orbit FUTURE ACADEMY
Skills For Future Jobs

Kampus Merdeka INDONESIA JAYA

Object Localization and Object Detection

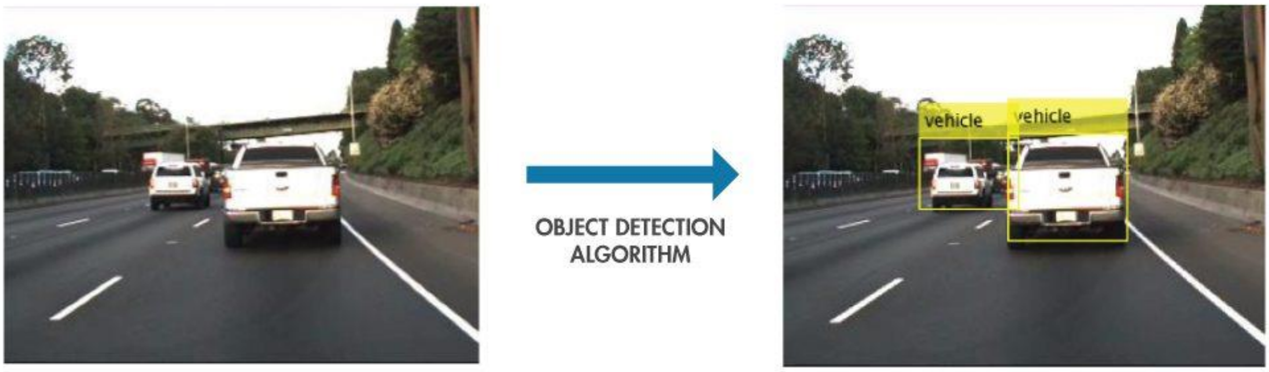
Topics Covered

1. Object Detection Overview
2. Applications of object detection
3. R-CNN
4. Fast R-CNN
5. Faster R-CNN
6. Comparing R-CNN, Fast R-CNN and Faster R-CNN
7. Summary

Object Detection Overview

Object detection is an important computer vision task used to detect instances of visual objects of certain classes (for example, humans, animals, cars, or buildings) in digital images such as photos or video frames. The goal of object detection is to develop computational models that provide the most fundamental information needed by computer vision applications: “What objects are where?”.

Object detection is one of the fundamental problems of computer vision. It forms the basis of many other downstream computer vision tasks, for example, instance segmentation, image captioning, object tracking, and more. Specific object detection applications include pedestrian detection, people counting, face detection, text detection, pose detection, or number-plate recognition.



Object Detection and Deep Learning

In the last few years, the rapid advances of deep learning techniques have greatly accelerated the momentum of object detection. With deep learning networks and the computing power of GPU’s, the performance of object detectors and trackers has greatly improved, achieving significant breakthroughs in object detection.

How Object Detection works

Object detection can be performed using either traditional (1) image processing techniques or modern (2) deep learning networks.

1. **Image processing techniques** generally don’t require historical data for training and are unsupervised in nature.
   - **Pro’s:** Hence, those tasks do not require annotated images, where humans labeled data manually (for supervised training).
   - **Con’s:** These techniques are restricted to multiple factors, such as complex scenarios (without unicolor background), occlusion (partially hidden objects), illumination and shadows, and clutter effect.

2. **Deep Learning methods** generally depend on supervised training. The performance is limited by the computation power of GPUs that is rapidly increasing year by year.
   - **Pro’s:** Deep learning object detection is significantly more robust to occlusion, complex scenes, and challenging illumination.
   - **Con’s:** A huge amount of training data is required; the process of image annotation is labor-intensive and expensive. For example, labeling 500’000 images to train a custom DL object detection algorithm is considered a small dataset. However, many benchmark datasets (MS COCO, Caltech, KITTI, PASCAL VOC, V5) provide the availability of labeled data.

Applications of object detection

Video surveillance

Because state-of-the-art object detection techniques can accurately identify and track multiple instances of a given object in a scene, these techniques naturally lend themselves to automating video surveillance systems.

For instance, object detection models are capable of tracking multiple people at once, in real-time, as they move through a given scene or across video frames. From retail stores to industrial factory floors, this kind of granular tracking could provide invaluable insights into security, worker performance and safety, retail foot traffic, and more.

Crowd counting

Crowd counting is another valuable application of object detection. For densely populated areas like theme parks, malls, and city squares, object detection can help businesses and municipalities more effectively measure different kinds of traffic—whether on foot, in vehicles, or otherwise.

This ability to localize and track people as they maneuver through various spaces could help businesses optimize anything from logistics pipelines and inventory management, to store hours, to shift scheduling, and more. Similarly, object detection could help cities plan events, dedicate municipal resources, etc.

Anomaly detection

Anomaly detection is a use case of object detection that’s best explained through specific industry examples.

In agriculture, for instance, a custom object detection model could accurately identify and locate potential instances of plant disease, allowing farmers to detect threats to their crop yields that would otherwise not be discernible to the naked human eye.

