École Internationale des Sciences du
Traitement de l'Information

Deep Learning and Convolutional Neural Network

Thibault Le Quéré

October 25, 2019

Contents

1 Convolutional neural network

1.1 Introduction

Convolutional neural network is a machine learning method for image recognition and classification. This method takes as input an 3D array representing the image. The output can be a class or a picture with a delimiter, box to show the interesting point in the input image.

1.2 Why cnn

In the past for image classification, we used to create features from images and feed those feature into classification algorithm. For example in CPI2 I had to detect license plate. For that I used different feature, like shape color and mix it with a letter detector.< CNN work better because they extract feature automatically and it's way more efficient. Image classification have a lot of application field. Detection of cancerous tumor, self driving cars, detect if vegetables are still consumable, bot for video games etc...

1.3 How does it works

Neural network works like the human brain. We feed the neural network with label images to allow it to learn what pictures can contain. For example cat and dogs. The bigger the dataset is the better. As training a CNN require a lot of images, sometimes you have to make your dataset bigger with data augmentation. Processing all those data can take a lot of computation time. If you don't have a good computer and or time you can use transfer learning to have easy pre train CNN. Before talking about those two subject let's go deep in neural network structure!

1.4 Kind of layer

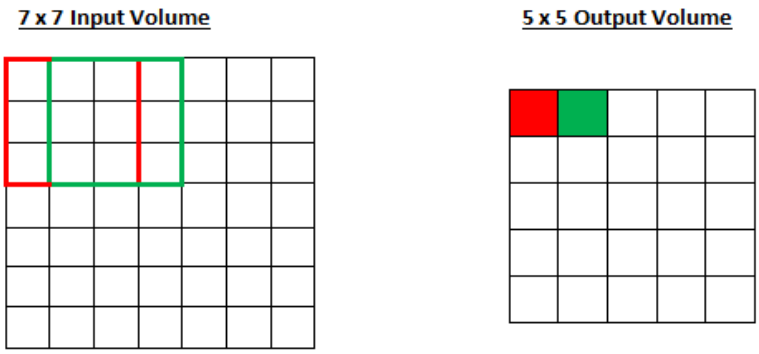
A convolutional neural network is structured by a superposition of layers. There are different kind of layers :

1. Convolutionnal layer
2. ReLu layer
3. pooling layer
4. fully connected layer
5. Dropout layer
6. Loss layer

1.4.1 Convolutional layer

This layer applies a convolutional matrix to the image, it's a filter. There are a lot of filters with they own specificity. With those filters, you can add a Gaussian blur, detect curve or line for example. A filter is an N * N matrix. To apply a filter each value in the red box (left) is multiplied by the value of the

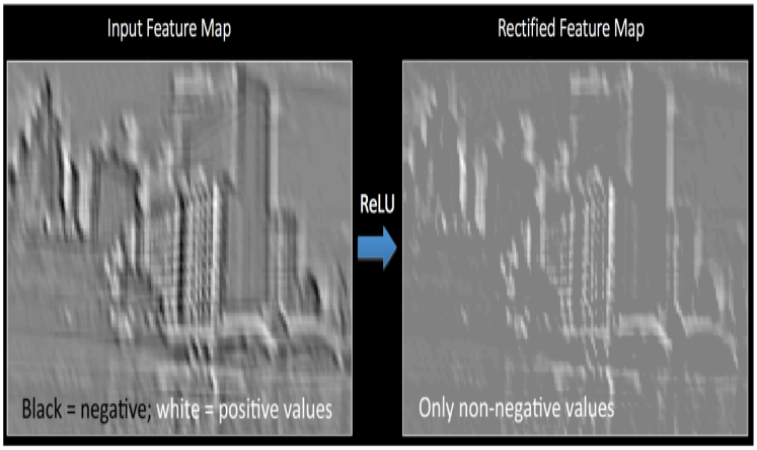
pixel associate, then you add all the value obtain and it gives you the value of the red little square (right). filter and different part of your input matrix (Here you multiply the filter by the red and green part of the matrix). So the output is smaller than the input. An M * M input will give a (M-(N-1)) * (M-(N-1)) output. To prevent the feature map (the image given as output of this layer) size to shrink a layer of zero-value pixels is added to surround the input with zeros.



