

2.2 Fast R-CNN

However, bounding box regression and using support vector machine for every object class appeared to be expensive in terms of memory usage. Moreover, R-CNN has shown less success when used with other networks that CaffeNet, and this meant poor speed of the system.

Thus, the network was modified to Fast R-CNN. The main idea is that CNN must take the whole image as the input, instead of thousands of candidate segments. The result was supposed to be combined with general feature map.

Furthermore, the authors of the Fast R-CNN suggested that combining all three training procedures (CaffeNet, SVM, BBR) into one. The method of doing this is extentively described in [3].

The result of the modifications mentioned above is impressive: Fast R-CNN works 9 times faster than R-CNN using the same CNN.

2.3 Faster R-CNN

Despite all the previous modifications there were still a few speed problems that could be solved. The algorithm of proposing candidate regions was the test-time bottleneck of all framework. Microsoft Research group proposed a new approach to this step, which contains Region Proposal Network ([4]).

We will briefly review the main point of their approach.

First, they use CNN to extract the “convolutional feature map”, which works just like in previous frameworks. Then RPN is used, which is generally organized as follows: there’s a CNN which takes an image as input and returns a a set of candidate regions; to generate the proposals, they use a neural network which takes a small square window as the input and outputs lower-dimensional feature so they have a feature map as a result of sliding this network over the convolutional feature map. Then the result of procedure described above is given to the box-regression layer (*reg*) and box-classification layer (*cls*). These layers outputs are based on *anchors*. At each location several region proposals are predicted, with maximum possible proposals per location is denotes as *k*. So there are *k* frames of different size and ratio, called anchors, and for each frame there are four possible coordinates (which is the output of the *reg* layer) and two probabilities: the probability of object or no object inside the frame (the output of the *cls* layer).

Then anchors with good probability of object inside them are given further to the object detection module and BBR module, which could be still implemented as in Fast R-CNN framework.

Thus, the process can be divided into the following steps:

1. RPN training for extracting the candidate regions;
2. Training Fast R-CNN with respect to result of the previous step;
3. Fast R-CNN is used for weight adjustion for the RPN module (without affecting shared convolutional layers);
4. Fast R-CNN module is being adjusted based on the result of the previous step.

3. Mask R-CNN

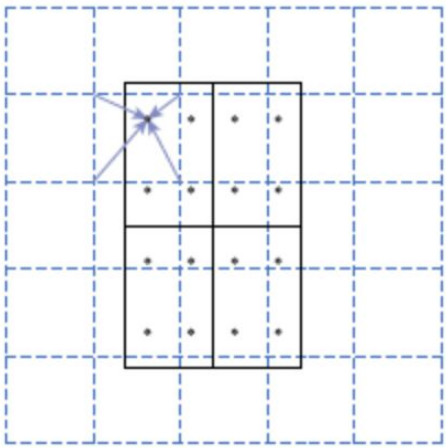
Faster R-CNN meets better success in object detection than Fast R-CNN. The next step was to apply Faster R-CNN methods to instance segmantation. Mask R-CNN (a framework developed by Facebook AI Research) is an extention of Faster R-CNN since it uses one more branch, which predicts the location of the object bounding box, so that it solves the instance segmentation problem. The mentioned branch is called *the mask*. The mask is a rectangular matrix, where elements accept values 0 or 1. “0” means that the corresponding pixel doesn’t belong to the specified object, and “1” means it does.

The authors divide the framework into the two parts: the *backbone*, in which features are extracted by CNN, and the *head*, in which classification problem is solved, bounding boxes are estimated, and the masks are predicted.

The authors [5] assume that the most effective way to estimate the masks is to predict them for each object class without any knowledge about semantic meaning of the object class, and then to choose the most accurate one.

This causes the following problem: the convolutional feature map can is lower-dimensional than the original image, but the mask must be estimated with respect to pixels, which was impossible in terms of state-of-the-art methods, since number of pixels (which is integer) can only be matched to a fractional number of features. Obviously, the same problem appeared in Faster R-CNN framework, but the solution was easy: number of corresponding pixels was rounded. But for this purpose this solution wouldn't work, since the mask would be too unaccurate.

So the authors change the module RoIPool (in which the feature map is extracted and matched to the origin picture) to the module called RoIAlign, the main point of which is using bilinear interpolation (which is described in [5]) with four nearest integer points.



(Figure 1) RoIAlign calculates the value of each feature by bilinear interpolation from the nearest grid points on the feature map

Mask R-CNN has shown rather good results in human pose estimation. The method is to extract keypoints (knees, elbows, shoulders, etc.). Masks for this case will have only one element with value "1" (the pixel that corresponds with the exact keypoint).

Some experiments were conducted with different CNNs used as the backbone. The most common, ResNet-50/101, showed good results. But several experiments were conducted with Feature Pyramid Network (FPN). These experiments showed that using FPN as the backnbone meets great success in accuracy and productivity. FPN contains two CNNs -- bottom-up, which extracts features (usually it is ResNet or VGG), and up-bottom, which generates feature map of the same size as the bottom-up one. There are also some connections between them which allow to summarize all the results into the final result.

References

[1] The Introduction to Neural Networks we all need ! (Part 1)

[2] Rich feature hierarchies for accurate object detection and semantic segmentation Tech report (v5)

[3] Fast R-CNN

[4] Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks

[5] Bilinear Interpolation Definition

1734 words