

**PROJECT REPORT**  
On  
**IMAGE SEGMENTATION USING FASTER-RCNN**

*Submitted in partial fulfillment of the requirement for the degree of*

**Bachelor of Technology**  
**In**  
**Computer Science Engineering**



**SUBMITTED BY:**

Akshat Upadhyay (17103039)

Harsh Pandey (17103043)

Shashwat Singh (17103047)

**UNDER THE SUPERVISION OF:**

Dr. Anuja Arora

**ASSOCIATE PROFESSOR**

Dept. of C.S.E, JIIT, Sec-62, Noida

Department of Computer Science & Information Technology  
JAYPEE INSTITUTE OF INFORMATION TECHNOLOGY, NOIDA

## **Table Of Contents**

1.	Certificate of Declaration.....	3
2.	Acknowledgement .....	4
3.	Introduction	
3.1.	General Overview .....	5
3.2.	Real-Life Use Case Scenario.....	5
3.3.	Problem Statement.....	6
3.4.	Feasibility.....	6
3.5.	Scope of Applications.....	6
4.	Research Papers Cited	
4.1.	Real Time object detection using R-CNN.....	7
4.2.	Real Time object detection using Faster R-CNN.....	9
4.3.	Reason Proposal Network.....	10
4.4.	Fully Convolution Network for Semantic Segmentation.....	12
5.	Implementation Details	
5.1.	Project Workflow.....	14
5.2.	Project Architecture.....	15
5.3.	Mask Generation On the RoI .....	16
5.4.	Intermediate Generated binary mask.....	16
6.	Differences And Conclusion.....	17
7.	Applications.....	17
8.	Implementation Results.....	18
9.	Tools and Technologies.....	19
10.	References.....	20

## CERTIFICATE

This is to certify that Akshat Upadhyay (Enroll. No. 17103039), Harsh Pandey (Enroll. No. 17103043) , Shashwat Singh (Enroll. No. 17103047) have successfully completed the project titled **“IMAGE SEGMENTATION USING FASTER-RCNN”** under my supervision and guidance in the fulfillment of requirements of Fifth Semester, Bachelor of Technology (Computer Science & Engineering) of Jaypee Institute of Information Technology, Sec-62, Noida .

Date: \_\_\_\_\_

\_\_\_\_\_  
Assoc. Professor Dr. Anuja Arora  
Dept. Of C.S.E, JIIT, Noida  
(Project Mentor)

## **ACKNOWLEDGEMENT**

We deem it a pleasure to acknowledge our sense of gratitude to our project guide Assoc. Prof. Dr. Anuja Arora under whom we have carried out the project work. Her incisive and objective guidance and timely advice encouraged us with constant flow of energy to continue the work. We also wish to thank her from the bottom of our hearts for guiding us and sharing with us her vast knowledge at points where we found ourselves lost in the project.

We would like to express our gratitude towards our parents & faculty members of the Institute for their kind cooperation and encouragement which helped us in completion of this project.

Date: \_\_\_\_\_

Place: Jaypee Institute of Information Technology, Noida

Akshat Upadhyay

17103039

Harsh Pandey

17103043

Shashwat Singh

17103047

## **General Overview:**

The vision community has rapidly improved commodity search and economics in a short period of time. For the most part, this progress has been made in Powerful infrastructure systems such as faster / faster RCNN and fully networked (FCN) Image detection framework and semantic category.

These methods are theoretically pleasurable as It provides flexibility and visibility while reducing training and access time. The aim of this project is to provide a relatively effective framework for the instance segmentation.

Partitioning will be an immediate issue when there is a need of Correct detection of all objects in the image and accurately track each incident

Instance segmentation is a very challenging task as It combines two elements from computer vision work such as .

- (1) the classical computer vision tasks of object detection, where the goal is to classify individual objects and localize each using a bounding box,
- (2) Semantic Segmentation where the goal is to classify each pixel into a fixed set of categories without differentiating object instances

## **Real-Life Use Case Scenario:**

Semantic segmentation finds its application in various real-life scenarios such as:

1. GeoSensing – For land usage
2. For Autonomous driving
3. For Facial Segmentation
4. Fashion – Categorizing clothing items
5. Precision Agriculture

## Problem Statement:

Instance segmentation is the problem of detecting and identifying each distinct object of interest in an image. Image segmentation is the process of partitioning a **digital image** into multiple segments (**sets** of **pixels**, also known as image objects). The goal of segmentation is to simplify and/or change the representation of an image into something that is more meaningful and easier to analyze. Image segmentation is typically used to locate objects and **boundaries** (lines, curves, etc.) in images. More precisely, image segmentation is the process of assigning a label to every pixel in an image such that pixels with the same label share certain characteristics.