Convolution layer

1.4.2 ReLu layer

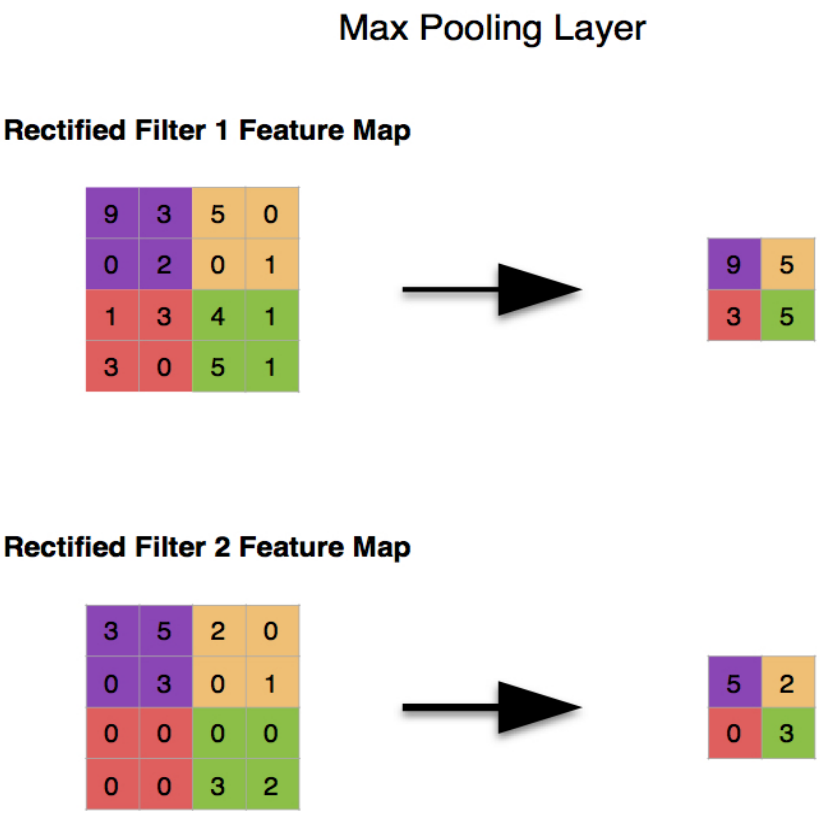
Most parts of the time ReLu layer are used after convolutional layer. ReLu layer is just put at zero every negative values of the image. The purpose of ReLU is to introduce non-linearity, most of the real-world data we would want our CNN is non-linear.



Effect of ReLu layer

1.4.3 Pooling layer

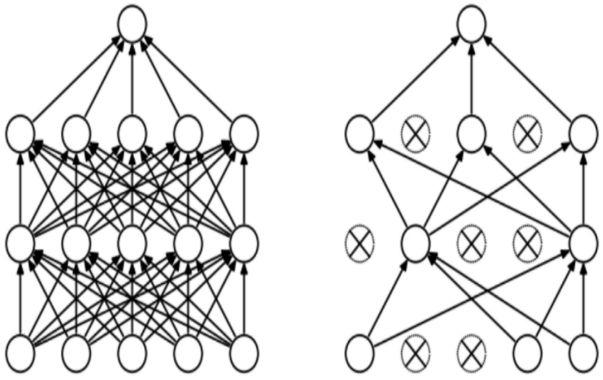
The pooling layer objective is to reduce the size of the image while keeping important information. The max pooling layer only keeps the most significant pixel of each subsection in the image. Soft max is used to predict one class between a group of class (where being in one class forbid to be in another one like cat, dog, horse and car). The sigmoid one is to predict K independent value between [0,1].



Max Pooling Layer

1.4.4 Drop out layer

To avoid overfitting, a dropout layer randomly sets the layer's input elements to zero with a given probability.



(a) Standard Neural Net (b) After applying dropout.

