Canny edge detection

The Canny edge detector is an edge detection operator that uses a multi-stage algorithm to detect a wide range of edges in images. It is composed of 5 steps:

1. Noise reduction
2. Gradient calculation
3. Non-maximum suppression
4. Double threshold
5. Edge tracking by hysteresis

Affine transformation?

Affine transformation is a geometric transformation that preserves lines and parallelism, but not necessarily Euclidean distances and angles. These include rotation, scaling, and shearing the image. We combine all these transformations into a single matrix called affine transformation matrix. It has 6 degrees of freedom.

Canny edge?

Canny edge detection is an advanced image processing technique used in computer vision to identify the edges of objects in an image. Canny edge detection is a fundamental technique in the field of computer vision that is primarily used for image processing. The technique's overall objective is to identify edges in an image with high accuracy and low false-positive rates. To start with, a Gaussian filter is applied to smooth the image to reduce noise. Once the image is smoothed, the technique computes the gradient magnitude and direction of the image by applying Sobel operators, which are convolutional filters. The operator computes the gradient vector for each pixel in the image, and the gradient magnitude represents the strength of the edges while the direction represents the orientation of the edges.

After computing the gradient magnitude and direction, the technique applies a non-maximum suppression technique to eliminate non-essential edge points. The non-maximum suppression takes advantage of the fact that edges will have a local maximum in the gradient direction, leading to the retention of the edge pixels that have the highest magnitude.

The edge detection process continues by applying double thresholding techniques. The technique involves setting two thresholds - a low threshold and a high threshold. Edges pixels whose magnitude is above the high threshold are considered strong edges, while those between the high and low thresholds are considered weak edges. Weak edges that are not part of a strong edge will be eliminated, while those that are part of strong edges will be retained.

The final step in the Canny edge detection process is edge tracking by hysteresis. This step connects weak edges that are part of strong edges with a continuous line. By doing so, the

Canny edge detection is able to obtain a highly accurate edge map that identifies the true edges in the image with few false positives.

SIFT?

=> SIFT (Scale-Invariant Feature Transform) is a widely used computer vision technique for image feature extraction and matching. SIFT keypoints are local features of an image that are invariant to scale, orientation, and affine distortion. SIFT algorithm works in steps as follows:

1. Scale-space extrema detection: The first step in SIFT is to detect the key points (or keypoints) in an image that are stable across scale and orientation changes. This is done by constructing a scale space representation of the image and then by detecting extrema of the difference-of-Gaussian function.

2. Keypoint localization: Once the keypoints are found, we must localize them accurately. This is done by fitting a 3D quadratic function to the scale-space extrema.

3. Orientation assignment: SIFT computes keypoint orientations based on image gradient directions. A histogram of gradient directions is built around the keypoint and the peak in the histogram is assigned as the orientation of the keypoint.

4. Descriptor computation: After orientation assignment, a feature descriptor is computed for each keypoint. The descriptor is a 128-dimensional vector that captures the local appearance around the keypoint. The vector is computed using a weighted histogram of gradient directions of the image patch around the keypoint.

5. Matching: Once the feature descriptors are computed for keypoints in multiple images, SIFT matches keypoints between the images using a distance metric. The nearest neighbor or best match is selected based on Euclidean distance between feature descriptors.

6. Outlier rejection: To remove outliers in the matching process, keypoints with a large ratio of second-best match distance to the best-match distance are rejected.

Hough transforms?

The Hough transform is a feature extraction technique commonly used in computer vision , image analysis, and digital image processing. The algorithm is used to detect imperfect instances of objects within a certain class of shapes by a voting procedure carried out in a parameter space. The classical Hough transform was first concerned with detecting straight lines in an image, but later extensions have made it possible to detect the positions of arbitrary shapes, most commonly circles or ellipses.
The Hough transform works by representing an image in a parameter space, where an image in Cartesian space usually consists of pixel intensities arranged as a 2D array, the Hough transform represents it in a parameter space that stores information about the spatial location of

straight lines in the image. It takes edge points from an image and maps them to a set of curves in parameter space. Each curve in parameter space corresponds to a line in the original image. By plotting the curves that correspond to detected lines in the same accumulator array, it is possible to detect intersecting curves in the 2D parameter space, which correspond to intersecting straight lines in the original image.

CNN

Output size = (X-F+2P/S)+1

Challenges Faced by Ann

- Images have 2D spatial features, hence the flattened pixel values that are passed into ANN are not useful for classification tasks.
- Not possible for the network to learn features at different places in the image
- It is computationally expensive

CNN Solves this

- By using local receptive fields which are hidden units that are connected to the previous layer. Responsible for reducing parameters and capturing spatial information.
- Weights sharing enables rotation,translation and scale invariance also reduces number of parameters. SIFT is not needed.
- Pooling accumulates info from the previous layer simply aggregates information from the previous layer also responsible for reducing output size which reduces the number of computations.

Image augmentation

is the process of artificially increasing your dataset size. It is helpful when the data provided to you has less images.which indirectly reduces the chances of your model to overfit.

