# Object Detection: R-CNN, Fast R-CNN, and Faster R-CNN

**CS 5330 Fall 2024**

**Teams: Neural Nets**

**Team Members:**

**YiJie Cao, Jiaqi Liu, Hao Niu, Yue Zhang**

**Oct 24 2024**

# Contents

## Part One

## 1. Introduction:

Object detection is a computer vision technique for locating instances of objects in images or videos[1]. It goes beyond simple image classification (which only says "what" is in an image) by also determining "where" the objects are. This task is crucial in various applications, such as autonomous driving, where the car must detect other cars, pedestrians, and obstacles, security systems, where objects of interest like faces or dangerous items are detected, and even in healthcare, where abnormal regions are identified in medical scans.

There are three key architectures in the evolution of object detection using deep learning:

- R-CNN (Region-based Convolutional Neural Network)
- Fast R-CNN
- Faster R-CNN

These architectures revolutionized object detection by applying convolutional neural networks (CNNs), which are great at extracting features from images.

## 2. The Working Principles and Evolution of R-CNN and Its Improved Models

### 2.1 R-CNN: Region-based Convolutional Neural Network

R-CNN (Region-based Convolutional Neural Network) was an epoch-making model in 2013 which successfully combined CNN with classical computer vision techniques for object detection and broke the previous record[2].

**How R-CNN Works:**

- Input Image and Region Proposals:

R-CNN starts by taking an input image and generating around 2000 region proposals using a technique called Selective Search. Each region proposal is a part of the image that could contain an object.

Example: Imagine an image of a crowded street. R-CNN will divide the image into many small areas, some containing cars, pedestrians, or street signs. Selective Search tries to generate these areas, but this step is slow.

- Feature Extraction with CNN:

Each region proposal is then resized to the same dimension and passed through a pre-trained Convolutional Neural Network (CNN) to extract features like edges, shapes, and textures.

Example: Each region containing a pedestrian or car is resized and analyzed by the CNN, which extracts features to understand what might be inside (e.g., the shapes of a car or the outline of a person).

- Classification and Bounding Box Regression:

After feature extraction, the model uses a classifier (like a Support Vector Machine, SVM) to decide whether the region contains an object, and if so, what type (e.g., car, pedestrian, or bike). It also adjusts the bounding box to improve accuracy.

### 2.2 Fast R-CNN: A Faster Approach

In 2013, Ross Girshick et al. introduced R-CNN, an object detection model that combined convolutional layers with existing computer vision techniques, breaking previous records. It was a groundbreaking model at the time. In 2015, Ross Girshick developed Fast R-CNN, setting a new record. It was more accurate, and the inference speed became 213 times faster[3].

**How Fast R-CNN Works:**

- Shared Feature Extraction:

Instead of applying a CNN to each region proposal individually, Fast R-CNN first runs the CNN on the entire image to create a feature map. This feature map contains all the information needed for object detection.

Example: If you input an image of a street, the CNN processes the entire image once and generates a feature map that contains information about the objects in all parts of the image.

- Region of Interest (RoI) Pooling:

After generating the feature map, Region of Interest (RoI) Pooling is applied to extract fixed-size feature maps for each region proposal from the feature map. This step allows the network to reuse the same feature map for all region proposals, making it faster.

Example: The feature map for the street image will contain regions for the cars, pedestrians, and street signs. RoI Pooling extracts each region proposal from this map, resizing them to a uniform size for further processing.

- Classification and Bounding Box Regression:

The extracted region features are passed through fully connected layers to classify the objects and refine the bounding box coordinates. This step is done in one go, unlike R-CNN, which performed it in separate stages.

### 2.3 Faster R-CNN: Solving the Speed Problem

Faster R-CNN is a deep convolutional network used for object detection, that appears to the user as a single, end-to-end, unified network. The network can accurately and quickly predict the locations of different objects[4].