Instance segmentation finds its applicability in various sectors few of which are mentioned below:

- Self-Driving Cars for obstacle identification and avoidance
- Implementation of bokeh effect in smartphone cameras
- Unwanted object removal in video editing softwares
- 3D reconstruction using aerial images

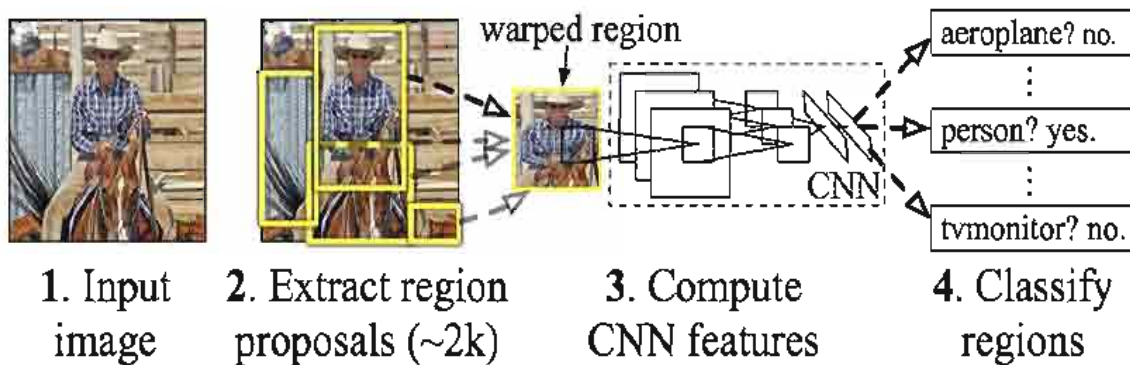
But the problem is how to precisely segment instances in an image precisely. Current instance segmentation approaches consist of a combination of modules that are trained independently of each other, thus missing opportunities for joint learning.

## Feasibility:

Earlier approaches were either of object detection or performing semantic segmentation. But in our project we are further extending these approaches to perform instance segmentation which involves both detection with classifying every pixel of the image to a particular class object. Nowadays great research is being performed in the field of Human Pose Estimation, Applying filters on images, Lane detection for self-driving cars, AR technology for creating a map of the surroundings. And hence, our approach will act as a pivot for further research and technological advancement.

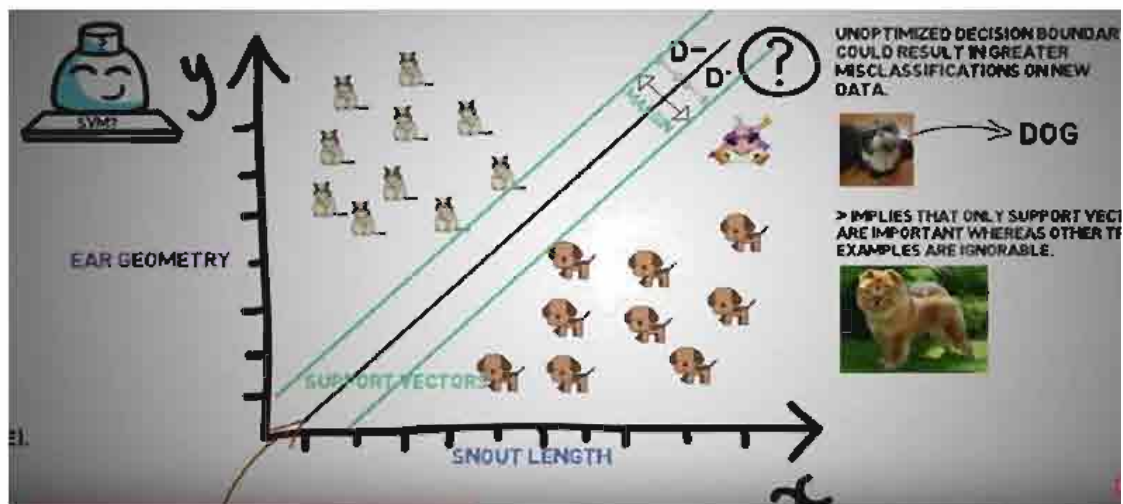
## Research papers cited:

### Real time Object Detection using R- CNN



The paper has considered 3 major modules in their object detection model:

1. First module -- generates region proposals independent of categories.
2. Second module -- large convolutional neural network which extracts fixed size feature vectors from the individual region proposed.
3. Third module -- set of class specific linear SVMs (*Support Vector Machine*).



Support Vector Machine(SVM)

## Region Proposals

They are using selective search so that they can have a check on their work with respect to various prior works done in this field.

## Feature Extraction

They first convert image data into fixed size image of  $227 \times 227$  pixel size RGB image and then this image is forward propagated through five convolutional layers and two fully connected layers. Every bounding boxes pixels are also wrapped to smaller size irrespective of their original size to new size of  $p$  pixels. Here they have used  $p = 16$ .

## Binary Classification

It's obvious that a Region proposal enclosing a desired object image should be positively classified whereas a Region proposal enclosing only natural scene background must be negatively classified. But Region Proposals which partially includes a desired object image should be either classified positively or negatively is decided using IoU (Intersection Over Union).