And in health care, object detection could be used to help treat conditions that have specific and unique symptomatic lesions. One such example of this comes in the form of skin care and the treatment of acne—an object detection model could locate and identify instances of acne in seconds.

What’s particularly important and compelling about these potential use cases is how they leverage and provide knowledge and information that’s generally only available to agricultural experts or doctors, respectively.

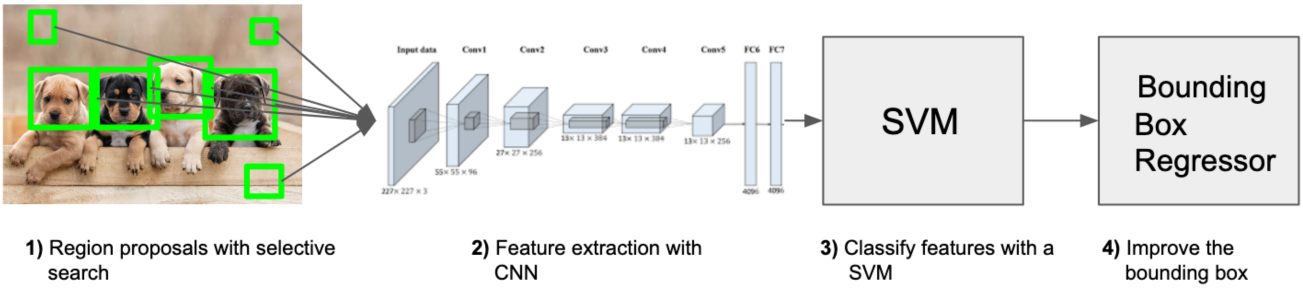
Self-driving cars

Real-time car detection models are key to the success of autonomous vehicle systems. These systems need to be able to identify, locate, and track objects around them in order to move through the world safely and efficiently.

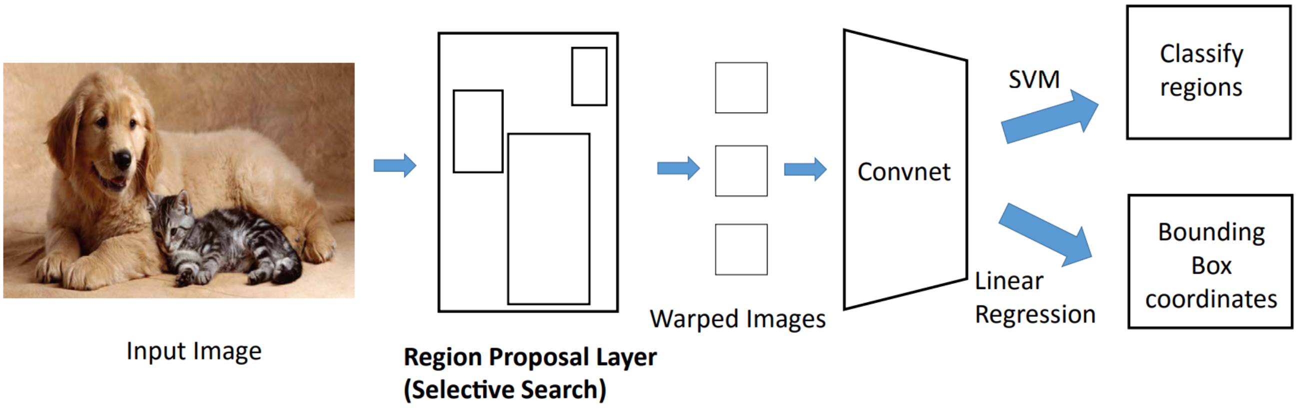
And while tasks like image segmentation can be (and often are) applied to autonomous vehicles, object detection remains a foundational task that underpins current work on making self-driving cars a reality.

R-CNN

R-CNNs (Region-based Convolutional Neural Networks) are a family of machine learning models used in computer vision and image processing. Specially designed for object detection, the original goal of any R-CNN is to detect objects in any input image defining boundaries around them.



An input image given to the R-CNN model goes through a mechanism called selective search to extract information about the region of interest. Region of interest can be represented by the rectangle boundaries. Depending on the scenario there can be over 2000 regions of interest. This region of interest goes through CNN to produce output features. These output features then go through the SVM(support vector machine) classifier to classify the objects presented under a region of interest.



The above image represents the procedures of an R-CNN while detecting an object using it. Using the R-CNN within an image we extract regions of interest using the region extraction algorithm. The number of regions can be extended to 2000. For each region of interest, the model manages the size to be fitted for the CNN, where CNN computes the features of the region and SVM classifiers classify what objects are presented in the region.

Who Proposed RCNN?

Inspired by the research of Hinton’s lab at the University of Toronto, a small team at UC Berkeley, led by

- Professor Jitendra Malik, comprised of
- Ross Girshick,
- Jeff Donahue, and
- Trevor Darrel

proposed R-CNN by testing on the PASCAL VOC challenge.