- Neural Network are nothing but trainable parametric nonlinear functions

Overfitting

is a scenario where the model has low bias but high variance. It performs well on the training data while the performance degrades when tested on the testing dataset.

To prevent overfitting one must use the below methods:

- Regularization methods add a penalty term to the objective function of a learning algorithm, which encourages the model to learn a simpler representation of the data.
    - Using L1 and L2 regularizers
    - Dropout layer
- Check the difference in the distribution of images for the respective labels of the train data.
- Acquire or generate more data. Data Augmentation might also help.
- Stop the training model earlier.

Dropout layer

- This layer is used in Cnn as it helps prevent overfitting due to co adaptation.
- Co adaptation refers to the process when behavior of a neuron affects other neurons in the networks which lowers the model performance .
- What it does is that it drops some neurons or nodes in the training process based on the probability provided to it.
- So it creates a new neural network from the parent neural network. It is used only in training not in testing or validation.
- During test and validation the final weights are multiplied by the probabilities fed to them.

Internal Covariance Shift

- in a neural network when data passes from one layer to another layer the inputs that are passed in each layer have different distributions. So if the model is very deep (has a lot of layers) the shift in the input distribution becomes high which is problematic to the neural networks this is known as Internal Covariance shift.

Batch Normalization

It can be fixed by using this layers in the model. It is generally added after the activation function. It zero centers the values and normalizes them and scales and shifts the result. During training it standardizes the intermediate inputs and also rescales and offsets them.

As standardization requires SD and mean when passing a single or few images for testing or validation DL frameworks estimates the mean and SD by using moving average of the input of the layer.

Global Average Pooling

Reduces the increase in the number of params caused by flattening layers.
Takes an average of all the feature maps of the layer.

Transfer Learning

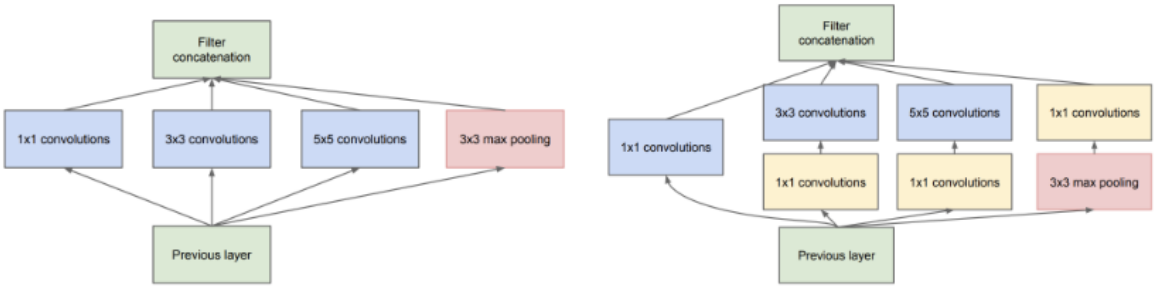
It is a process in which we transfer knowledge of a pretrained network to perform some task and use it knowledge or weights , then fine tune them and use it to perform the task we want it for. Mostly used when the model is trained on a similar dataset to our problem statement.

AUC ROC

The area under the ROC curve (AUC) is a metric that measures the performance of the classifier, which represents the probability that a classifier will rank a randomly chosen positive instance higher than a randomly chosen negative instance. AUC ranges between 0 and 1, where 0 represents a poor classifier and 1 represents a perfect classifier.

The ROC and AUC curve are commonly used in machine learning for evaluating and comparing the performance of binary classification models, especially in cases where the dataset is imbalanced.

ADVANCED CNN



(a) Inception module, naïve version (b) Inception module with dimension reductions