If IoU value is  $< 0.3$  is considered a negative example, and  $\text{IoU} > 0.3$  is considered a positive example. Various thresholds have been considered for this purpose but if the threshold is changed to 0.5 decreases mAP (mean Average Precision) by 5 points similarly decreasing it to 0.0 decreases mAP value by 4 points.

Once features are extracted image data is then passed through the third module to give final output.

## Summary

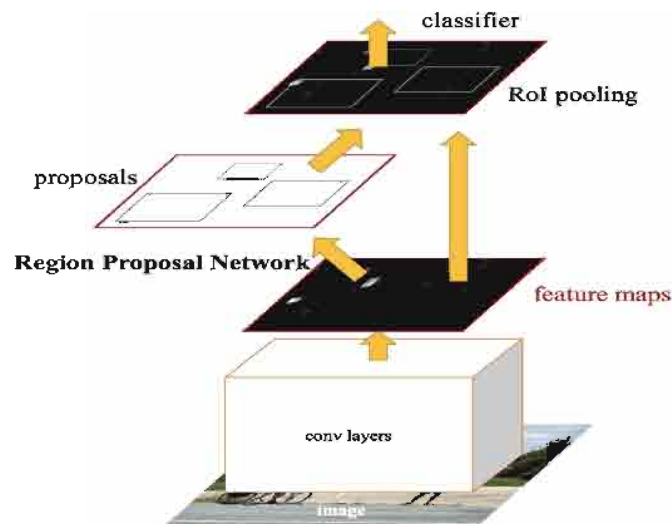
- Generate a Set of Proposals
- Runs the image through pre-trained Alexnet
- Run the box through a linear regression model

VOC 2010 test	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mAP
DPM v5 [17]	49.2	53.8	13.1	15.3	35.5	53.4	49.7	27.0	17.2	28.8	14.7	17.8	46.4	51.2	47.7	10.8	34.2	20.7	43.8	38.3	33.4
UVA [32]	56.2	42.4	15.3	12.6	21.8	49.3	36.8	46.1	12.9	32.1	30.0	36.5	43.5	52.9	32.9	15.3	41.1	31.8	47.0	44.8	35.1
Regionlets [30]	65.0	48.9	25.9	24.6	24.5	56.1	54.5	51.2	17.0	28.9	30.2	35.8	40.2	55.7	43.5	14.3	43.9	32.6	54.0	45.9	39.7
SegDPM [45]	61.4	53.4	25.6	25.2	35.5	51.7	50.6	50.8	19.3	33.8	26.8	40.4	48.3	54.4	47.1	14.8	38.7	35.0	52.8	43.1	40.4
R-CNN	67.1	64.1	46.7	32.0	30.5	56.4	57.2	65.9	27.0	47.3	40.9	66.6	57.8	65.9	53.6	26.7	56.5	38.1	52.8	50.2	50.2
R-CNN BB	71.8	65.8	53.0	36.8	35.9	59.7	60.0	69.9	27.9	50.6	41.4	70.0	62.0	69.0	58.1	29.5	59.4	39.3	61.2	52.4	53.7

Above table is the mAP (mean Average Precision) score of R-CNN model on data set VOC 2010



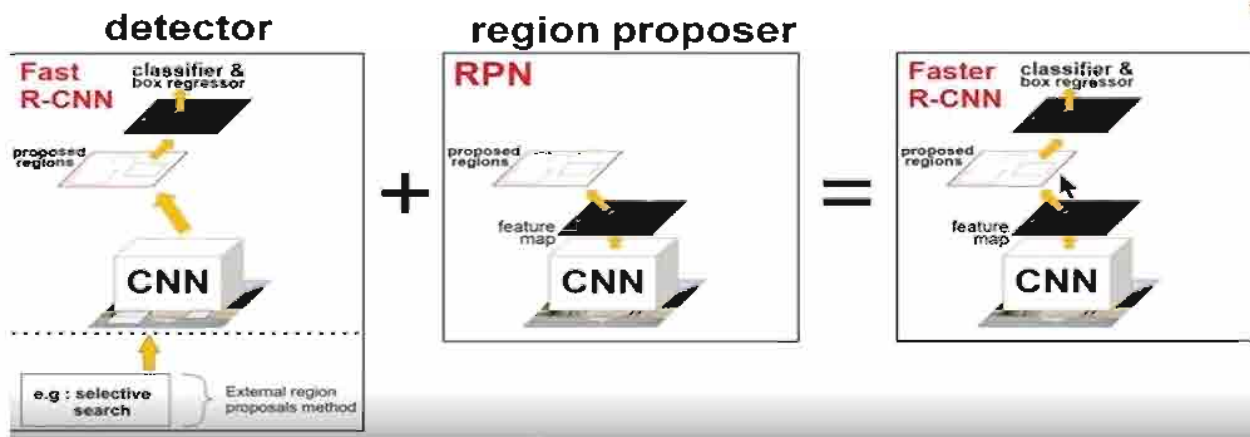
## Real time Object Detection using Faster R-CNN



In this architecture, In spite of passing image data to Region Proposal module and then applying a module of convolutional neural network they first pass image data through their convolutional layers and then through Region Proposals Network (RPN).