Selective Search

There can be various approaches to perform object localization in any object detection procedure. Using sliding filters of different sizes on the image to extract the object from the image can be one approach that we call an exhaustive search approach. As the number of filters or windows will increase, the computation effort will increase in an exhaustive search approach.

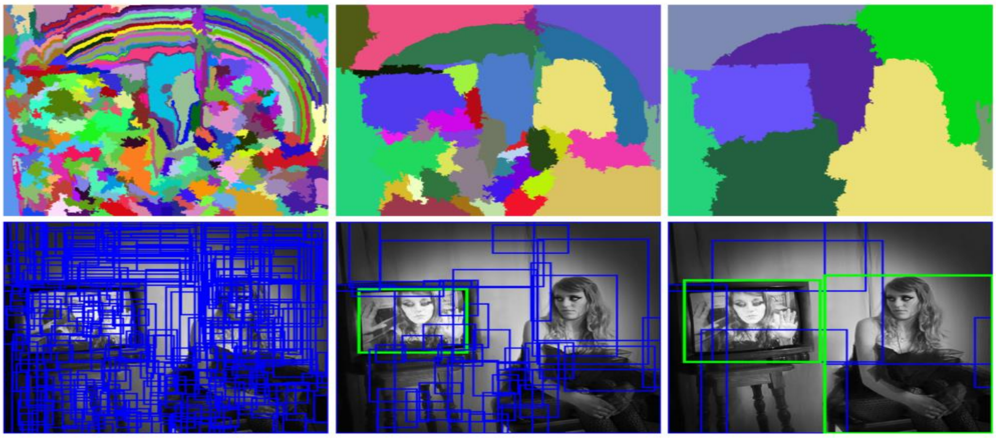
The selective search algorithm uses exhaustive search but instead of using it alone it also works with the segmentation of the colours presented in the image. More formally we can say selective search is a method that separates objects from an image by providing different colours to the object.

This algorithm starts with making many small windows or filters and uses the greedy algorithm to grow the region. Then it locates the similar colours in the regions and merges them together.
The similarity between the regions can be calculated by:

$$S(a,b)=Stexture(a,b)+Ssize(a,b)$$

Where the Stexture(a,b) is visual similarity and Ssize(a,b) similarity between the regions.

Using this algorithm, the model continues to merge all the regions together to improve the size of the regions. The image is a representation of a selective search algorithm.



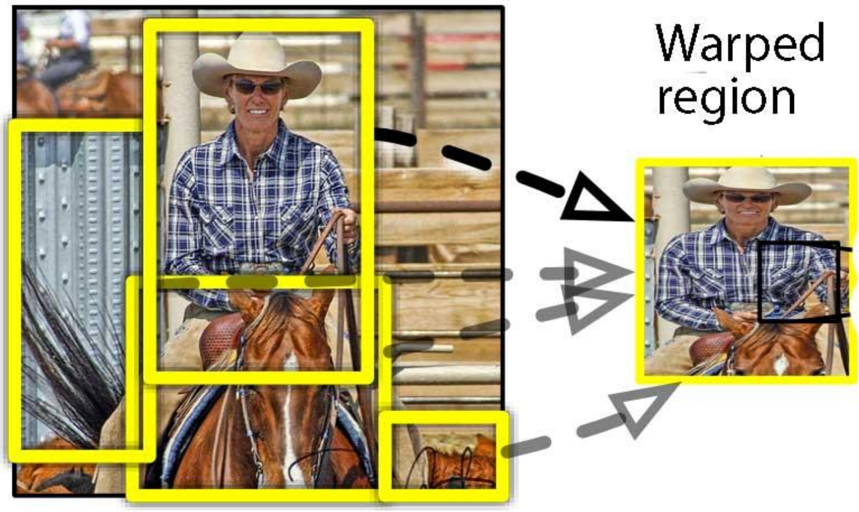
In the image, we can see the making of tiny regions to the selection of the objects, space under the region increases as the similarity between regions increases.

Selective search algorithms are a basic phenomenon for object localization. In object detection after localization, there are three processes left from which an extracted object will go.