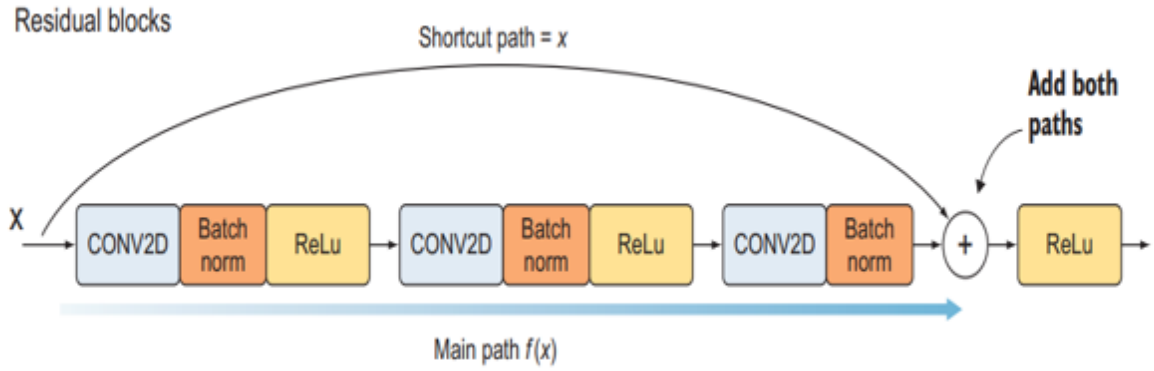
GoogleNet/Inception Model

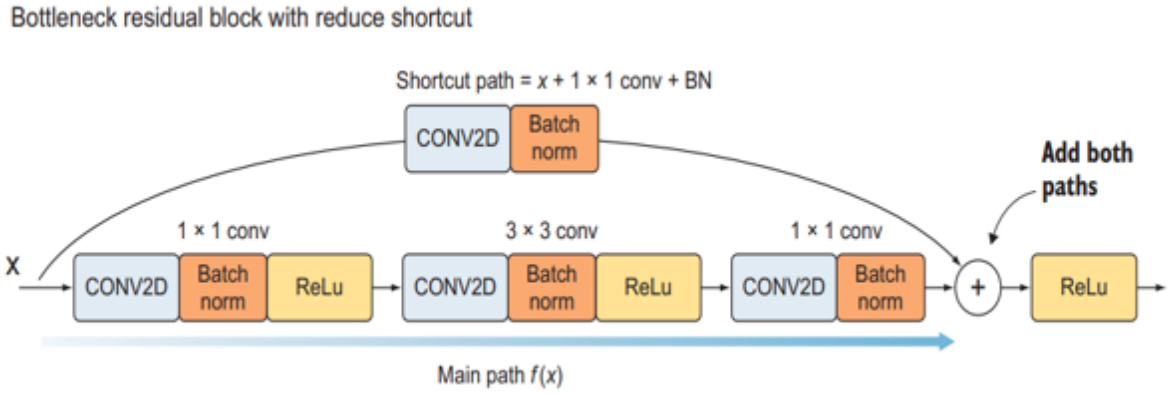
The inception model uses a network of 22 layers deep while reducing the number of parameters by 12 times and achieving accurate performance.

The naive version consisted of 3 convolution layers and one max pooling layer. The details is given below
1X1 convolution
3X3 convolution
5X5 convolution
3X3 Max pooling

The 1X1 convolution layer reduces the operation cost by one tenth of the time. That is why it's called the **reduce layer or bottleneck layer.**
Adding this layer before bigger kernels like 3X3 and 5X5 , would help reduce depth and in return will reduce the number of operations.
So by running experiments that adding the 1X1 Convolution layers we can reduce the representation size without hurting the performance.
Hence we apply 1X1 before 3X3 and 5x5 to reduce computational cost. Apply it after 3x3 Max pooling as pool layers don't reduce the depth for their inputs but reduces spatial features.

Uses Global Average pooling instead of flatten.
It also uses auxiliary classifiers for prediction and gradient descent at intermediate steps of the network to solve the vanishing gradient problem.





ResNet

It is a residual module which also uses skip connection and residual blocks. The network uses a lot of batch normalization in the model. The deeper the networks the better capacity of the model to extract features from the input or images. As deeper networks allow the model to learn not only simple but complex features as well.
Problem of deeper networks:
Overfitting
Which can be fixed by using dropout and batch normalization.
and **Vanishing and exploding Gradients**

During backpropagation chained multiplication og gradient from the layers becomes very small by the time it reaches to the initial layers , for activation with diminishing gradient like tanh and sigmoid

Skip Connection

It is also possible that during backpropagation the gradient while traveling from layers to initial layers grow exponentially or become very large thus exploding , for activations like relu.
To solve the vanishing gradient problem for resent microsoft decided to create a shortcut that allows the gradient to directly back propagate to previous layers.These are called **skip connection.** Skip connections also allow the model to learn an **identity function** which ensures that the layer will perform at least as well as the previous layer.

Residual Block

Unlike traditionally just stacking the convolutions just one after the other they also add the original input to the output of the stacked convolutional blocks. The shortcut arrow/skip connection is added at the end of the convolution layer. The reason is that we add both paths before we apply the ReLU activation function of this layer. This combination of skip connect and convolution layers is called **residual blocks**.

It has two branches:

1.Shortcut path: Connects the input to an addition of the second branch

2.Main path: A series of convolutions and activations. The main path consists of three convolutional layers with ReLU activations. We also add batch normalization to each convolutional layer to reduce overfitting and speed up training. The main path architecture looks like this: [CONV -> BN -> ReLU] × 3.

Summary

Residual blocks contain two paths

1.The shortcut path and the main path.

2.The main path consists of three convolutional layers, and we add a batch normalization layer to them:
1.1 × 1 convolutional layer

2.3 × 3 convolutional layer

3.1 × 1 convolutional layer

3.There are two ways to implement the shortcut path:
1.Regular shortcut: Add the input dimensions to the main path.

