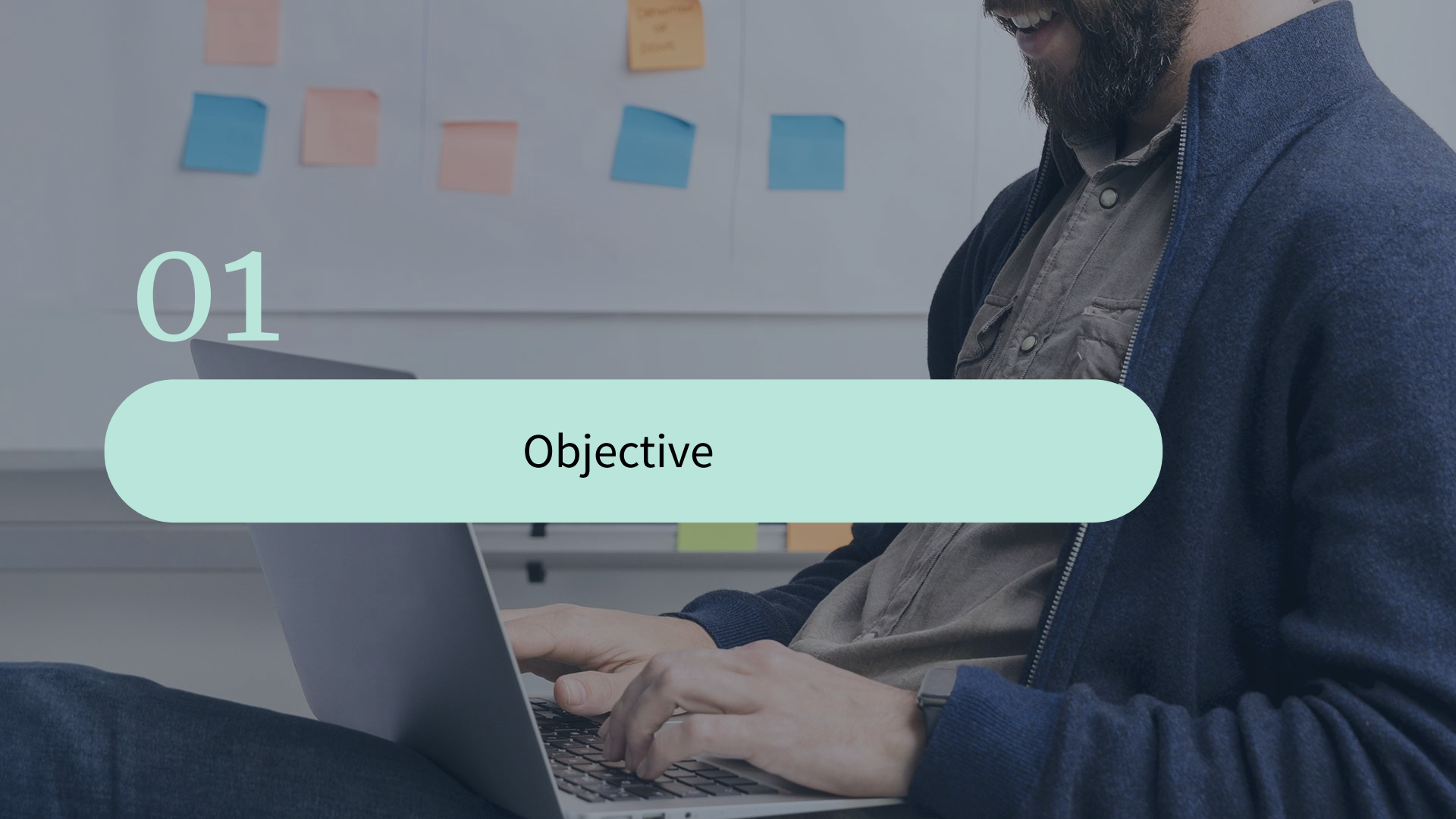
Rich Feature Hierarchies

Object Detection, and Semantic Segmentation

Team reVision | CSE 478 Computer Vision Course Project Spring 2021

Team Members

- Dolton Fernandes
- George Tom
- Naren Akash R J
- Shodasakshari Vidya



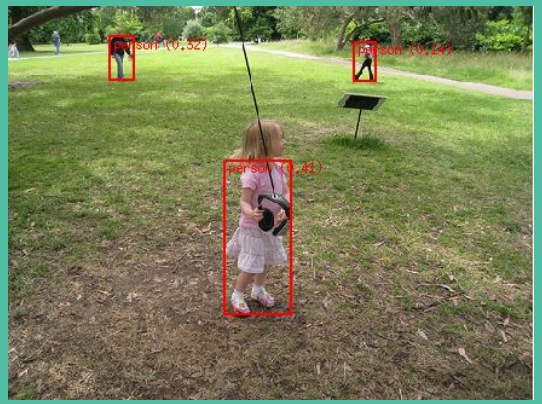
01

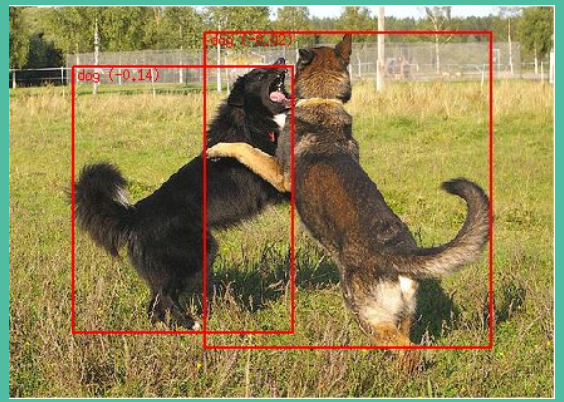
Objective

Objective

The objective of the project is to do object detection on images by combining region proposal with CNN.