**How Faster R-CNN Works:**

- Region Proposal Network (RPN):

Faster R-CNN adds a new network called the Region Proposal Network (RPN), which runs alongside the main CNN. The RPN looks at the feature map and predicts region proposals directly, without needing Selective Search.

Example: Instead of manually searching for objects, the RPN automatically suggests areas in the feature map that likely contain objects, such as a pedestrian or car.

- Shared Feature Map:

Both the RPN and the object detection head share the same feature map, so there's no need to re-process the image.

- RoI Pooling and Classification:

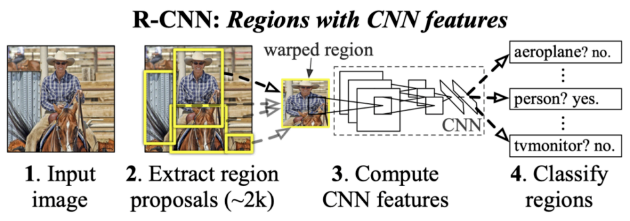
Similar to Fast R-CNN, Faster R-CNN uses RoI Pooling to resize the proposed regions. These regions are then classified and adjusted to find the most accurate bounding boxes.

## 3. Technical Details

### 3.1 R-CNN (2014) - Pioneering Object Detection Model

The year 2014 marked a pivotal moment in the field of computer vision with the publication of the R-CNN paper. This paper not only demonstrated the potential of Convolutional Neural Networks (CNNs) for achieving high performance in object detection but also addressed a challenging problem: the precise localization of objects within images through the creation of bounding boxes. While CNNs had already gained prominence for image classification tasks, object detection required a more intricate solution. This is where the R-CNN algorithm entered the scene, offering a novel approach to this complex problem[5].

- **R-CNN Technical Implementation**



Architectural Components:

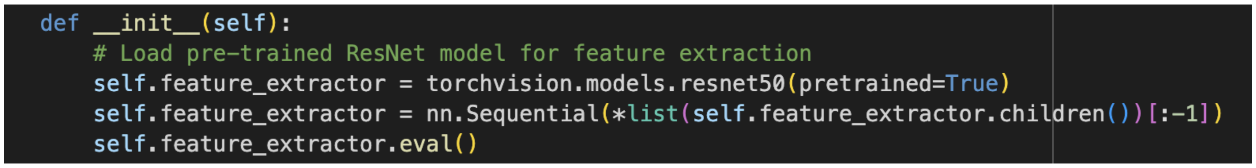
➢ Step 1: Region Proposal Network (RPN)

The key innovation of R-CNN is the use of the Selective Search algorithm to generate potential object regions. Instead of using a dense sliding window approach across the entire image, Selective Search generates a set of candidate regions by merging pixels with similar colors, textures, and intensities. This greatly reduces computational costs while maintaining high detection accuracy.

Selective Search Algorithm: It generates multiple possible candidate regions through image segmentation and then merges these regions using a hierarchical aggregation algorithm, outputting bounding boxes of different scales. Selective Search generates around 2000 regions for each image, which are then fed into the feature extraction network for further processing.

➢ Step2: Feature Extraction Network

R-CNN uses a deep convolutional neural network (such as ResNet50) to extract high-level semantic features from images. Through multiple layers of convolution and pooling operations, the network encodes the input image into a feature vector with rich representational power.

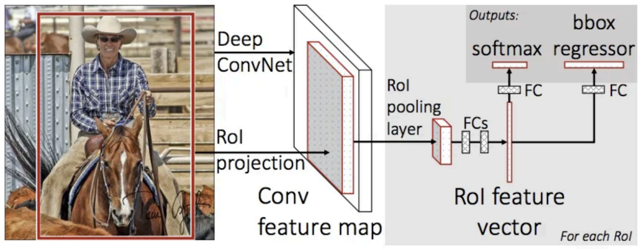


```
def __init__(self):
    # Load pre-trained ResNet model for feature extraction
    self.feature_extractor = torchvision.models.resnet50(pretrained=True)
    self.feature_extractor = nn.Sequential(*list(self.feature_extractor.children())[:-1])
    self.feature_extractor.eval()
```

ResNet50 Architecture: ResNet addresses the common issue of vanishing gradients in deep networks by introducing the residual network structure, allowing the network to easily train models with over 100 layers. R-CNN uses this network by removing the final fully connected classification layer, keeping only the convolutional layers as the feature extractor.