1.4.5 Fully connected layer

The fully-connected layer is where the final "decision" is made. On this layer, the CNN returns the the probability that the object in the photo is of a certain type. The name fully connected is due to the fact that every neuron of the previous layer is connected to every output possible. It's important to know that

we apply an activation function on a fully connected layer to set the neurons value between 0 and 1. For example we often use the sigmoid function when our neural network classifies stuff.

1.5 Loss layer

The "loss layer" specifies how training penalizes the deviation between the predicted (output) and true labels and is normally the final layer of a neural network. Various loss functions appropriate for different tasks may be used.

Softmax loss is used for predicting a single class of K mutually exclusive classes.[nb 3] Sigmoid cross-entropy loss is used for predicting K independent probability values in [0,1]. Euclidean loss is used for regressing to real-valued labels $(-\infty, \infty)(-\infty, \infty)$.

1.6 Back propagation

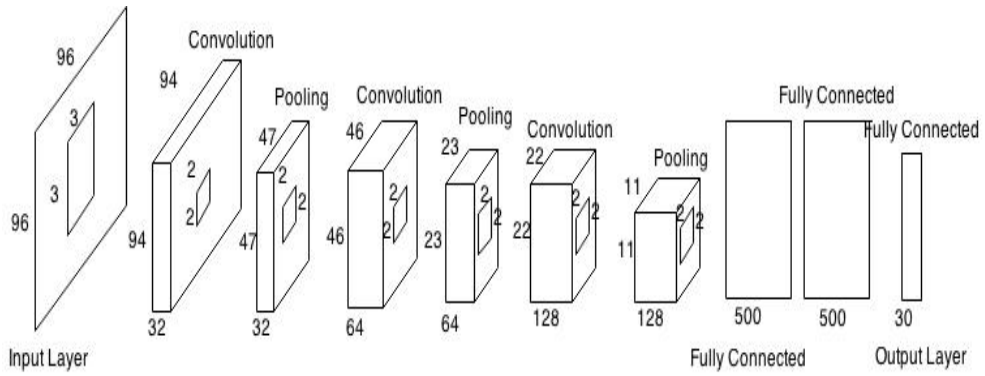
First, let's talk about the cost function. The cost function of the neural network is the average of the cost of each example. Where the cost for one example is the sum of the difference between the output given by the network and the real result at a square. Then you minimize this function with a gradient descent step. That's mean that for each output neurons you modify the bias or weight of previous neurons and do it for every previous neuron. But for that, we need to have the gradient of the cost function. But the gradient is really complicated to get. So we use a trick. As the gradient of this function got a dimension equal to your number of example to calculate an equivalent of it. For that, we take a little part of our training dataset (100 for example) and apply a gradient descent on all those little data set. It's less efficient but it's far faster. This method is a stochastic gradient descent.

1.7 Neural network structure

1.7.1 Structure

There are no explicit rules about making neural networks. The best thing to do is to seek for similar problems taken a structure that I've been already done and try to improve it. It's important to note that for convolution neural network convolutional layer are followed most part of the time by a relu layer and by a pool layer. Here an example of a cnn structure INPUT -> [CONV -> RELU -> POOL] * 2 -> FC -> RELU -> FC

On the picture below we can see another example of cnn architecture. We can see the dimension of the layer change at each layer. During a convolution layer the "width" of the layer increase. The width of the convolutional layer represents the number of filters that have been applied in on the input. The dimension decreases during the pooling layer depending on the size of the pooling.



Exemple of a neural network architecture

1.7.2 Hyper parameters

Hyperparameters are the choice you make to shape your CNN. Here some indication about how choosing hyper parameters. Filter size change with the input size. 28*28 image can have a 5*5 filter bigger images can have a 15*15 filter. Max pooling is generally 2*2 but a really high definition image can justify a 4*4 max pooling. The learning rate controls how much to update the weight in the optimization algorithm. Number of epochs is the number of times the entire training set pass through the neural network. The batch size defines the number of samples to work through before updating the internal model parameters (back propagation). The activation function is used to determine the output of neural networks like yes or no. It maps the resulting values in between 0 and 1 or -1 to 1 etc.