Given an image, we should be able to identify image regions and the corresponding objects correctly.





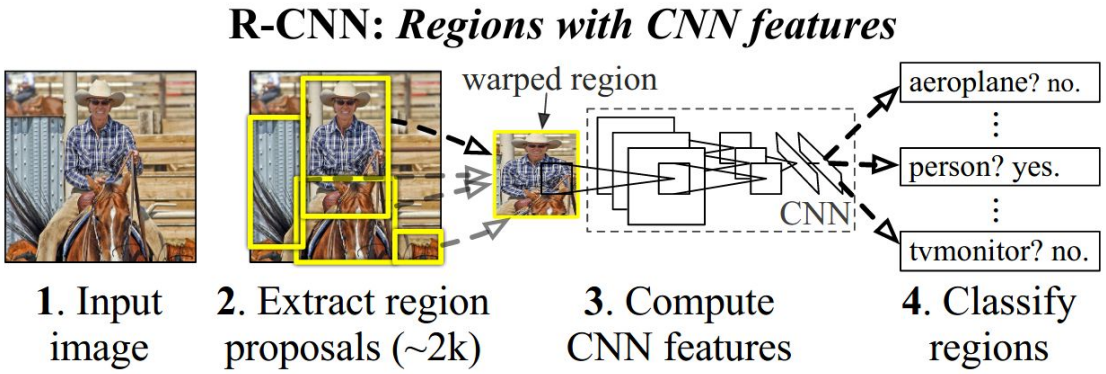
02

Method

Outline of the Method:

The system consists of 3 parts:

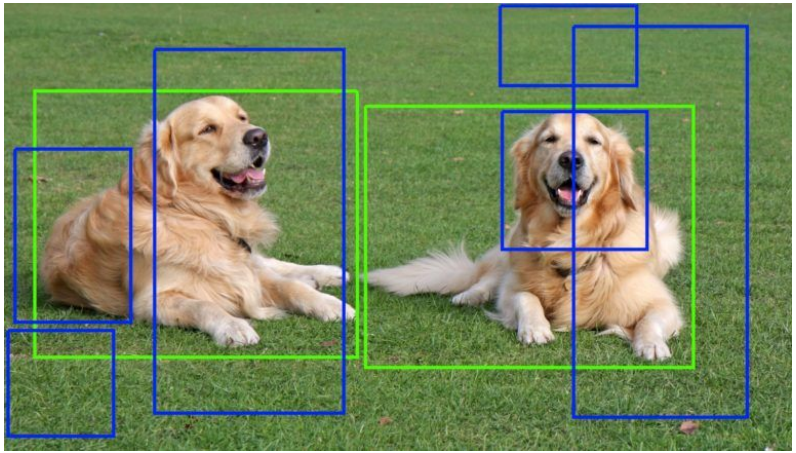
1) Region proposal generation
2) CNN-based feature extraction per region proposal
3) Object Classification



1) Region proposal generation

First of all, what is a region proposal?