Feature Output: The output for each region is a 2048-dimensional feature vector, which captures high-level semantic information of the object for subsequent use by the classifier and regressor.

## 3.2 Fast R-CNN Technical Implementation

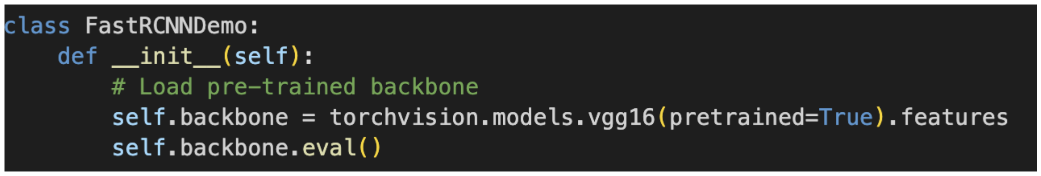


Fast R-CNN

Fast R-CNN is a major improvement over R-CNN, significantly increasing speed by performing a single forward pass on the entire image. R-CNN requires extracting features for each candidate region individually, while Fast R-CNN generates a unified feature map for the entire image and then uses an ROI pooling layer to process the features for each region.

➢ Single Forward Pass Design

The key to Fast R-CNN is using a deep convolutional neural network (such as VGG16) to generate the feature map for the entire image in one pass. This allows each candidate region to be processed directly on this feature map, without the need to recompute features for each region individually.

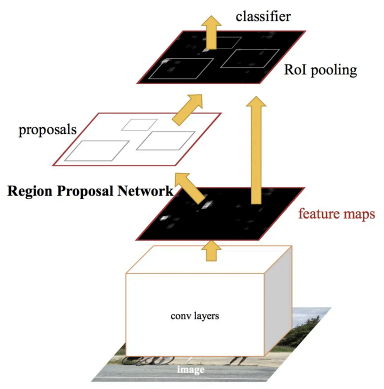

```
class FastRCNNDemo:
    def __init__(self):
        # Load pre-trained backbone
        self.backbone = torchvision.models.vgg16(pretrained=True).features
        self.backbone.eval()
```

VGG16 Backbone Network: VGG16 is a classic deep convolutional network architecture. Through deep convolutional and pooling layers, it extracts features from the image. Although the network is relatively simple, its depth makes it well-suited for capturing multi-level information from the image.

➢ RoI Pooling Layer

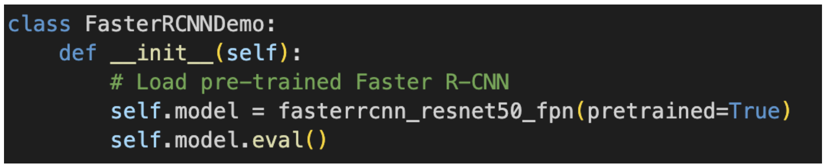
The RoI pooling layer is another key improvement in Fast R-CNN. It addresses the issue of different-sized candidate regions. RoI pooling maps regions of different sizes to a fixed-size feature map (e.g., 7×7), allowing the subsequent classification and bounding box regression steps to operate on inputs with consistent dimensions.