- Warping
- Extracting features with a CNN
- Classification

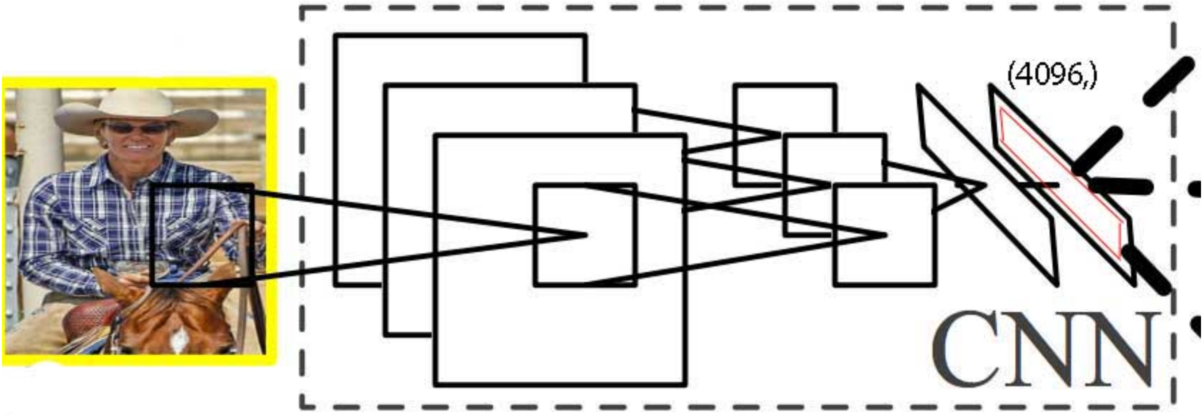
Warping

After selection of the region, the image with regions goes through a CNN where the CNN model extracts the objects from the region. Since the size of the image should be fixed according to the capacity of CNN we require some time or most of the time to reshape the image. In basic R-CNN we wrap the region into 227 x 227 x 3 size images.



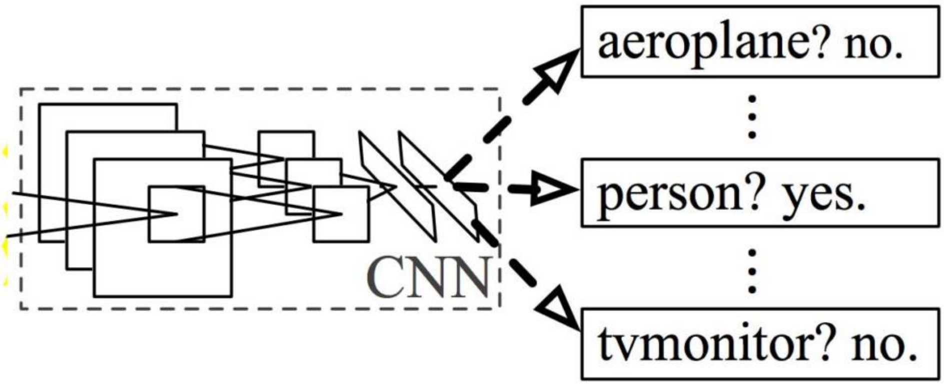
Extract objects with a CNN

A wrapped input for CNN will be processed to extract the object of size 4096 dimensions.

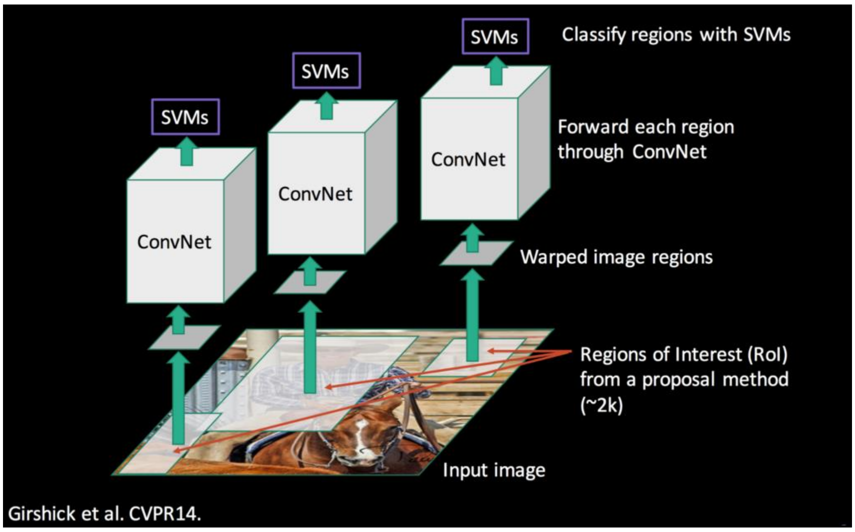


Classification

The basic R-CNN consists of an SVM classifier to segregate different objects into their class.



The whole process architecture of R-CNN can be represented as.


SVMs
SVMs
SVMs
Classify regions with SVMs
ConvNet
ConvNet
ConvNet
Forward each region through ConvNet
Warped image regions
Regions of Interest (RoI) from a proposal method (~2k)
Input image

Girshick et al. CVPR14.

At the end of the model, the boundary box regressor works for defining objects in the image by covering the image by the rectangle.