1.8 Resume

Different step of how cnn works
step 1 : choose the shape of the neural network and initialize weight with random values.
step 2 :the network take one training image.
step 3 :do it for each image in the sub dataset
step 4 : calculate the cost of the sub dataset
step 5 : back propagation using gradient descent
step 6 : repeat until you did it for all sub dataset.
step 7 : try the neural network on the test dataset
step 8 : use your neural network for classify new image.

2 Transfer learning

If you want to create your own neural network from scratch it will take a lot of time for computation and a huge dataset for training. Moreover there already existing models that are really good for classifying image. For example imageNet and inception. That is why if you have a specific dataset you can use transfer learning to gain time or getting better performance. "In transfer learning, we first train a base network on a base dataset and task, and then we repurpose the learned features, or transfer them, to a second target network to be trained on a target dataset and task. This process will tend to work if the

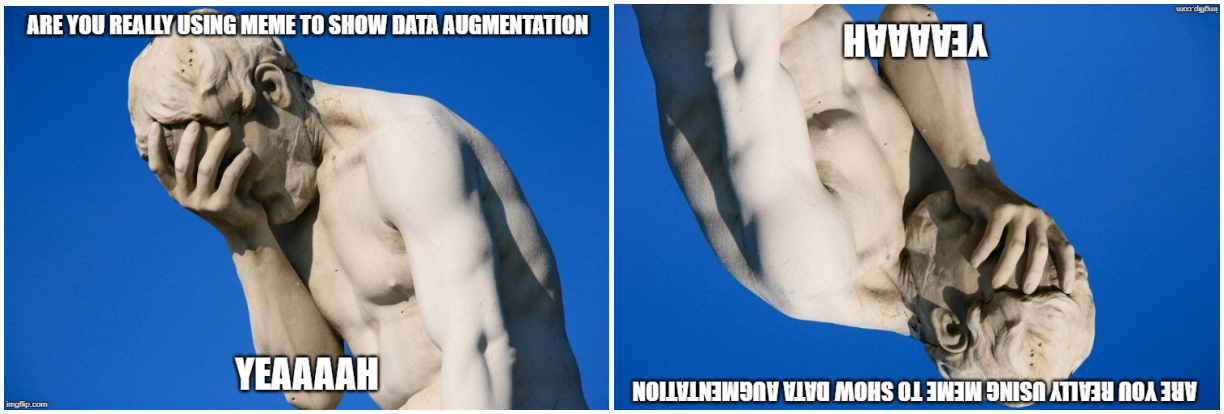
features are general, meaning suitable to both base and target tasks, instead of specific to the base task." Transfer learning method is use a lot for problems that use images. This is an effective method but have to be use carefully because the last layers of neural networks are generally specific to the dataset. For transfer learning, you have to preprocess your dataset to make the transfer learning possible.

3 Data augmentation

Before we dive into the various augmentation techniques, let's decide whether we perform transformation on the dataset before train or if we will do transformation during on batch just before train the CNN. If you can't afford to multiply the size of your dataset by 5 or so because you don't have enough hard drive or because your dataset is big the best solution is to augment your data on the fly (during the training). If you have a small dataset then just perform all the transformation before.

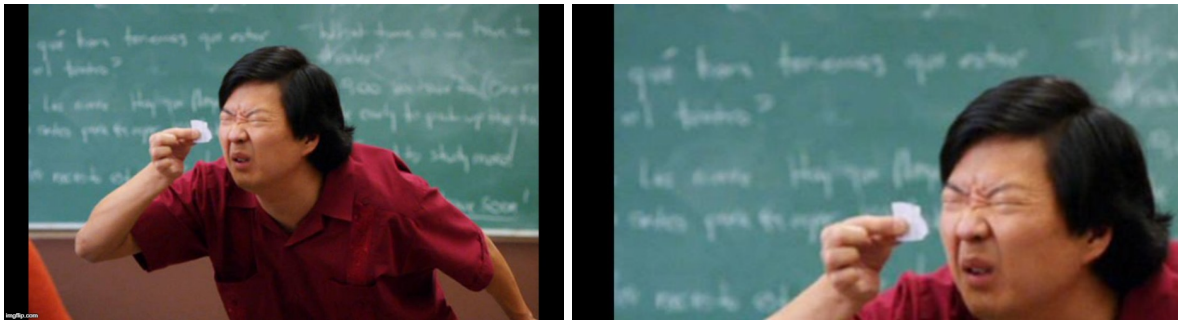
3.1 Do a barrel roll

You basically flip the image. Rotation works to.