## 3.3 Faster R-CNN Technical Implementation



Faster R-CNN

Faster R-CNN further improves speed by introducing the Region Proposal Network (RPN), replacing the Selective Search used in Slow R-CNN and Fast R-CNN. This improvement allows region proposals and feature extraction to share the same convolutional feature map, significantly enhancing efficiency.



```
class FasterRCNNDemo:
    def __init__(self):
        # Load pre-trained Faster R-CNN
        self.model = fasterrcnn_resnet50_fpn(pretrained=True)
        self.model.eval()
```

➢ Multi-Scale Feature Extraction:

Through FPN's multi-scale architecture, the model can capture features from different levels, which is especially useful for detecting objects with significant size differences.

➢ Region Proposal Network (RPN):

The RPN performs dense sampling across the feature map using a sliding window to generate a set of candidate anchor boxes. It replaces the traditional Selective Search algorithm, greatly speeding up the region proposal process.

The inference process of Faster R-CNN consists of several key steps, including image preprocessing, feature extraction, region proposal, RoI pooling, classification and regression, followed by post-processing to output the final detection results.

### 3.4 Performance Comparison and Experimental Results

Faster R-CNN significantly surpasses the previous two versions in both speed and accuracy. The experimental results are as follows:

| **Model** | **Processing Time per Image** | **Number of Region Proposals** |
|---|---|---|
| R-CNN | 47 seconds | 2000 |
| Fast R-CNN | 0.32 seconds | 2000 |
| Faster R-CNN | 0.2 seconds | 300 |

| **Model** | **PASCAL VOC 2007 mAP** |
|---|---|
| R-CNN | 58.5% |
| Fast R-CNN | 70.0% |
| Faster R-CNN | 73.2% |

Speed: Faster R-CNN processes each image in just 0.2 seconds, whereas R-CNN takes 47 seconds.

Accuracy: On the PASCAL VOC 2007 dataset, Faster R-CNN achieves a mAP of 73.2%, which is notably higher than R-CNN's 58.5% and Fast R-CNN's 70.0%.

## 4. Applications of R-CNN, Fast R-CNN, and Faster R-CNN

R-CNN, Fast R-CNN and Faster R-CNN are powerful object detection frameworks used in all kinds of fields because they're good at detecting and classifying objects in the image. These models have huge usage in real world applications where object detection must be accurate and robust.

- Autonomous Cars and Driver-Assistant Devices

Object is one of the most important aspects for making decisions in autonomous cars. Faster R-CNN will identify cars, pedestrians, traffic lights and lane markers in order to stay safe while driving. Even driver assistance systems like advanced collision warning use these architectures for rapid object detection and tracking in a complicated environment.

- Surveillance and Security Systems

Public places and secure zones are monitored using R-CNN models from security systems. Fast R-CNN and Faster R-CNN let you look in real time on surveillance video and find intruders, weapons, or abnormal activities. This automation increases security as early warnings enable fast responses to any attack.

- Medical Imaging

In medicine, R-CNN models help radiologists detect abnormalities on medical images (X-rays, CT scans, MRI). Faster R-CNN has already been used to find tumours, to cut down lesions and to diagnose diseases allowing us to identify the problems early and make better decisions for patients with more accurate diagnosis.

- Retail and Inventory Management

Stores use Faster R-CNN to manage inventory. By pointing to empty shelves or products lost on the shelf using visual inspection tools, such models enable best-in-class stock management and elimination of unnecessary operations. Shop Automation using Object Recognition automates processes and enhances customer service.

- Agriculture and Precision Farming

We also have R-CNN based farms for crop monitoring and pest monitoring. Drones fitted with Faster R-CNN process acreage extracted from aerial images and pick up weeds, diseases or insects. This helps farmers to act early and maximize yields, and promote sustainable agriculture through precision agriculture.

- Industrial Automation and Robotics

Fast R-CNN and Faster R-CNN in manufacturing are placed inside automated production lines to identify parts and control quality. Robots using such models can recognise, grasp and assemble parts accurately to increase production speed and minimize factory errors.

- Wildlife Monitoring and Conservation

Object detection based on R-CNN are applied for wildlife surveillance and illegal hunting. Camera traps and drones with these structures will track animals and their movements in sanctuaries. And they're useful for keeping conservationists up to date on threatened species, which in turn protects biodiversity.