2.Reduce shortcut: **Add a convolutional layer in the shortcut path before merging with the main path.**

****NOTE**** does not use pooling either. It uses strides of 2 for downsampling width and height

Mobile Net

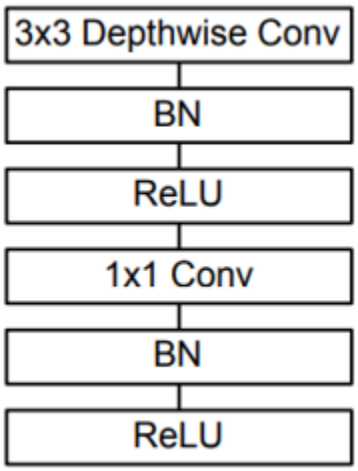
It uses **depth wise separable Convolutions** along with **reduce layer** (1x1 convolution layer) to build a lightweight deep neural network.
Depth Wise separable convolution is the convolution along the spatial channel of the image. Difference between normal convolution and depthwise is that depthwise performs convolution along each spatial channel of the input where normal convolution does it for all of the channels at once.
After depthwise we apply point wise convolution layers. It is responsible to expand the layer not to reduce it.

Input-224x224x3

Using a normal 64x3x3 kernel we get parameters as 1792
3x3x3x64+64=1792

DSC +PWC
3x3x1x3+3=30
1x1x3x64+64=256
total=286

Architecture



****NOTE**** does not use pooling either. It uses strides of 2 for downsampling width and height

Two hyperparameters of mobilenet

1.Width Multiplier:used to make the mobile net model even smaller and faster if needed.

- The role of the width multiplier α is to thin a network uniformly at each layer.
- For a given layer and width multiplier α, the number of input channels M becomes αM and the number of output channels N becomes αN. The value of α is typically between [0,1]

2.Resolution Multiplier: A resolution multiplier ρ is used on the input image and the internal representation of every layer is subsequently reduced by the same multiplier. Value of ρ is typically between [0,1].

What is Object Detection?

Object detection is a computer technology related to computer vision and image processing that deals with detecting instances of semantic objects of a certain class in digital images and videos.

It is a computer vision technique for locating instances of objects in images or videos. Object detection algorithms typically leverage machine learning or deep learning to produce meaningful results.

How do you localize an object in an image? - Bounding Box

Bounding Box Regression

Syntax:

cv2.rectangle(image, start_point, end_point, color, thickness)

Where, start_point = (xmin, ymin) and end_point = (xmax, ymax)

So, we perform a regression task to predict 4 values of 2 pairs of coordinates: xmin, ymin, xmax, ymax.

We feed the images into CNN after normalizing the coordinates.

Classification with Bounding Box Regression

Class predictor maybe a sigmoid or a softmax depending on the number of class of objects that we are predicting

How to assess the prediction of Bounding box?

Intersection over Union(IOU)

This measure evaluates the overlap between two bounding boxes: the ground truth bounding box (Bground truth) and the predicted bounding box (Bpredicted). By applying the IoU, we can tell whether a detection is valid (True Positive) or not (False Positive)

Intersection over Union:

The intersection over the union value ranges from 0 (no overlap at all) to 1 (the two bounding boxes overlap each other 100%). The higher the overlap between the two bounding boxes the better(IoU value).

Confusion Matrix for IoU:

Note: True Negative is not applicable to object detection. True Negative means it is correctly detecting the background of an object as background. Which is equivalent to not detecting anything.

Multi Class Object Detection

If you have multiple objects of a similar class in an image it will only detect one of them which it is most confident about. We assess the issue by Regions of Interest (ROI).

Region based CNN Models

RCNN (2013)

It is the first region based architecture in the region proposal based object detection family.

It was one of the first large, successful applications of convolutional neural networks to the problem of object detection and localization

RCNN Components:

- Region Proposal: Extract Regions of Interest (ROI)
- Feature Extraction Module
- Classification Module
- Localization Refinement Module (optional component)

RCNN – Region Proposal

- The regions proposed in this step have a very high probability of containing an object
- An algorithm called selective search scans the input image to find regions that contain blobs and proposes RoIs (2000 RoIs in fast mode)which have to be processed by the next components of the RCNN.
- The proposed RoIs are of different sizes
- But since a CNN is fed an image of fixed size, we warp the RoIs to a fixed size.

Note: Since the selective search algorithm proposes region proposal bounding boxes, bounding box regression becomes an optional component.

RCNN – Feature Extraction

- We extract features with the help of the pretrained CNN model

- As the proposed regions vary in size and are typically warped to fixed size, an imagenet pretrained CNN's has a poor performance.

- So we finetuned the pre trained network on the warped region proposals using the softmax classification output layer.

- VGG16 was used as a pretrained network.

RCNN – Classification Module & Bounding box Refinement

We use a linear SVM model to perform classification on the features extracted using the pretrained cnn.

Linear SVM performs a one vs rest classification to classify an object into a particular class.

Since Selective Search takes care of the localization using its own proposed regions bounding box, we do not perform the bounding box regression step.

To improve the localization process we use bounding box regression to get refined coordinates of the boxes.

Multiple Bbox single object: NMS

Non max suppression is a technique used mainly in object detection that aims at selecting the best bounding box out of a set of overlapping boxes

We sort the bounding boxes in descending order of class confidence score.