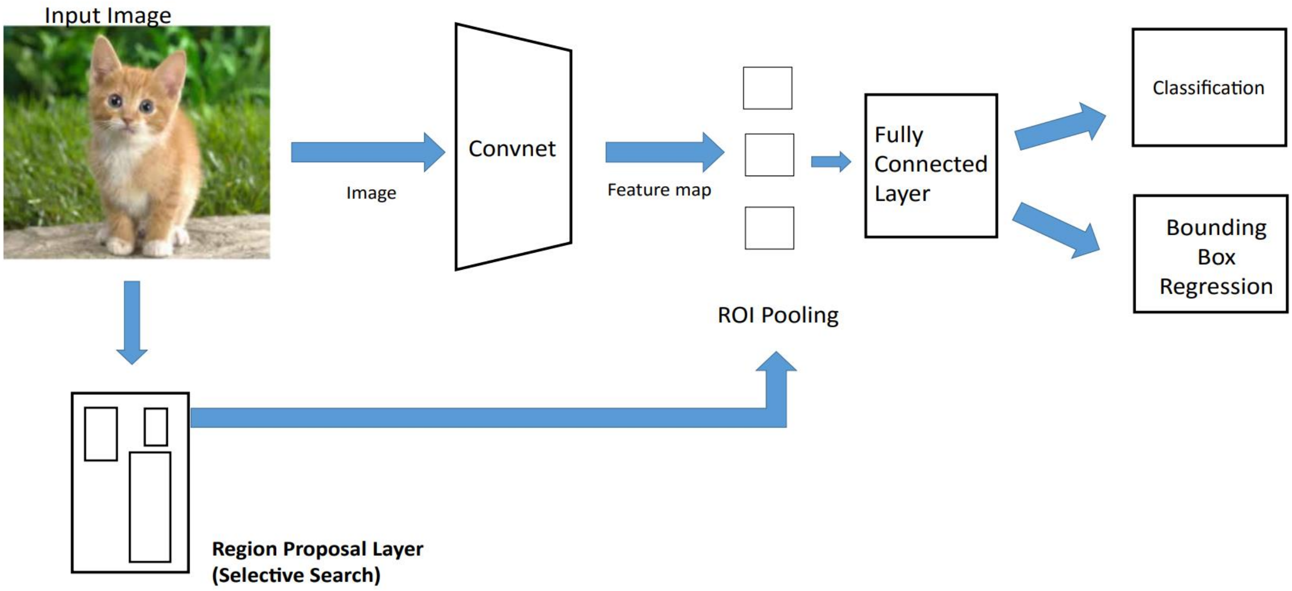
Limitations of RCNN

- Training an RCNN model is expensive and slow
- Takes a huge amount of time to train the network as there are 2000 region proposals per image.
- Cannot be implemented real time as it takes around 47 seconds for each test image.

Fast R-CNN

In fast R-CNN instead of performing maximum pooling, we perform ROI pooling for utilising a single feature map for all the regions. This warps ROIs into one single layer; the ROI pooling layer uses max pooling to convert the features.

Fast RCNN inputs an image and a group of region proposals. The convolutional network is trained on the whole image instead of individual regions. Hence the computation time is reduced from 2000 convolutions to 1 convolution.

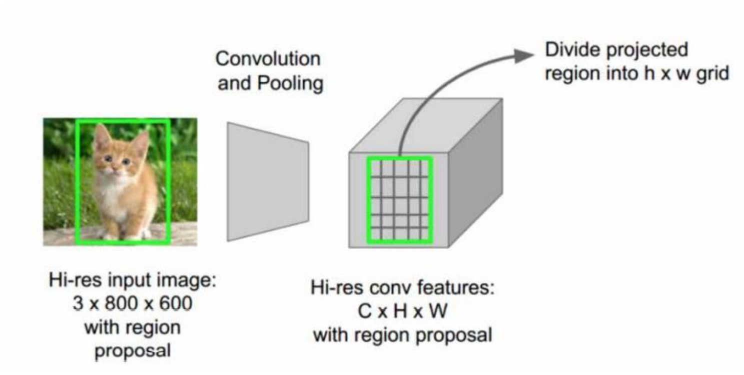


Working flow of Fast RCNN

1. First, pre-train a convolutional neural network on image classification tasks.
2. Propose regions by selective search (~2k candidates per image).
3. Alter the pre-trained CNN:
    a. Replace the last max pooling layer of the pre-trained CNN with a RoI pooling layer. The RoI pooling layer outputs fixed-length feature vectors of region proposals. Sharing the CNN computation makes a lot of sense, as many region proposals of the same images are highly overlapped.
    b. Replace the last fully connected layer and the last softmax layer (K classes) with a fully connected layer and softmax over K + 1 classes.
4. Finally, the model branches into two output layers:
    a. A softmax estimator of K + 1 classes (same as in R-CNN, +1 is the "background" class), outputting a discrete probability distribution per RoI.
    b. A bounding-box regression model which predicts offsets relative to the original RoI for each of K classes.

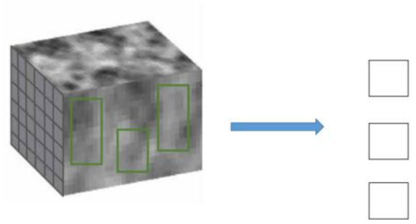
Convolutional Layers

- CNNs extract high level features from image like lines, edges, shapes.
- The Fast R-CNN detector processes the entire image. Hence we did the work of 2000 convolutions in a single step.
- Fast R-CNN chooses CNN features corresponding to each region proposal.