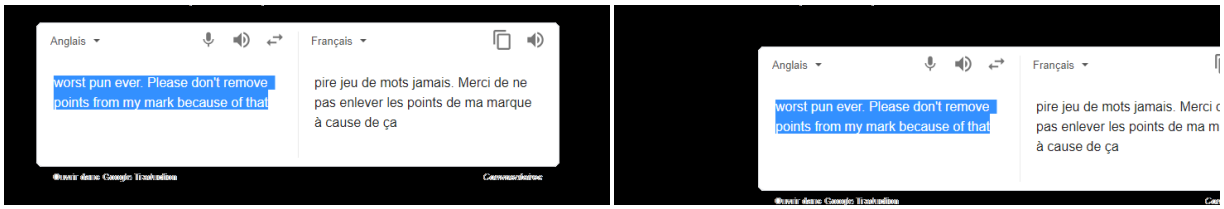
3.2 Scaling

Zooming the image. Almost the same as random cropping.



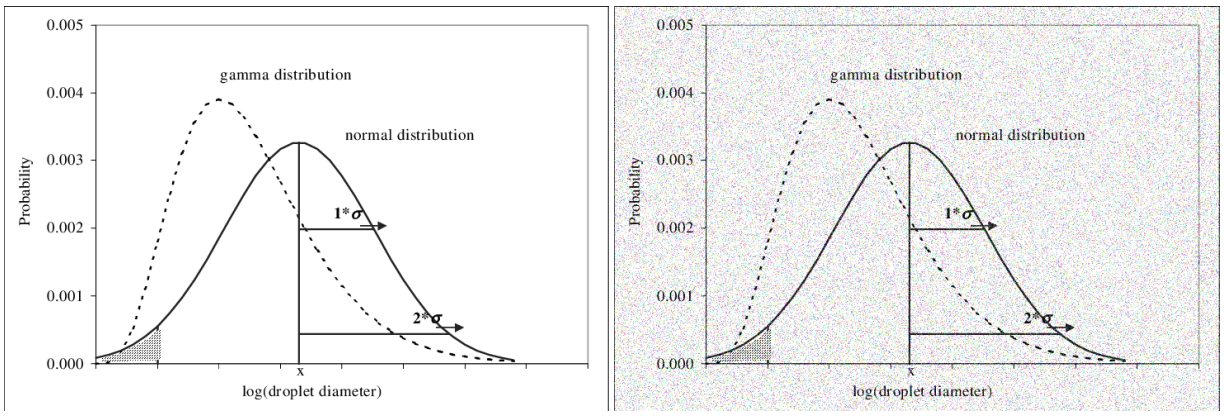
3.3 Translation

Translation just involves moving the image along the X or Y direction



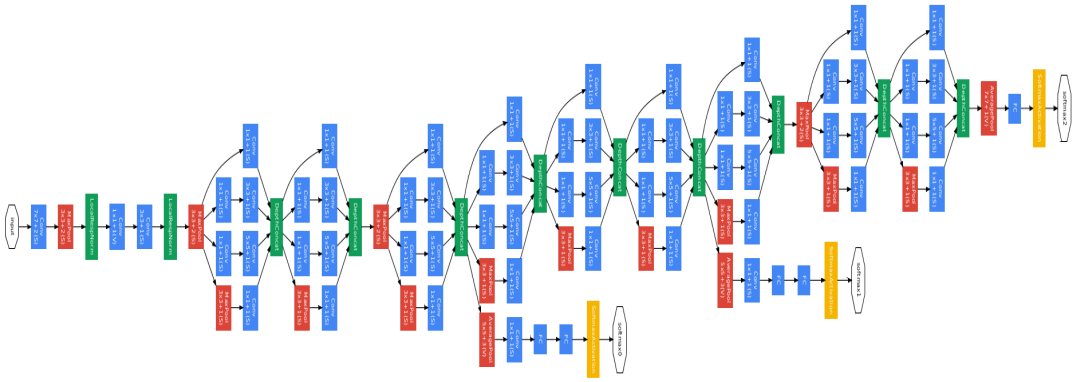
3.4 Gaussian noise

You can add some noise.