We remove the boxes which have a confidence score lesser than the threshold confidence score

Loop over the remaining boxes:

1. Start with the box which has the highest confidence

2. Discard other boxes if they have an IoU > predefined IoU thresh with the highest confidence box

3. Repeat the steps until all the boxes are either taken as output prediction or discarded

Disadvantages of RCNN:

1. **Object Detection is very slow**

For each image selective search proposes 2000 RoIs to be examined by the entire pipeline. These 2000 RoIs are then passed on to the CNN 2000 times for a single image i.e. 2000 forward propagations for a single image. This process makes it computationally very expensive. During inference object detection in a single image takes 50 seconds (47.5 seconds for CNN + time taken for selective search)

2. **Multiple Training stages (3 stages):**

Finetuning of CNN feature extraction

Training of object classes using SVMs

Training of Bbox regressors for bounding box refinement

3. **Two stage pipeline:**

First stage is for Region Proposals

Second stage is for Object Detection on Region Proposals

4. **Training is expensive in terms of both time and space:**

2000 RoIs per image need to fine tuned

2000 RoIs per image need to be stored on the disk

Fast RCNN

Bounding box coordinate format: (x_c, y_c, w, h)

x_c – x center, y_c – y center, w – width, h - height

1. Pass the input image to the feature extractor just once
2. Pass the input image to selective search algorithm to get RoIs/Region Proposals
3. Feature maps are produced by the pooling layer of the pretrained network
4. RoIs from step 2 are projected onto the feature maps
5. RoI pooling is performed on the projected RoIs
6. RoI pooling output is passed to the fully connected layers
7. Fully connected layer is connected to the multi-head output:
    - Classifier
    - Bbox Regressor

RoI Projection:

- The final max pooling layer produces a feature map of size 7x7x512
- An image of size 224x224 gets downsampled to 7x7, therefore we can calculate its subsampling ratio by 224/7 = 32
- Any object that is present in the image of size 224x224 can be projected to the feature map of size 7x7 if we divide the height and width of that object by 32.

ROI Pooling:

- The projected RoIs on the feature maps are then pooled and passed on to the fully connected layers
- But since the projected RoIs will vary in size we need to do something about it
- We perform a special type of pooling known as ROI pooling.
- ROI pooling takes the max value from each grid
- We use a 7x7 grid to perform ROI pooling on a feature map which will produce a vector of size 49x1
    - 1x1 grid ROI pooling will produce 1x1 vector

- 2x2 grid will produce a 4x1 vector
- 4x4 grid will produce a 16x1 vector

Classification and Bounding box Regression Module:

· Classification: Softmax classifier is used for classification prediction

· Bounding box Regression: Offsets between the selective search region proposals and ground truth bounding boxes are predicted for each class (this method is known as Relative Bounding Box Regression)

1. tx = x_center_gt – x_center_ss
2. ty = y_center_gt – y_center_ss
3. tw = w_gt – w_ss
4. th = h_gt – h_ss

Advantages and Disadvantages of Fast RCNN:

Advantages:

1. Requires only one propagation for each image unlike 2000 in RCNN
2. Three stage training from RCNN reduced to one stage training
3. Faster in inference time than RCNN: 2 seconds (0.32 seconds for CNN + time required for selective search)

Disadvantages:

1. It is still a two stage pipeline
    - Selective Search Region Proposal
    - CNN

2. Selective search still remains to be a bottleneck as it is very slow to generate region proposals. 2 seconds of inference time is too much for real time object detection in videos.

Loss Function:

Fast R-CNN

- Multi-task loss

$$L(p, u, t^u, v) = L_{\text{cls}}(p, u) + \lambda[u \geq 1]L_{\text{loc}}(t^u, v)$$

Where $$L_{\text{loc}}(t^u, v) = \sum_{i \in \{x,y,w,h\}} \text{smooth}_{L_1}(t_i^u - v_i)$$

$$\text{smooth}_{L_1}(x) = \begin{cases} 0.5x^2 & \text{if } |x| < 1 \\ |x| - 0.5 & \text{otherwise} \end{cases}$$

Faster RCNN

· We now know that the major bottleneck with region based CNNs till now is the traditional and slow region proposal method (selective search)

· We need to replace the selective search method with some other region proposal method based on the following criteria

- < 2000 region proposals
- As fast as SS or better
- As accurate as SS or better
- Should be able to propose overlapping ROIs with different aspect ratios and scales

Faster RCNN: Region Proposal Aspect Ratios

- To detect most kinds of objects you typically use one of the 3 aspect ratios given below
    - Vertical rectangle
    - Square
    - Horizontal rectangle
- Most of the object will be obstructing the one behind and hence you need overlapping ROIs
- Objects maybe of different size therefore you need ROIs of different scale

Faster RCNN: Components

To address the constraint of selective search for region proposals we have the following addition in Faster RCNN