Convolution and Pooling
Divide projected region into h x w grid
Hi-res input image: 3 x 800 x 600 with region proposal
Hi-res conv features: C x H x W with region proposal

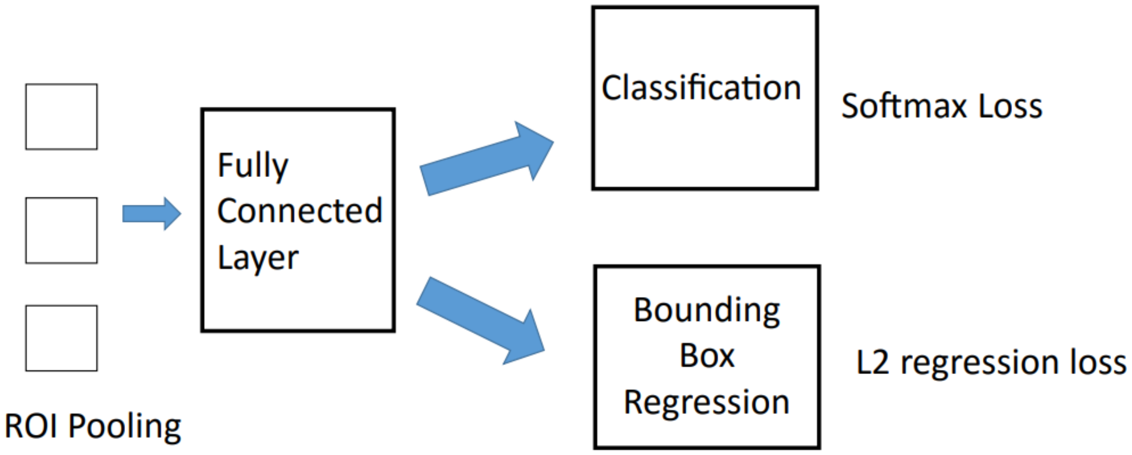
Region Proposal layer & ROI Pooling

- ROI pool layer convert these regions to a smaller feature map of fixed size before feeding to fully connected neural network.
- This is done using max pooling or average pooling.



ROI Pooling

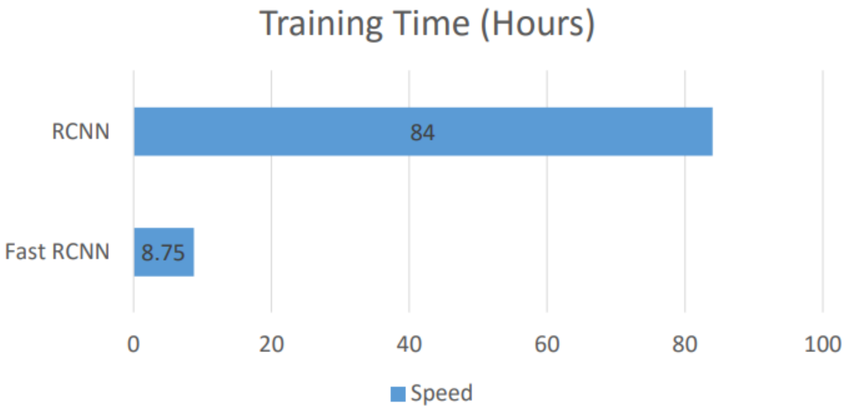
- Fully Connected layers are used to produce 2 outputs.
- Predict the class label of the object in ROI
- Regression to find (x, y, w, h) coordinates of Bounding Box
  - x, y = X and Y coordinates of the bounding box
  - w = width of bounding box
  - h = height of bounding box



Performance of Fast RCNN

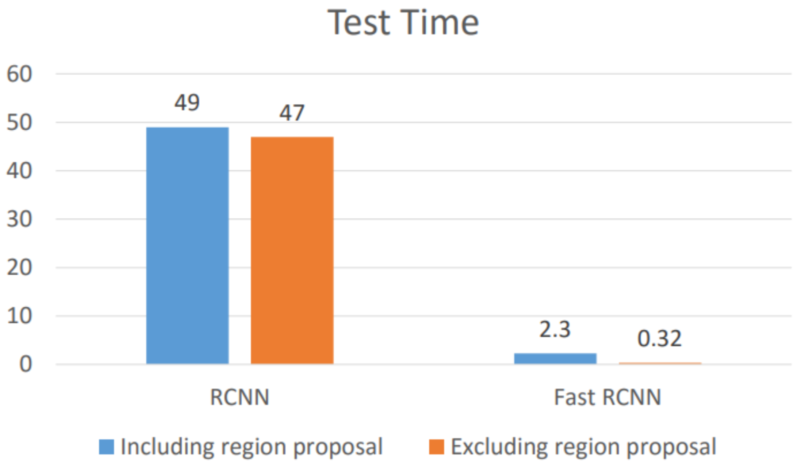
Training Performance

- Fast RCNN is10x faster compared to RCNN because
- Only a single image is processed by a CNN, instead of 2000 CNNs on images obtained from selective search in RCNN algorithm.