They have considered 3 major modules in their object detection model:

1. Convolutional Layers.
2. Region Proposal Network.
3. Fast RCNN part
  - a. Extract features using ROI pooling
  - b. Forms binary classification

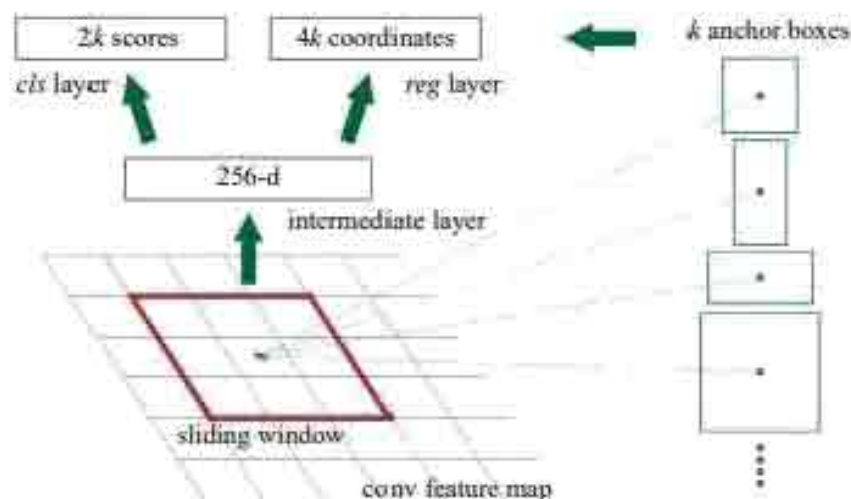


### Region Proposal Network

The time taken for regenerating regions is very less as compared to Selective search technique whereas in this module improvement of AlexNet (ZFNET) . A small window is slid over the image and the feature map generated is passed into two layers namely:

- (1) Box regression
- (2) Box classification

hence, the total number of data with this layer is  $4k$  coordinates and box-classification contains  $2k$  scores.



But here  $k$  is taken as to be 9 hence we have in total  $4 \times 9$  coordinates in regression layer and  $2 \times 9$  scores in the classification layer. To enhance the accuracy for we use the idea of Anchor boxes whose count is represented as  $k$  in the above figure



For Region Proposals containing more than one objects anchor boxes of different dimensions are tested on the region and classes are selected using the formulae:

$$\text{IoU} = \frac{\text{Anchor\_box} \cap \text{Ground\_truth}}{\text{Anchor\_box} \cup \text{Ground\_truth}}$$

If IoU > 0.7 object

If IoU > 0.3 not object

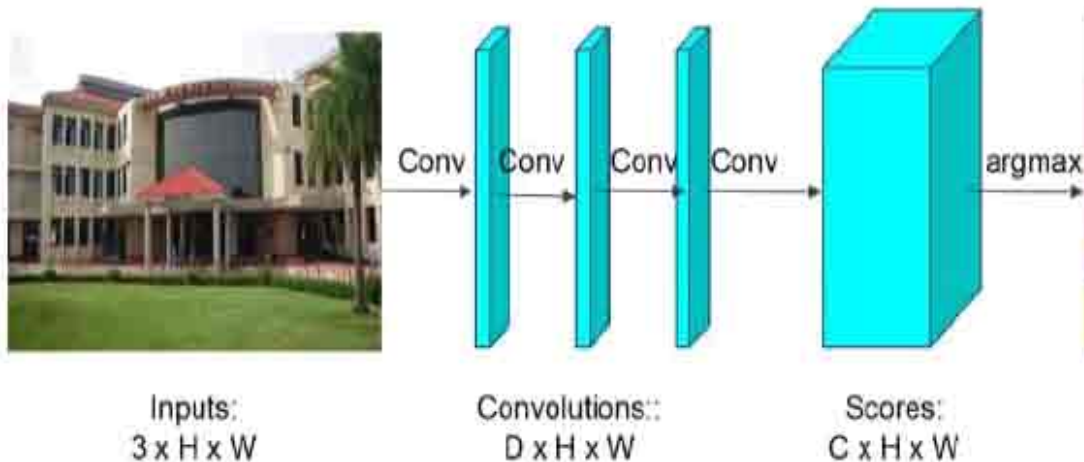
Hence in

- **Reg layer** =  $[X_{\text{mid}} \ Y_{\text{mid}} \ H \ W] * 9$
- **Cls layer** =  $[\text{Score}_{\text{yes}} \ \text{Score}_{\text{no}}] * 9$