- Sports Analytics

Faster R-CNN is used in sports to follow athletes and ball in real time. It allows for performance breakdown in real-time to coaches and analysts. This same technology also drives broadcast capabilities that allows for live stats and live replays on-the-go.

- E-commerce and Visual Search Engines

Visual search engines in the eCommerce industry use Fast R-CNN to pick and rank products based on images uploaded by customers. It provides the option for "search by image", which enhances the shopping experience as the product can be quickly discovered by users and attracts traffic with easy search results.

Lastly, R-CNN, Fast R-CNN, and Faster R-CNN are object detectors that have changed the face of everything from driving without human help, to medical, retail, and industrial automation. Their capacity to provide true-to-life detection made them vital tools in real-world problems, decision-making and process optimization in many industries. R-CNN with high speeds and accuracies, is especially suitable for applications that require high speed.

## 5. Strengths and Limitations of R-CNN Models

### 5.1 For R-CNN Strengths and Limitations

- **Strengths:**

Accurate Object Detection:

R-CNN achieves high accuracy in object detection by leveraging region-based convolutional features. This approach is particularly effective for scenarios that require precise object localization and recognition.

Robustness to Object Variations:

R-CNN models are capable of handling objects with varying sizes, orientations, and scales. This makes the architecture well-suited for real-world applications where objects are diverse and appear in complex environments.

Flexibility:

The R-CNN framework is highly adaptable and can be modified to suit a variety of object detection tasks, such as instance segmentation and object tracking. By adjusting the network's final layers, R-CNN can be tailored to different use cases and datasets.

- **Limitations:**

Computational Complexity:

R-CNN is computationally expensive, as it requires the extraction of region proposals, applying a CNN to each proposal, and running the resulting features through a classifier. This multi-step process can be slow and demands significant computational resources.

Slow Inference:

Due to the sequential processing of region proposals, R-CNN has slow inference times. For real-time applications, this latency may be unacceptable, limiting its use in scenarios where speed is crucial.

Overlapping Region Proposals:

R-CNN may generate multiple overlapping region proposals, leading to redundant computations. This can impact both efficiency and detection performance, as the model spends unnecessary resources on processing overlapping areas.

Not an End-to-End Model:

Unlike more modern architectures, such as Faster R-CNN, R-CNN is not end-to-end. It involves separate modules for region proposal and classification, which can result in suboptimal performance compared to models that jointly optimize both tasks in a unified framework.

## 5.2 For Fast R-CNN Strengths and Limitations

- **Strengths:**

Speed and Efficiency:
Fast R-CNN significantly improves detection speed by integrating region proposal generation, feature extraction, classification, and regression into a single network. This end-to-end approach eliminates the need for the multi-stage training process required by its predecessor, R-CNN.

End-to-End Training:
The combination of these steps into one framework allows for unified, end-to-end training. This improves model efficiency and simplifies the overall pipeline, avoiding the need for separate training stages.

ROI Pooling Layer:
Fast R-CNN introduces the ROI Pooling layer, which resolves the issue of variable-sized candidate regions by resizing them to a fixed size. This allows for seamless processing of different-sized objects and regions through the network.

- **Limitations:**

Sensitive to Background Clutter:
Fast R-CNN may struggle in environments with significant background noise. Its classification accuracy tends to degrade when faced with noisy or cluttered regions, leading to potential misclassifications.

Difficulty Handling Objects of Varying Scales:
The model has difficulty with objects that vary significantly in scale, especially with smaller objects, which can lead to reduced detection accuracy.

Dependence on Region Proposal Techniques:
While Fast R-CNN improves performance, it still relies on region proposal techniques like Selective Search, which can be computationally expensive and act as a bottleneck in the detection process.