Test Performance

- Fast RCNN is really fast in prediction.
- The only bottleneck is region proposal which takes almost 2 seconds which is a lot of time.

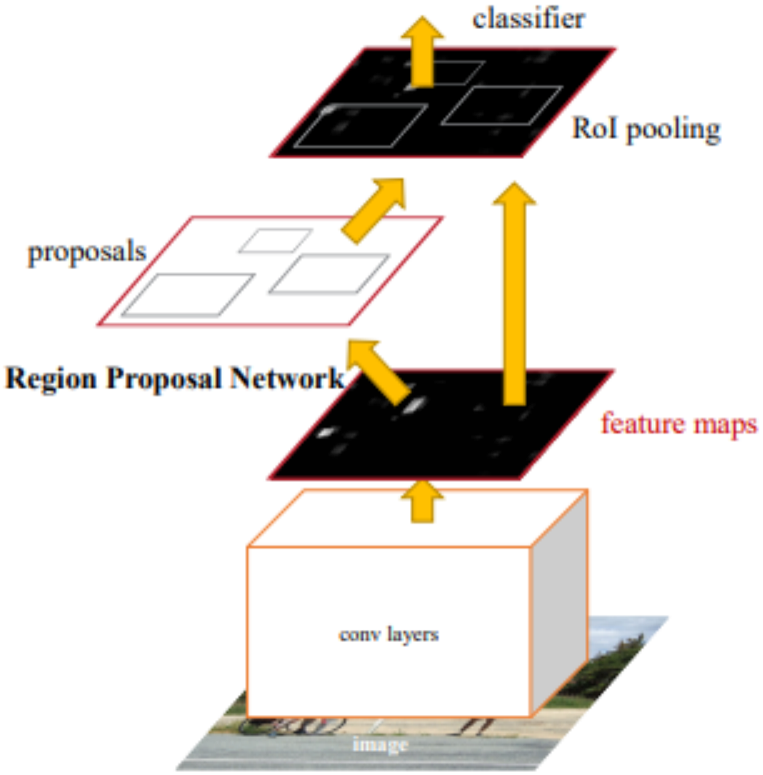


Limitations of Fast RCNN

- Region proposal takes a lot of time (2 seconds).
- Selective Search is a slow and time consuming process which is unsuitable for real world, large datasets.
- This issue is resolved with Faster RCNN.

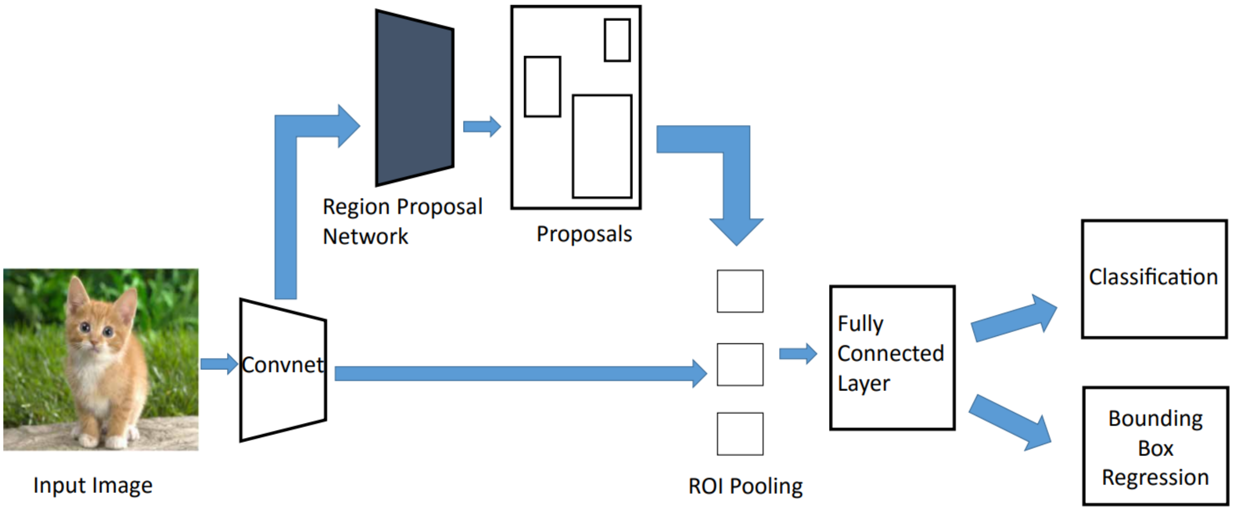
Faster R-CNN

Faster RCNN uses an inbuilt region proposal network (RPN) to generate region proposals directly in the network without using an external algorithm like Selective search.