method	# box	data	mAP	area	bird	boat	bus	bottle	bus	car	cat	chair	cow	table	dog	horse	motor	person	plant	sheep	sofa	train	tv
SS	2000	07	66.9	74.3	78.3	69.2	53.2	36.6	77.3	78.2	82.0	40.7	72.7	67.9	79.6	79.2	73.0	69.0	36.1	65.4	20.2	75.8	65.8
SS	2000	07+12	70.0	77.0	78.1	69.3	59.4	38.3	81.6	78.6	86.7	42.8	78.8	68.9	84.7	82.0	76.6	66.9	31.8	70.1	74.8	80.4	70.4
RPN*	300	07	68.5	74.1	77.2	67.7	53.9	51.0	75.1	79.2	78.9	50.7	78.0	61.1	79.1	81.9	72.2	75.9	37.2	71.4	62.5	77.4	86.4
RPN	300	07	69.9	70.0	80.6	70.1	57.3	49.9	78.2	80.4	82.0	52.2	75.3	67.2	80.3	79.8	75.0	76.3	39.1	68.3	67.3	81.1	67.6
RPN	300	07+12	73.2	76.5	79.0	70.9	65.5	52.1	83.1	84.7	86.4	52.0	81.9	65.7	84.8	84.6	77.5	76.7	38.8	73.6	73.9	83.0	72.6
RPN	300	COCO+07+12	78.8	84.3	82.0	77.7	68.9	63.7	88.1	88.4	88.9	63.6	86.3	70.8	85.5	87.5	80.1	82.3	51.6	80.4	75.8	86.6	78.9

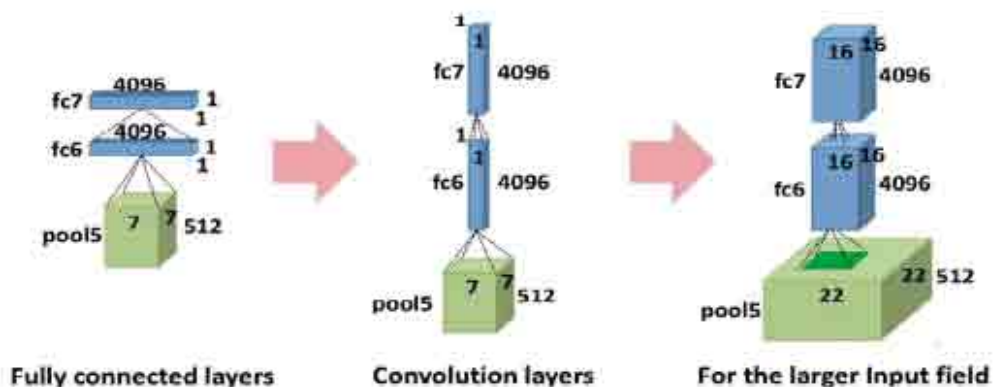
*Above table is the mAP (mean Average Precision) score of Faster R-CNN model on data set VOC 2007*

## Fully Convolutional Networks For Semantic Segmentation



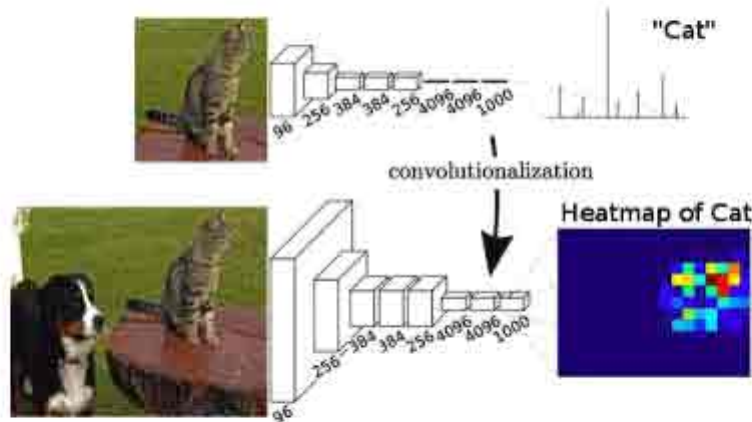
In this paper a fully convoluted network is used where the first layer is the image, with pixel size  $h \times w$ , and  $d$  color channels. Locations in higher layers correspond to the locations in the image they are path-connected to, which are called their receptive fields. Each layer of data in a convnet is a three-dimensional array of size  $h \times w \times d$ , where  $h$  and  $w$  are spatial dimensions, and  $d$  is the feature or channel dimension. Their basic components (convolution, pooling, and activation functions) operate on local input regions. An FCN naturally operates on an input of any size, and produces an output of corresponding (possibly resampled) spatial dimensions.

**Fully connected layer to Fully convoluted layer:**





All the remaining fully connected layers are converted by introducing kernels that cover their entire input region. Basically fixing the number of channels (say 4096) what we get is a  $w \times h$  feature map.