- A region proposal is a bounding box candidate that *might* contain an object.



Here the blue and green boxes are the region proposals.

Green proposals are the ones that contain an object.

1) Region proposal generation

We first generate ~2000 proposals per image using selective search.

Selective Search works by over-segmenting an image using a superpixel algorithm.

We could also use other methods like CPMC, exhaustive search, etc.

Selective Search for Region Proposal

The general idea is that a region proposal algorithm should inspect the image and attempt to find regions of an image that likely contain an object

The region proposal algorithm should:

- Be faster and more efficient than sliding windows and image pyramids
- Accurately detect the regions of an image that could contain an object
- Pass these “candidate proposals” to a downstream classifier to actually label the regions, thus completing the object detection framework

Selective Search for Region Proposal

The Selective Search algorithm implemented in OpenCV was first introduced by Uijlings et al. in their 2012 paper, Selective Search for Object Recognition.

Selective Search works by over-segmenting an image using a superpixel algorithm.

Selective Search for Region Proposal

Selective Search merges superpixels in a hierarchical fashion based on five key similarity measures:

1. **Color similarity**: Computing a 25-bin histogram for each channel of an image, concatenating them together, and obtaining a final descriptor that is 25×3=75-d. Color similarity of any two regions is measured by the histogram intersection distance.
2. **Texture similarity**: For texture, Selective Search extracts Gaussian derivatives at 8 orientations per channel (assuming a 3-channel image). These orientations are used to compute a 10-bin histogram per channel, generating a final texture descriptor that is 8x10x=240-d. To compute texture similarity between any two regions, histogram intersection is once again used.

Selective Search for Region Proposal

3. **Size similarity**: The size similarity metric that Selective Search uses prefers that smaller regions be grouped earlier rather than later.

Anyone who has used Hierarchical Agglomerative Clustering (HAC) algorithms before knows that HACs are prone to clusters reaching a critical mass and then combining everything that they touch.

By enforcing smaller regions to merge earlier, we can help prevent a large number of clusters from swallowing up all smaller regions.