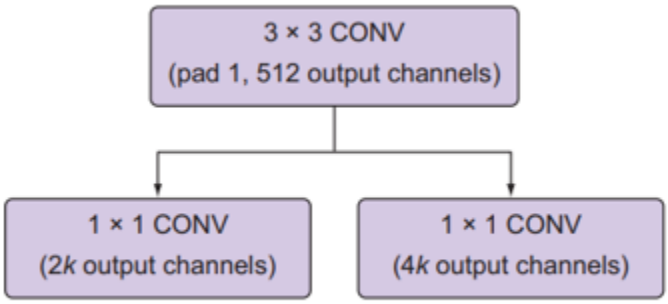
1. Region Proposal Network (RPN): Selective Search is replaced by a ConvNet that proposes RoIs from the last feature maps of the feature extractor that are considered for investigating objects of interest. RPN has two outputs
    1. Objectness Score: whether there is an object present (1) or absent(0)
    2. Box location
2. Fast RCNN: same components
    1. Base network for feature extraction
    2. RoI pooling layer
    3. Output layer that contains two fully connected layers: softmax classifier and bounding box regression

Region Proposal Network (RPN)

RPN identifies RoIs based on the last feature map of the pretrained CNN. Faster RCNN uses RPN to perform the region proposals instead of using selective search like its earlier versions

RPN Architecture

The architecture of RPN is composed of two layers



- A 3x3 conv layer with 512 filters
- Two parallel 1x1 conv layers

    1. Classification layer: Predicts whether the is an object present in the proposed region (or just random background
    2. Bounding box regression layer: bounding box of proposed RoIs

Note: Remember that RPN is not doing object detection but it is only finding objects of interest that the later components can both localize and classify

How are the region proposed bounding boxes predicted?

Bounding box can be defined as (x,y,w,h) where x and y are the centers of the bounding box and w and h are the widths and heights respectively

We perform relative bounding box regression i.e. we create reference boxes called as anchor boxes in the image and make the regression layer predict the offsets from these boxes called deltas (Δx, Δy, Δw, Δh) to adjust the anchor boxes to better fit the object to get final proposals.

Anchor Boxes

RPN generates k RoIs for each location in the feature map using a sliding window approach. These regions are represented using anchor boxes.

The anchors are centered in the middle of their corresponding sliding window and are of different aspect ratios and scales to cover a wide variety of different sized objects

They are fixed bounding boxes that are placed throughout the image to be used for reference when first predicting the region proposals for objects of interest

In the Faster RCNN paper, anchor boxes with 3 different aspect ratios and 3 different scales were proposed.

Training the RPN:

Need to ensure there are less than 2000 RoIS

RPN is used to classify whether an object is present or not and propose the bounding box for an object

How does one prepare the data for whether there is an object present or not inside the anchor box?

For each anchor the overlap value is computed which indicates how much these anchor boxes overlap with the ground truth boxes (or how useful these anchor boxes are going to be to help us find the region proposals)

Only the following anchor inputs are taken forward for the objectness classification

1. Anchor boxes having high IoU with ground truth boxes IoU > 0.7 (positive anchor boxes)

2. Anchor boxes having low IoU with ground truth boxes IoU < 0.3 (negative anchor boxes)

3. Rest of the anchor boxes are ignored 0.3< IoU <0.7

During the training process the positive and negative anchors are passed as input to two fully connected layers corresponding to the classification of anchors as containing an object or no object, and to the regression of location parameters (four coordinates)

Corresponding to the k number of anchors from a location, the RPN network outputs 2k scores and 4k coordinates. Thus, for example, if the number of anchors per sliding window (k) is 9, then the RPN outputs 18 objectness scores and 36 location coordinates

Training the Fast RCNN component of Faster RCNN:

The final fully connected layers of the network take two inputs:

- Feature maps from the feature extractor
- RoIs coming from the RPN

The RoIs are projected on to the feature maps in the same way like Fast RCNN

Softmax Classifier and Bounding Box regressor are used to predict the object class and bounding box offsets respectively for the detected object

RPN: Loss Function:

$$\mathscr{L}(p_i, t_i) = \frac{1}{N_{cls}} \sum_i \mathscr{L}_{cls}(p_i, p_i^*) + \frac{\lambda}{N_{reg}} \sum_i p_i^* \mathscr{L}_{reg}(t_i, t_i^*)$$

$$\begin{aligned} p_i^* &= 1 \quad \text{if anchor box contains ground truth object} \\ &= 0 \quad \text{otherwise} \\ p_i &= \text{predicted probability of anchor box containing an object} \\ N_{cls} &= \text{batch-size} \\ N_{reg} &= \text{batch-size} \times k \\ k &= \text{anchor boxes} \end{aligned}$$

Faster RCNN Training in a Nutshell:

- Finetune RPN using a pretrained network
- Finetune Fast RCNN using a pretrained network on the proposed images from step 1
- Finetune RPN using the pretrained network used in step 2
- Now finetune the Fast RCNN using the pretrained network used in step 3 on the images proposed in step 3

Review of RCNN Family

1. mAP on Pascal VOC 2007
    1. RCNN: 66%
    2. Fast RCNN: 66.9%
    3. Faster RCNN: 66.9%

2. Limitations
    1. RCNN: 3 stage training, high computation time and memory requirements
    2. Fast RCNN: Selective search is slow and hence computation time is still high
    3. Faster RCNN: It is still a 2 stage network