Transforming fully connected layers into convolution layers enables a classification net to output a heatmap with every pixel showing the probability of belonging to that object. But the output is coarse as it is downsampled and so are the probabilities. Therefore outputs of multiple convolutional layers go through  $1 \times 1$  convolution operation to generate multiple feature maps and then upsample them and add, this upsampling is learned.

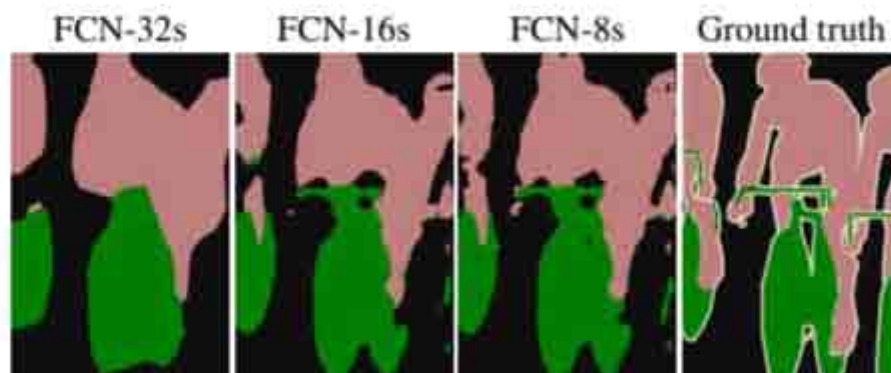


Figure 4. Refining fully convolutional nets by fusing information from layers with different strides improves segmentation detail. The first three images show the output from our 32, 16, and 8 pixel stride nets (see Figure 3).

The 32 pixel stride at the final prediction layer limits the scale of detail in the upsampled output. We address this by adding skips that combine the final prediction layer with lower layers with finer strides, combining coarse and fine layers to help the model make better predictions that are similar to the original image.

## Implementation Details:

### Workflow:

The main modules involved in Instance segmentation are:

1. Object Detection: Finding and classifying variable number of objects in an image

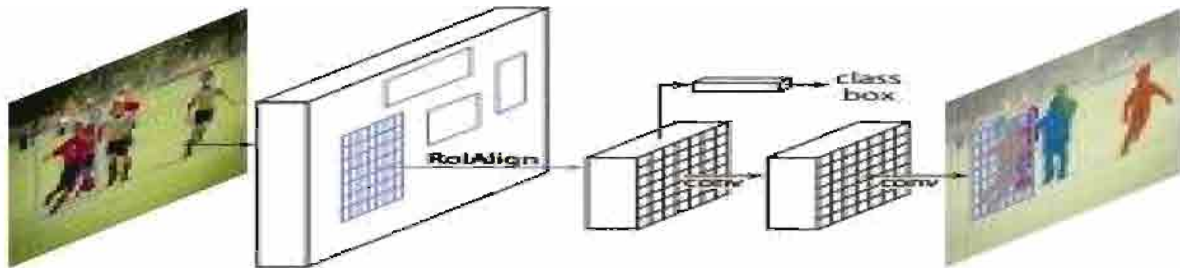


2. Semantic Segmentation: This module involves not only detecting variable number of objects in an image but also assigning object class to each pixel in the image



In the above image not only detecting motorcyclist and his bike have to explicitly provide object class to every pixel in bounding box for the motorcyclist and his bike Using the above two modules collectively on an image gives us the Instance segmentation in which bounding boxes are formed using Object Detection module and shaded mask are generated using semantic segmentation.





### **Architecture:**

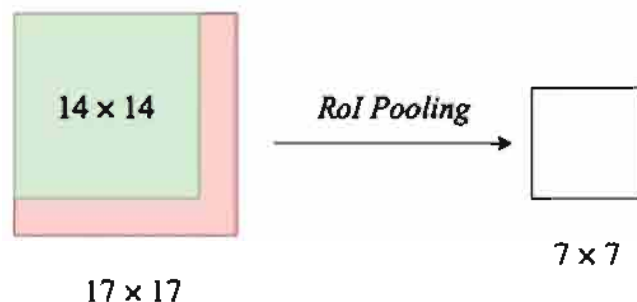
#### **Object Detection:**

We have used architecture reasonably similar to Faster R-CNN and For Semantic Segmentation Module it uses architecture similar to Fully Convoluted Network.

Like Faster R-CNN ROI(Region of Interest) i.e bounding boxes are generated using by passing the RPN network of Faster R-CNN but in order to refine the bounding boxes and applying classifier on the ROI unlike Faster R-CNN ROI ALIGN method is used rather than ROI POOLing as in ROI POOLing quantization of Rol boundaries is performed as this shatters the Robustness of the module whereas ROI Align has no interest in quantization of the boundaries of Rol Boundaries and hence led to decrement in loss of features of image.