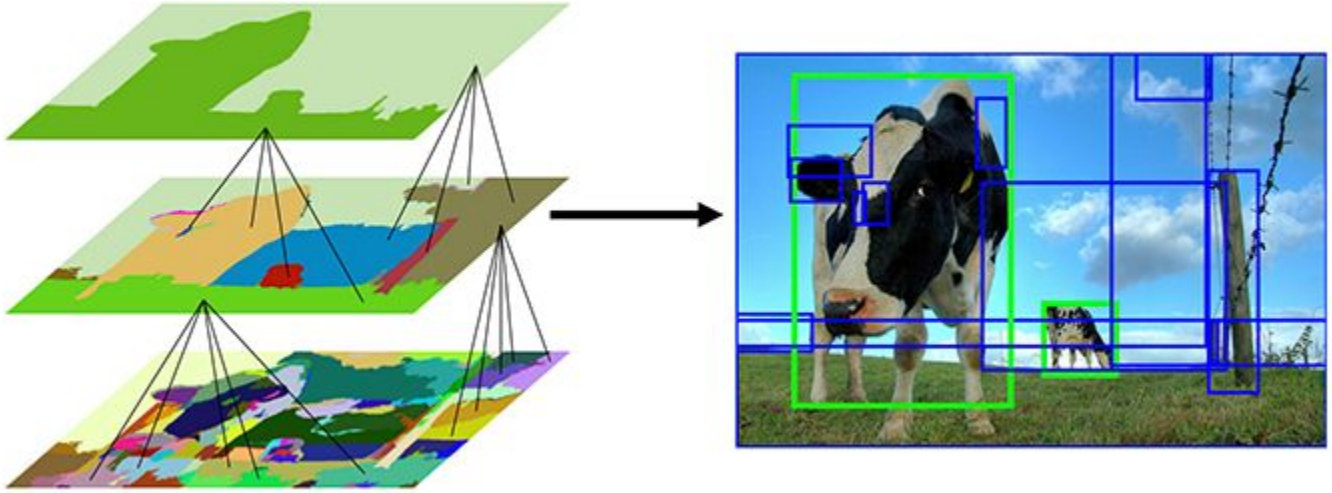
Selective Search for Region Proposal

4. **Shape similarity/compatibility**: The idea behind shape similarity in Selective Search is that they should be compatible with each other. Two regions are considered “compatible” if they “fit” into each other (thereby filling gaps in our regional proposal generation). Furthermore, shapes that do not touch should not be merged.

5. **A final meta-similarity measure**: A final meta-similarity acts as a linear combination of the color similarity, texture similarity, size similarity, and shape similarity/compatibility.

Selective Search for Region Proposal

The results of Selective Search applying these hierarchical similarity measures can be seen in the following figure:



2) CNN Features

Then for each region proposal we use a pretrained Convolutional Neural Network (CNN) to extract features.

We use AlexNet by Krizhevsky et al. as the CNN (takes 227x227 RGB images, converts them into 4096-dimensional vectors).

To make the image in proposal fit the input constraint (227x227 RGB image), we append p=16 pixels to all the four sides of the region proposal and then resize them to 227x227.

So for each proposal, we get a feature representation of it in the form of a 4096 dimensional vector.

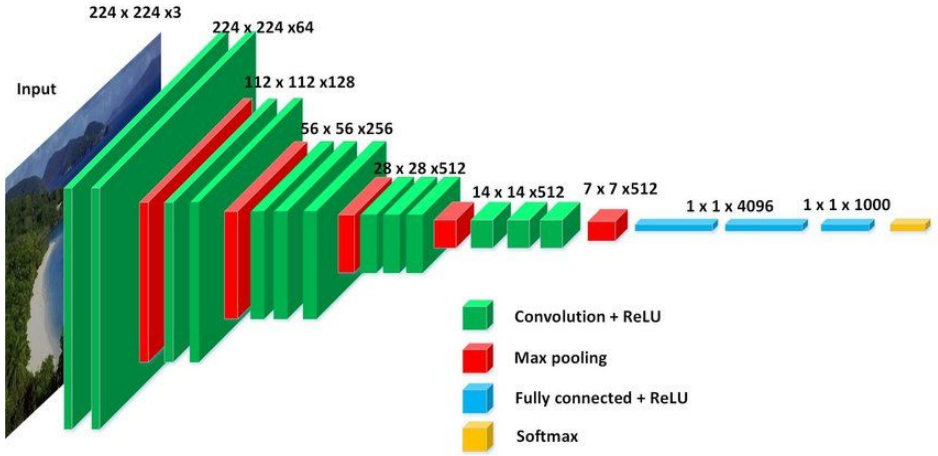
2) CNN Features

Additions to this:

1) We can use a alternate method like center cropping for making the region fit the input criteria of 227x227.

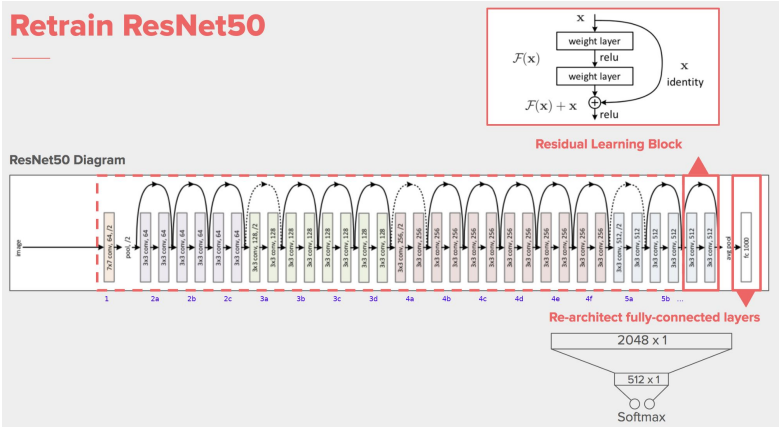
2) We can try alternate feature extractors like VGG16, ResNet50, GoogleNet, Inception-v3, etc.