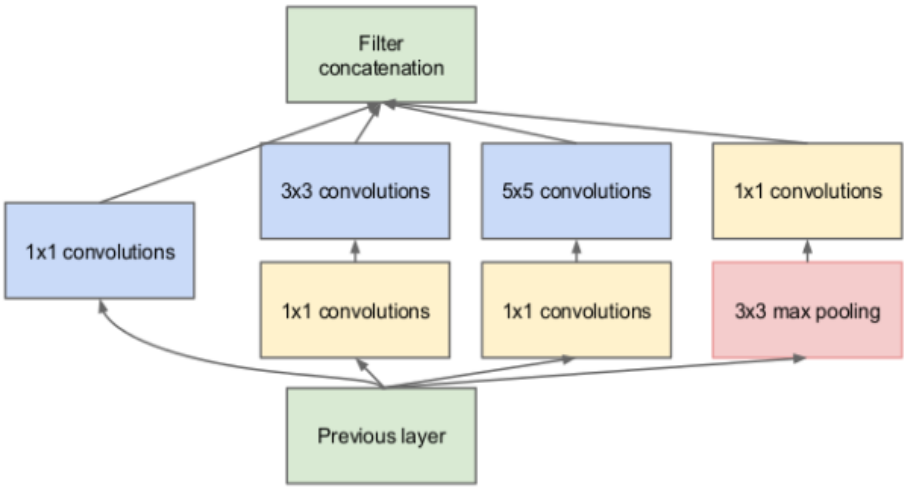
4 Inception model

inception is a model created by google recently. It's a convolutional neural network for classification tasks. One of the most accurate model, trained on more than 1000 classes.



Inception architecture

The main idea of inception I why would I bother me to choose between 1x1, 3x3, 5x5 convolution or pooling when I can do all of them and see what is the best option? inception do it by doing each convolution in parallel and concatenate the resulting feature map before going to the next layer. Now we got the basic idea lets look at the architecture of one layer.



Architecture of an inception module

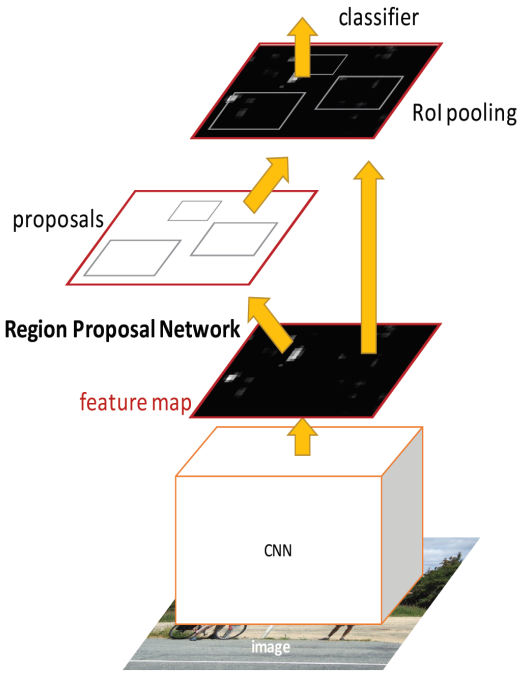
5 Detection neural network

In this section we are going to see tow main method for object detection with neural network. The first one is faster R-cnn.

5.1 Faster R-cnn

5.1.1 Architecture

The first part of Faster R-cnn is a region proposal network that will generate boxes where there are potential objects in it. The second part is a RoI pooling. The goal of this part is to propose regions with an object in it.