*RoI Pooling:* Stride is quantized.

$$stride = \frac{17}{7} = 2.42 \quad stride_{RoIPool} = [2.42] = 2$$



*RoI Align*: Stride is *not* quantized.

$$stride = \frac{17}{7} = 2.42$$

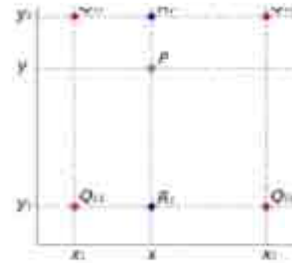


17 × 17

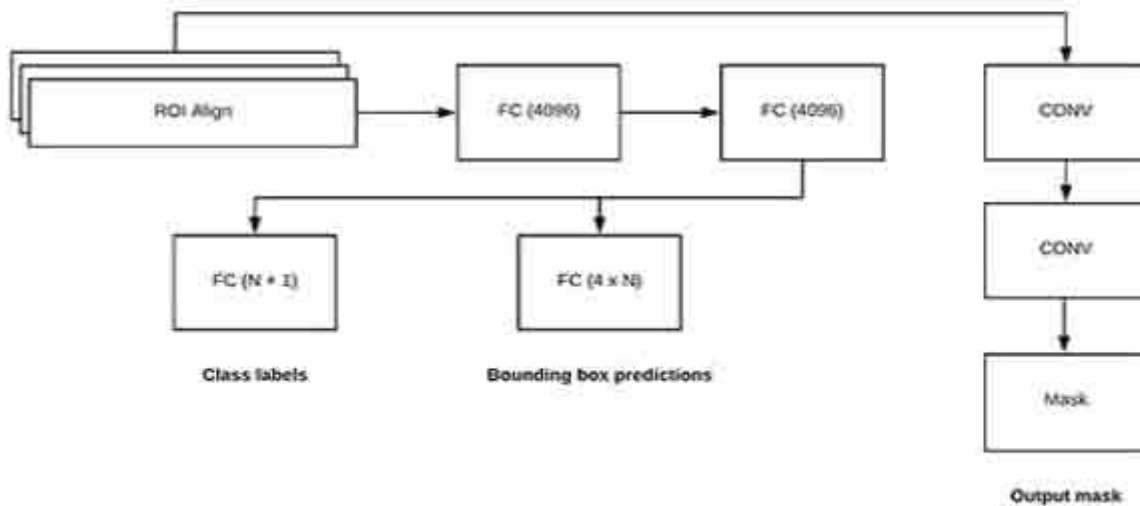
*RoI Align*



7 × 7



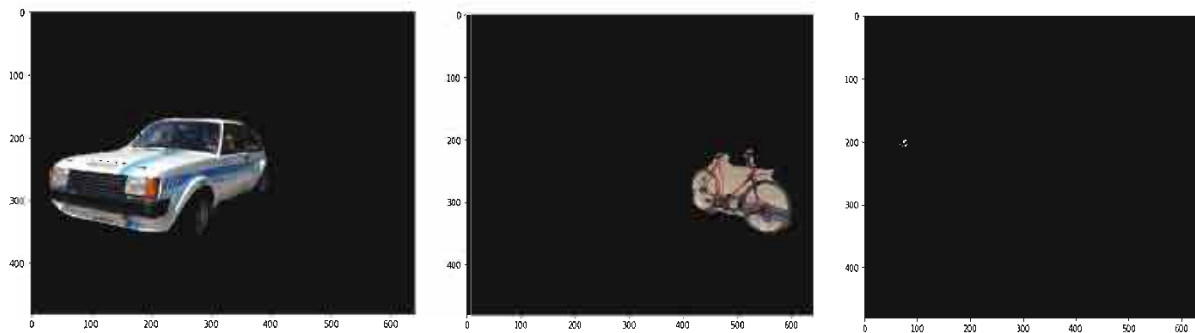
### Mask Generation Module:



We generate masks for each ROI output from an additional FCN layer which keeps the spatial structure of the masks in check as pixel-to-pixel correspondence is provided by the convolutional. As in FastR-CNN, an RoI is considered positive if it has IoU with a ground-truth box of at least 0.5 and negative otherwise. The mask loss  $L_{mask}$  is defined only on positive RoI<sup>s</sup>. The mask target is the intersection between an RoI and its associated ground-truth mask. The mask branch generate  $K$  masks where  $K$  is the number of classes predicted by classification branch



The masks obtained are often resized consistent with the input image. The mask tensors are padded, in order to avoid boundary effects caused by the scale of sample operation we had earlier. The bounding box coordinates are re-scaled according to the new mask and converted to the nearest integer. The mask is additionally re-scaled consistent with the image size, the interpolation method has been used is bi-linear. Mask threshold is a hyper parameter (default: 0.7). For each pixel, if the worth is above 0.7, the thing is assumed to be present therein pixel, else, absent. We finally get a [image\_height, image\_width] mask of the thing , similarly for all the objects of the image.











### Differences and Conclusion:

- Our approach is considerably simple and faster.(can run on video input)
- Using ROI align gives better results than ROI pool as images are better localized on pixel-level.
- Given the effectiveness of Mask R-CNN for extracting object bounding boxes, masks, and keypoints, we expect it to be an effective framework for other instance-level tasks.