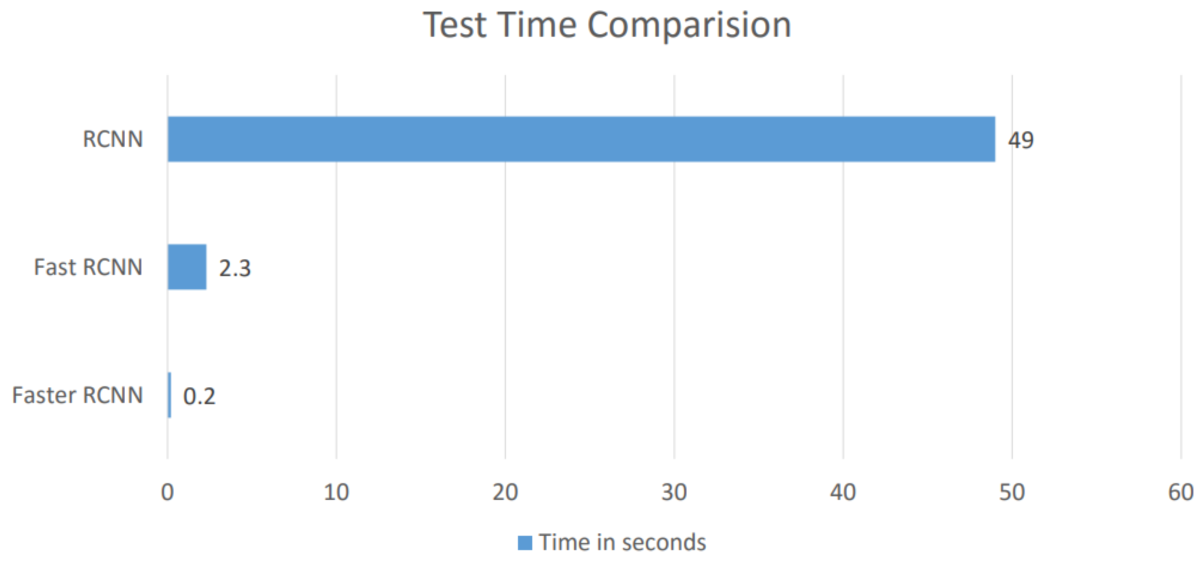


Working flow for Faster RCNN

1. Pre-train a CNN network on image classification tasks.
2. Fine-tune the RPN (region proposal network) end-to-end for the region proposal task, which is initialized by the pre-train image classifier. Positive samples have IoU (intersection-over-union) > 0.7, while negative samples have IoU < 0.3.
    a. Slide a small n x n spatial window over the conv feature map of the entire image.
    b. At the center of each sliding window, we predict multiple regions of various scales and ratios simultaneously. An anchor is a combination of (sliding window center, scale, ratio). For example, 3 scales + 3 ratios => k=9 anchors at each sliding position.
3. Train a Fast R-CNN object detection model using the proposals generated by the current RPN
4. Then use the Fast R-CNN network to initialize RPN training. While keeping the shared convolutional layers, only fine-tune the RPN-specific layers. At this stage, RPN and the detection network have shared convolutional layers!
5. Finally fine-tune the unique layers of Fast R-CNN
6. Step 4-5 can be repeated to train RPN and Fast R-CNN alternatively if needed.



Performance of Faster RCNN



Comparing R-CNN, Fast R-CNN and Faster R-CNN

Now, let us compare the important features of all these models that we have gone through.

	R-CNN	Fast R-CNN	Faster R-CNN
region proposals method	Selective search	Selective search	Region proposal network
Prediction timing	40-50 sec	2 seconds	0.2 seconds
computation	High computation time	High computation time	Low computation time
The mAP on Pascal VOC 2007 test dataset(%)	58.5	66.9 (when trained with VOC 2007 only) 70.0 (when trained with VOC 2007 and 2012 both)	69.9(when trained with VOC 2007 only)

The mAP on Pascal VOC 2012 test dataset (%)	53.3	65.7 (when trained with VOC 2012 only)
		68.4 (when trained with VOC 2007 and 2012 both)
			67.0(when trained with VOC 2012 only)
			70.4 (when trained with VOC 2007 and 2012 both)
			75.9(when trained with VOC 2007 and 2012 and COCO)

Summary

- Object detection is one of the fundamental problems of computer vision. It forms the basis of many other downstream computer vision tasks, for example, instance segmentation, image captioning, object tracking, and more.
- An input image given to the R-CNN model goes through a mechanism called selective search to extract information about the region of interest.
- After selection of the region, the image with regions goes through a CNN where the CNN model extracts the objects from the region.
- In fast R-CNN instead of performing maximum pooling, we perform ROI pooling for utilising a single feature map for all the regions. This warps ROIs into one single layer; the ROI pooling layer uses max pooling to convert the features.
- Faster R-CNN possesses an extra CNN for gaining the regional proposal, which we call the regional proposal network.