3. Test Time Per Image
    1. RCNN: 50 seconds
    2. Fast RCNN: 2 seconds
    3. Faster RCNN: 0.2 seconds

4. Speedup from RCNN
    1. Fast RCNN: 25x
    2. Faster RCNN: 250x

Difference Between Fast and Faster RCNN

- Region Proposal Network (RPN): Faster R-CNN uses a Region Proposal Network (RPN) to generate region proposals, while Fast R-CNN uses the Selective Search algorithm. The RPN in Faster R-CNN is faster and more accurate than the Selective Search algorithm used in Fast R-CNN.
- Speed: Faster R-CNN is faster than Fast R-CNN. This is because the RPN in Faster R-CNN shares convolutional layers with the detection network, resulting in faster computation.
- Training Time: Faster R-CNN takes longer to train than Fast R-CNN. This is because of the additional training required for the RPN.

- Accuracy: Faster R-CNN is more accurate than Fast R-CNN. This is because of the improved RPN and the fact that the RPN shares convolutional layers with the detection network, allowing it to learn more discriminative features.
- Architecture: Faster R-CNN is an extension of Fast R-CNN. It adds the RPN to the Fast R-CNN architecture, resulting in a more efficient and accurate model.

Difference between RCNN, Fast RCNN and Faster RCNN

- R-CNN (Region-based Convolutional Neural Networks): R-CNN was the first deep learning-based object detection algorithm. It operates in four stages: region proposal, feature extraction, classification, and bounding box regression. First, it proposes regions of interest (ROIs) using a selective search algorithm. Then, it extracts features from each proposed region using a convolutional neural network (CNN). Next, it classifies each region into object or non-object using a support vector machine (SVM). Finally, it refines the bounding box coordinates of each object using linear regression.

- Fast R-CNN: Fast R-CNN builds on top of R-CNN to make it faster and more accurate. The main difference is that instead of extracting features for each region proposal separately, it extracts features from the entire input image using a CNN. This allows it to share computation across proposals and avoid redundant feature extraction. Another difference is that instead of using an SVM for classification, it uses a softmax classifier that can handle multiple object classes.

- Faster R-CNN: Faster R-CNN builds on top of Fast R-CNN to make it even faster by adding a region proposal network (RPN). The RPN is a small CNN that predicts objectness scores and bounding box offsets for a set of anchor boxes (fixed-size boxes placed at different locations and scales). The predicted boxes with high objectness scores are used as proposals, and then the features of these proposals are extracted and classified using a Fast R-CNN-like network. By combining region proposal and object detection into a single network,

mAP: mean Average Precision

Mean Average Precision (mAP) is a performance metric popularly used for object detection models. It is the most popular metric that is used by benchmark challenges for object detection.

Precision measures the proportion of predicted positives that are actually correct. If you are wondering how to calculate precision, it is simply the True Positives out of total detections. Mathematically, it's defined as follows.

P = TP/(TP + FP)

= TP / Total True Predicted

The value ranges from 0 to 1.

Recall measures the proportion of actual positives that were predicted correctly. It is the True Positives out of all Ground Truths. Mathematically, it is defined as follows.

R = TP / (TP + FN)

= TP / Total Ground Truths

Similar to Precision, the value of Recall also ranges from 0 to 1.

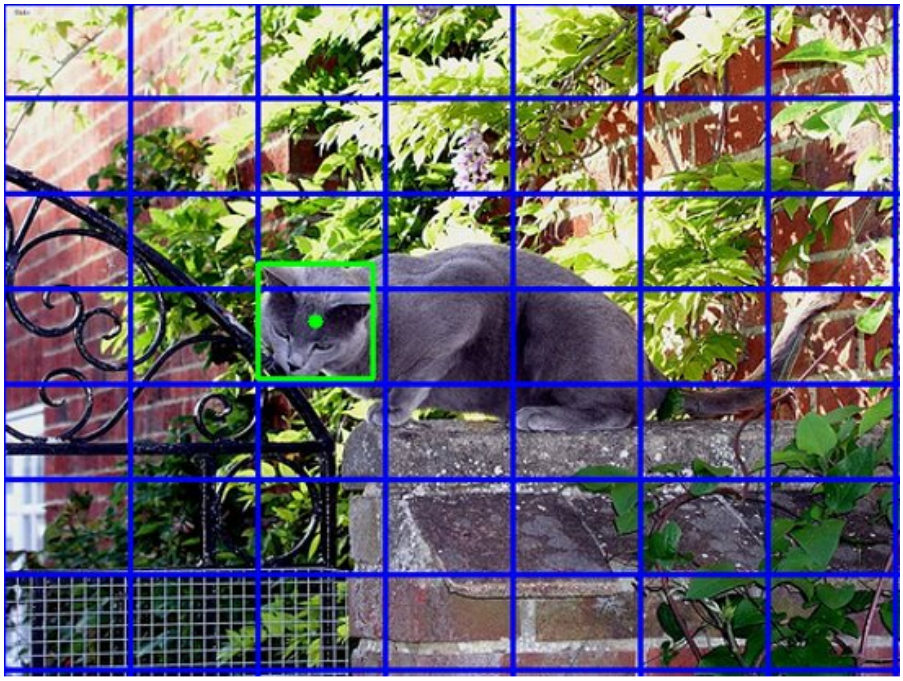
Average Precision (AP) is actually not the average of Precision (P). The term AP has evolved with time. For simplicity, we can say that it is the area under the precision-recall curve.