### Applications demonstrated:

1. We have demonstrated the usage of this technique for applications in street view surveillance of CCTV Cameras in real time.
2. Using OpenCV we have demonstrated the use of our segmentation technique to apply a color pop effect on the most major object in the input which can also be changed manually.
3. The project also demonstrates the application of segmentation in user fed video files, which provides the complete mask generated video as output.

## Implementation results:

INPUT IMAGE	OUTPUT IMAGE
	
	
	
	

## Tools and Technologies:

**Python:** It is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python's design philosophy emphasizes code readability with its notable use of significant whitespace. Its language constructs and object oriented approach aim to help programmers write clear, logical code for small and large-scale projects.

**IPython (Interactive Python):** It is a command shell for interactive computing in multiple programming languages, originally developed for the Python programming language, that offers introspection, rich media, shell syntax, tab completion, and history.

**Project Jupyter:** Spun-off from IPython in 2014 by Fernando Pérez, It is a web-based interactive computational environment for creating Jupyter notebook documents. A Jupyter Notebook document is a JSON document, following a versioned schema, and containing an ordered list of input/output cells which can contain code, text(usingMarkdown), mathematics, plots and rich media.

**NumPy:** it is a library for the Python programming language, adding support for large, multidimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.

**Pandas:** is an open source, BSD-licensed library providing high-performance, easy-to-use data structures and data analysis tools for Python programming.

**Matplotlib:** is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms.

**Scikit-learn:** (formerly scikits.learn and also known as sklearn) is a free software machine learning library for the Python programming language.

**Keras:** is a neural network library while **TensorFlow** is the open source library for a number of various tasks in machine learning. **TensorFlow** provides both high-level and low-level APIs while **Keras** provides only high-level APIs.

**OpenCV (Open Source Computer Vision)** is a library of programming functions mainly aimed at real-time computer vision. In simple language it is a library used for Image Processing. It is mainly used to **do** all the operations related to Images.

## Our References:

1. <https://medium.com/@smallfishbigsea/faster-r-cnn-explained-864d4fb7e3f8>
2. <https://medium.com/coinmonks/review-r-cnn-object-detection-b476aba290d1>
3. <https://medium.com/@andrewjong/how-to-use-roi-pool-and-roi-align-in-your-neural-networks-pytorch-1-0-b43e3d22d073>
4. <https://medium.com/@kegui/what-is-the-difference-between-nearest-neighbor-bilinear-interpolation-and-cubic-convolution-9589c58e2f17>
5. <https://www.youtube.com/watch?v=v5bFVbQvFRk>
6. <https://medium.com/@purnasaigudikandula/a-beginner-intro-to-convolutional-neural-networks-684c5620c2ce>
7. <https://plotly.com/python/cufflinks/>
8. <https://towardsdatascience.com/understanding-and-coding-a-resnet-in-keras-446d7ff84d33>
9. <https://medium.com/@14prakash/understanding-and-implementing-architectures-of-resnet-and-resnext-for-state-of-the-art-image-classification-cf51669e1624>
10. <https://medium.com/@whatdhack/a-deeper-look-at-how-faster-rcnn-works-84081284e1cd>
11. <https://www.tensorflow.org/resources/models-datasets>
12. <https://towardsdatascience.com/faster-r-cnn-for-object-detection-a-technical-summary-474c5b857b46>
13. [https://keras.io/examples/nlp/bidirectional\\_lstm\\_imdb/](https://keras.io/examples/nlp/bidirectional_lstm_imdb/)
14. <https://link.springer.com/article/10.1007/s10032-019-00335-y>
15. [http://openaccess.thecvf.com/content\\_CVPR\\_2019/html/Wang\\_Region\\_Proposal\\_by\\_Guided\\_Anchoring\\_CVPR\\_2019\\_paper.html](http://openaccess.thecvf.com/content_CVPR_2019/html/Wang_Region_Proposal_by_Guided_Anchoring_CVPR_2019_paper.html)
16. <https://www.geeksforgeeks.org/r-cnn-vs-fast-r-cnn-vs-faster-r-cnn-ml/>
17. <https://patents.google.com/patent/US20180260415A1/en>
18. <https://towardsdatascience.com/faster-r-cnn-for-object-detection-a-technical-summary-474c5b857b46>
19. <https://medium.com/@purnasaigudikandula/a-beginner-intro-to-convolutional-neural-networks-684c5620c2ce>
20. <https://medium.com/coinmonks/review-r-cnn-object-detection-b476aba290d1>
21. <https://medium.com/@prvnrk10/object-detection-rcnn-4d9d7ad55067>
22. <https://towardsdatascience.com/r-cnn-3a9beddfd55a>
23. <https://arxiv.org/abs/1311.2524>
24. <https://www.quora.com/What-is-the-difference-between-CNN-and-R-CNN>