(1)

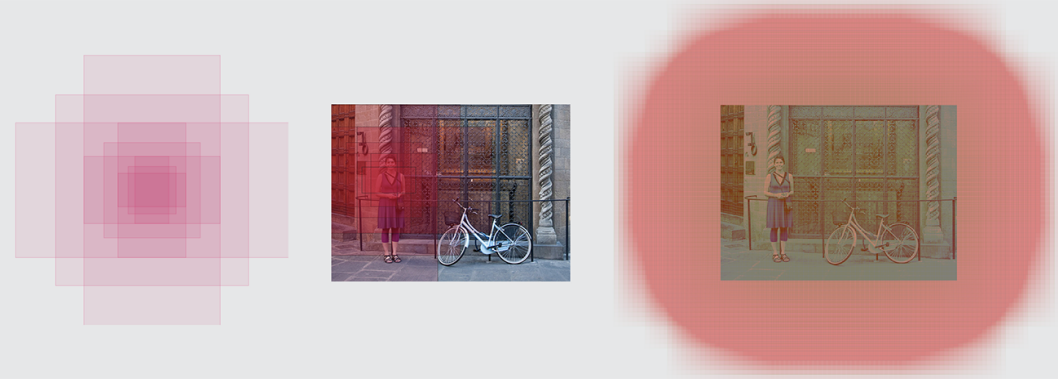
$$Faster R-cnn architecture. \tag{2}$$

5.1.2 Feature extractors

The goal of this first part is to have a good feature map so we need a convolutional neural network. This first part is called the feature extractor part. Most part of the time we use cnn as VGG16, ResNet, Inception and MobileNet for this part.

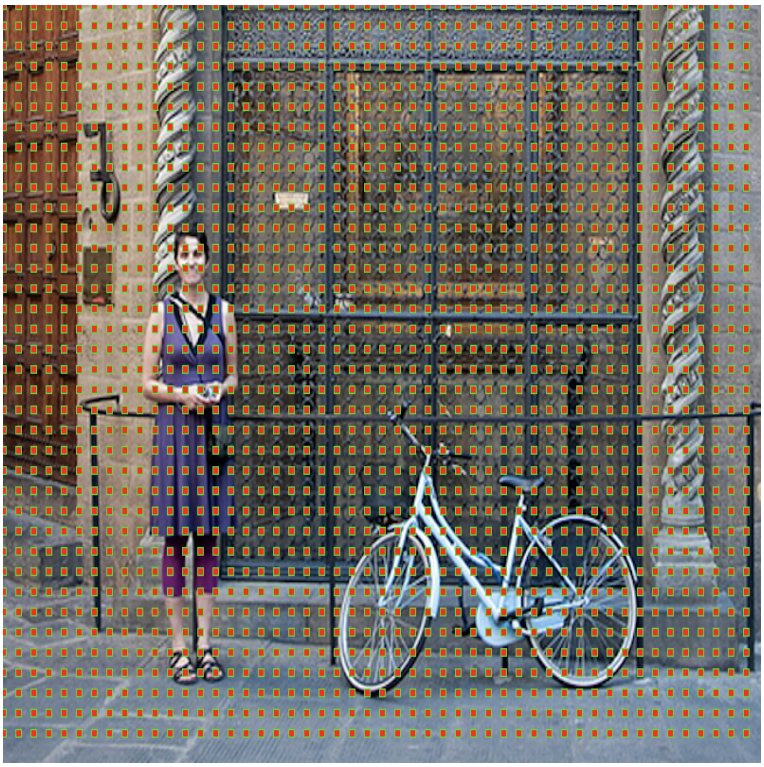
5.1.3 Region proposal network

Before going into the region proposal network we need to know what is an anchor. "Anchors are fixed bounding boxes that are placed throughout the image with different sizes and ratios that are going to be used for reference when first predicting object locations."



Different representations of bounding boxes

Anchors are based on the feature map but final anchors are referenced on the original image.