Overabundance of Negative Samples:
The model may encounter too many negative samples during training, which can adversely affect its performance. Techniques such as hard negative mining and alternate training strategies have been proposed to mitigate this issue.

Improvements via Advanced Architectures:
Various improvements have been made by adopting newer architectures, such as using MobileNet for the base convolutional layers or adding context-aware ROI Pooling. These updates have shown enhancements in both detection accuracy and inference speed.

## 5.3 For Faster R-CNN Strengths and Limitations

- **Strengths:**

Superior Performance:
Faster R-CNN achieves high detection accuracy by utilizing a two-stage network, which includes a Region Proposal Network (RPN) to improve detection performance.

Two-Stage Network:
Compared to single-stage networks, Faster R-CNN's two-stage approach offers better precision and is more effective at handling multi-scale and small object detection tasks.

Versatility and Robustness:
Faster R-CNN performs exceptionally well across different datasets and can be easily adapted to various object classes, demonstrating its robustness and flexibility for a wide range of applications.

Plenty of Optimization Potential:
This model has significant room for improvement and optimization, making it a suitable choice for researchers to explore new algorithmic optimizations and advancements.

Comprehensive Code Availability:
Faster R-CNN has been implemented in various deep learning frameworks, with open-source code readily available, making it convenient for use and adaptation.

- **Limitations:**

Single-Layer Convolutional Feature Maps:
Whether using VGGNet or ResNet, Faster R-CNN’s feature extraction network generates single-layer feature maps, which often have low resolution. This presents challenges for detecting small objects or handling multi-scale problems. Combining multi-layer feature maps or increasing resolution are potential areas for improvement.

Non-Maximum Suppression (NMS):
In RPN, NMS is used to eliminate overlapping candidate proposals based on their classification scores. However, this approach is not ideal for detecting occluded objects, as one of the overlapping proposals may be incorrectly filtered out. Improving the NMS process could enhance detection performance.

RoI Pooling Precision Loss:
Faster R-CNN’s original RoI Pooling method involves two rounds of quantization, which can lead to a loss in precision. Later models, like Mask R-CNN, introduced RoI Align to address this issue, improving detection accuracy.

Fully Connected Layers:
The final fully connected layers in Faster R-CNN contribute significantly to the model’s parameter size, and each RoI is processed individually through these layers without shared computations. Exploring ways to replace these fully connected layers with more lightweight alternatives could help reduce the model’s complexity.

Positive and Negative Sample Imbalance:
In both the RPN and RCNN stages, hyperparameters are used to limit the number of positive and negative samples to maintain balance. For instance, the number of positive samples is limited to a maximum of 64 during proposal generation, and negative samples are chosen at three times the number of positive samples. While this approach ensures balance, it may not always be optimal for all tasks and datasets, making it an area worth further research.

Two-Stage Network Complexity:
Although the two-stage network in Faster R-CNN provides higher accuracy, it is significantly slower compared to single-stage networks. This makes it unsuitable for real-time applications. Balancing network depth for faster performance while retaining accuracy is a key area of exploration.

## 6. References

[1]“What Is Object Detection?,” Mathworks.com, 2019. https://www.mathworks.com/discovery/object-detection.html
[2]“KiKaBeN - R-CNN: Region-based Convolutional Neural Network,” KiKaBeN, Jun. 11, 2022. https://kikaben.com/r-cnn-original/
[3]“KiKaBeN - Fast R-CNN: Understanding why it’s 213 Times Faster than R-CNN and More Accurate,” *kikaben.com*, Jun. 21, 2022. https://kikaben.com/fast-r-cnn-213/
[4]“Faster R-CNN Explained for Object Detection Tasks | DigitalOcean,” *Digitalocean.com*, 2024. https://www.digitalocean.com/community/tutorials/faster-r-cnn-explained-object-detection
[5]P. P. Sep 25, “What is R-CNN?,” *Roboflow Blog*, Sep. 25, 2023. https://blog.roboflow.com/what-is-r-cnn/