YOLOV1

Approach:

•The network only looks at the image once to detect multiple objects. Thus, it is called YOLO, You Only Look Once.
•Unlike the two stage detectors like the RCNN family YOLO is a one stage detector and hence the name YOLO

•The input image is divided into an S×S grid (S=7). If the center of an object falls into a grid cell, that grid cell is responsible for detecting that object.



1. Each grid cell predicts B bounding boxes (B=2 in the paper) and confidence scores for those boxes.

2. These confidence scores reflect how confident the model is that the box contains an object. i.e. the probability of prediction of the box

3. Each bounding box consists of 5 predictions: x, y, w, h, and confidence.
4. The (x, y) coordinates represent the center of the box relative to the bounds of the grid cell.

5. The width w and height h are predicted relative to the whole image.

6. The confidence represents the Intersection Over Union (IOU) between the predicted box and any ground truth box.

Out of the B bounding boxes for each grid cell only one is selected based on the highest confidence.

Each grid cell also predicts C conditional class probabilities, where C is the number of classes. YOLOv1 was originally trained on PascalVOC2007 dataset which had 20 classes so in that case C=20 (S is the grid size, B is the no. of bounding boxes, C is the no. of classes)

S = 7, B = 2, C = 20
S x S x (B x 5 + C) = 7x7x(2x5+20) = 7x7x30 is the output shape of the network

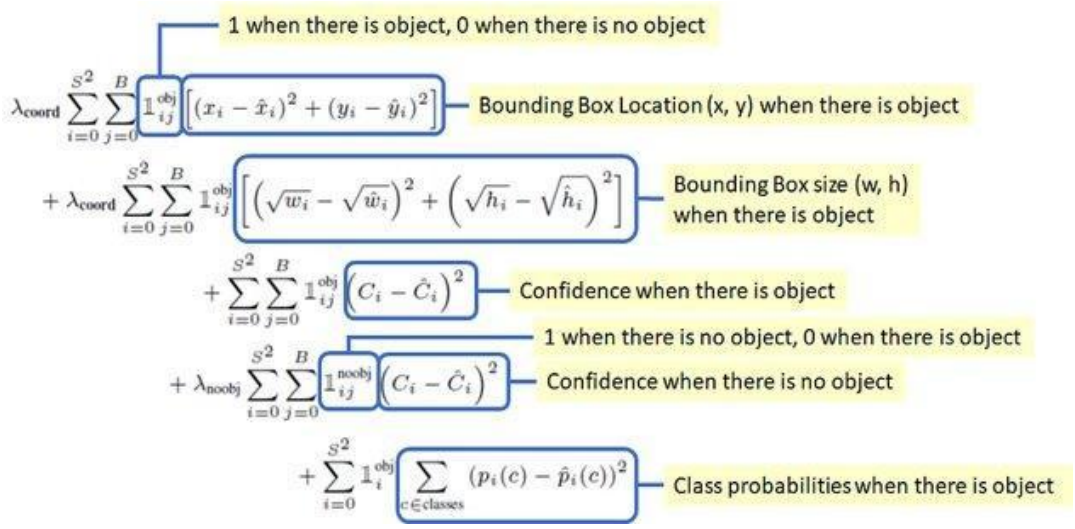
The ground truth data for object detection consists of bounding box coordinates and class predictions and should be structured in the shape of the above output.

Architecture:

- The model consists of 24 convolutional layers followed by 2 fully connected layers. Alternating 1×1 convolutional layers reduce the features space from preceding layers.

- Input Size: 448x448x3, Output size: 7x7x30 (PascalVOC2007)

Loss of YOLO:



Surprisingly YOLO's loss function is the most odd one to have ever been defined as it comprises of squared errors for both box coordinates and class predictions.

1. The bounding box x and y coordinates is parametrized to be offsets of a particular grid cell location so they are also bounded between 0 and 1. And the sum of square error (SSE) is estimated only when there is object.
2.
3. The bounding box width and height are normalized by the image width and height so that they fall between 0 and 1. SSE is estimated only when there is object. Since small deviations in large boxes matter less than in small boxes. square root of the bounding box width w and height h instead of the width and height directly to partially address this problem.

4. In every image many grid cells do not contain any object. This pushes the "confidence" scores of those cells towards zero, often overpowering the gradient from cells that do

contain objects, and makes the model unstable. Thus, the loss from confidence predictions for boxes that don't contain objects, is decreased, i.e. $\lambda_{noobj}=0.5$.

5. SSE of class probabilities when there is objects.

6. Due to the same reason mentioned in 3 and 4, $\lambda_{coord} = 5$ to increase the loss from bounding box coordinate predictions.