Representation of anchor

The main part now is to choose which anchor contains an object. For that, the rpn give 2 value for each anchor. The first one is a score of objectness and the second one is the bounding box regression for adjusting the anchors to better fit the object. The score objectness is determined by the percentage of background in the anchor. That is why we got to train first the rpn on a dataset of hand label boxes to train it to recognize background from objects. This method to determine if an anchor representing an object is called Intersection over Union. To avoid getting a lot of proposals for the same object we apply a nonmax suppression algorithm. The nonmax suppression algorithm will take the anchor with highest objectness score and delete all the boxes that overlap too much on the first boxes. Usually, the algorithm allows a superposition of 60 % to avoid loosing too much proposals and avoid having too much proposals. The last problem is that the proposals are not always of the same size. To solve this problem we use a

max pooling method to resize all the proposal.

6 Detecting lung cancer with CNN

6.1 Dataset

All the data used in the project was downloaded from the LUNA16 Challenge (https://luna16.grand-challenge.org/). The main objective of LUNA is to improve deep learning models dedicate to lung cancer image analysis method.

This dataset is composed of 888 CT (computed tomography) scans with annotations describing coordinates and ground truth labels. There are about 200 images in each CT scan. There were a total of 551065 annotations with 1351 were labeled as nodules, others were labeled negative. There are 3 nodule categories. (nodule $>$ 3mm, nodule $<$ 3mm, and non-nodule $>$ 3mm)

6.2 Prepossessing

As 551065 annotations where negative and only 1351 were labeled as true positives. To avoid getting a CNN specializes to recognize healthy lung instead of unhealthy lung we down sample the negative class and over sample the positive class. To over sample the positive class we use the well known method of image rotation.

6.3 Training

Then we had to divide the dataset into 3 parts Training 60%, testing 20%, validating 20%. Validating part is to estimate how well your model has been trained and to estimate model properties. The testing part is to try our model with "new data". This CNN network includes 3 convolutional layers of 50 filters which are 3 x 3, a ReLU as an activation function, 2 pooling layers has a kernel size 2 x 2, a dropout layer with probability of 0.5, in which the unit will be hidden in 50%, to prevent over fitting. Two fully connected layers and a softmax function is following after that.

6.4 Results

We have 3 measurements to evaluate our CNN. Accuracy sensitivity and specificity. The accuracy is the number of correct predictions made as a ratio of all predictions made. The sensitivity is the measure of the proportion of actual positives that are truly positive. Finally, the specificity calculates the percentage of actual negatives that are really negative. After testing the CNN model on 1622 images, we noticed that this model has an accuracy of 93%, a sensitivity of 0.794% and a specificity of 96%.

		Predicted label		
		Non-Cancer	Cancer	Total
True label	Non-Cancer	1281	59	1340
	Cancer	55	227	282
	Total	1336	286	1622

$$ACC = \frac{TN+TP}{TN+TP+FN+FP} = \frac{1281+227}{1281+286+55+59} = 0.93\%$$

$$SE = \frac{TP}{TP+FP} = \frac{227}{227+59} = 0.794\%$$

$$SP = \frac{TN}{TN+FN} = \frac{1281}{1281+55} = 0.96\%$$

Even with the oversampling of the true positive dataset we still have a sensitivity less important than the specificity. We could try to change the oversampling method and compare the result. Or maybe down sampling the negative class even more. I am not sure but the weak sensitivity could be the reason why our specificity is so good. More over it could be interesting to try our model on the original dataset to see if the sensitivity is affected by the oversampling. So we can say we still have a lot of progress to do because if we apply this model to new lung cancer case approximately 19.5% of people who got a cancer will not know they have a cancer.

References

[1] https://tryolabs.com/blog/2018/01/18/faster-r-cnn-down-the-rabbit-hole-of-modern-object-detection/

[2] https://medium.com/@smallfishbigsea/faster-r-cnn-explained-864d4fb7e3f8

[3] https://arxiv.org/pdf/1506.01497.pdf

[4] https://medium.com/nanonets/how-to-do-image-segmentation-using-deep-learning-c673cc5862ef

[5] https://www.youtube.com/watch?v=aircAruvnKk

[6] https://www.youtube.com/watch?v=IHZwWFHWa-w&t=239s

[7] https://www.youtube.com/channel/UCWN3xxRkmTPmbKwht9FuE5A