2) CNN Architecture for Feature Extraction



VGG is a convolutional neural network model for image recognition proposed by the Visual Geometry Group in the University of Oxford, where VGG16 refers to a VGG model with 16 weight layers, and VGG19 refers to a VGG model with 19 weight layers.

2) CNN Architecture for Feature Extraction



ResNet-50 is a convolutional neural network that is 50 layers deep. You can load a pre-trained version of the network trained on more than a million images from the ImageNet database. The pretrained network can classify images into 1000 object categories, such as keyboard, mouse, pencil, and many animals.

3) Classification with SVM

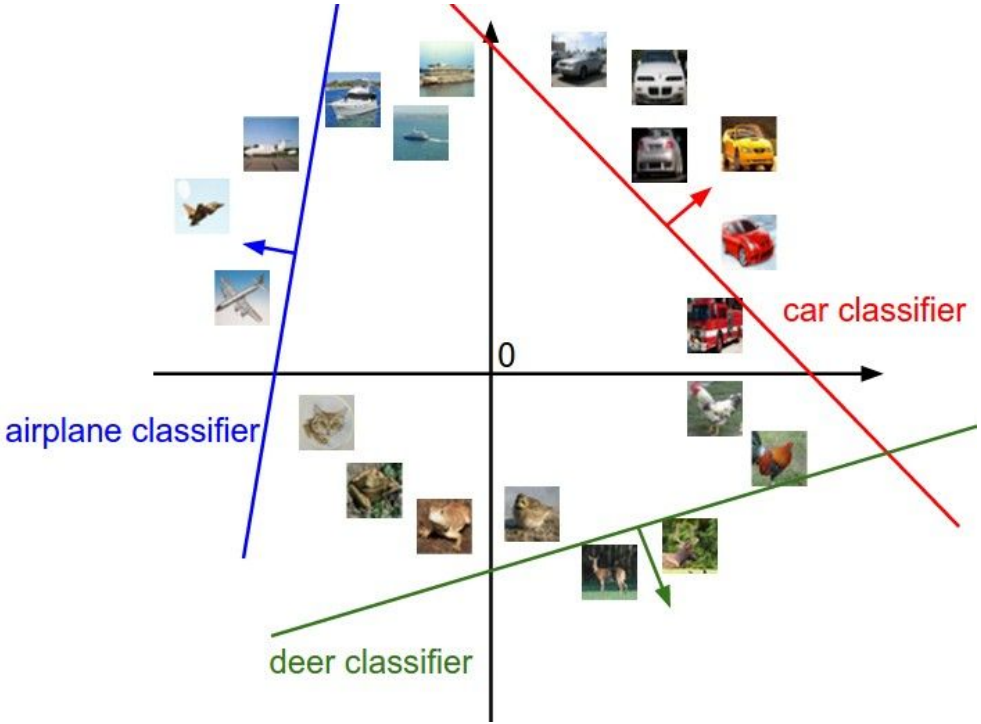
For the classification of each region (i.e. 4096-dimensional vector), we train one SVM per available class.

One SVM predicts the confidence score of the region belonging to the assigned class.

There might be multiple bounding boxes for the same object. To filter this out we use Non maximum suppression (NMS). This method simply rejects regions if they overlap strongly with another region that has higher score.

Instead of completely removing the proposal with high IoU and losing out on close distinct proposals, we can use an alternate method called Soft-NMS to reduce the confidence score of proposals with high IoU.

3) Classification with SVM



Support Vector Machine is a supervised classification algorithm where we draw a line between two different categories to differentiate between them. SVM is also known as the support vector network.

03

Goals Revisited

Project Timeline and Milestones

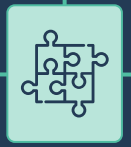
February 20, 2021

Literature Survey, and Related Works

February 28, 2021

Region Proposal Implementation

March 10, 2021



Implementation of Feature Extraction with ConvNets

Project Timeline and Milestones

March 28,2021

Implementation of Object Category Classifier(using SVM), and Training

April 10, 2021

Visualization and Analysis of Results

Final Project Presentation

reVision