Hochschule für Technik
und Wirtschaft Berlin

University of Applied Sciences

Independent Course work Report

Object Detection with Bounding Boxes through Machine Learning

Caglar Özel

supervised by
Prof. Dr. Klaus Jung

Contents

Abstract Today where computer software and technologies are developing fast and new usages are being discovered frequently, Machine Learning is evolving and redefining Image Processing and Object / Image Detection as we knew it.

In this report I will talk about visual information retrieval (computer vision) through machine learning where I will use state of the art object detection systems depending on region proposal algorithms such as faster region-based convolutional neural networks (Faster R-CNN) and single shot detection (SSD) or also known as you only look once (YOLO) networks.

The end goal is to understand how the training proces of these models work and have a application which will utilize the Tensorflow framework with the Neural Network Models it uses to perform Object Detection on visual media. The result of this Object Detection will be an image, video or live footage with objects highlighted through bounding boxes.

Introduction While humans effortlessly recognize shapes and objects through our complex visual cortex system where we have millions of neurons and billions of connections between them, which were fine tuned through generations. The difficulty of visual pattern detection becomes apparent when you try implemented the logic in a algorithm. [1]

Object detection which is a technology related to computer vision or image processing is a field that deals with the detection of objects in a digital image or video.
Machine Learning on the other hand is a subset of Artificial Intelligence where the computer will perform tasks through algorithms and models without explicit instructions, relying on patterns and models instead.

Combining these two technologies and the recent advances in object detection through region proposal methods and region-based convolutional neural networks opened up new possibilities which were impossible to achieve before through low level feature analysis only such as mean color comparison, shape and filter analysis or other methods which were common before.

Region proposal methods typically rely on inexpensive features and economical inference schemes. Selective Search, one of the most popular methods, greedily merges superpixels based on engineered low-level features. Yet when comparing to efficient detection networks, Selective Search takes 2 seconds per Image on a CPU implementation, where as fast region-based CNN's take advantage of the GPU and therefore are not applicable for comparison. [2]

1 Brief introduction to Neurons and Networks

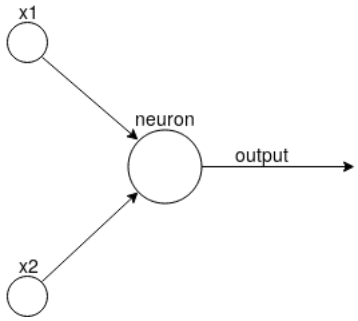
Similar to the neurons in the human brain, neurons in computer science are approaching the same concept and same ideology.
The functionality of a neuron is to retrieve data, possibly and ideally multiple at the same time, process and output a single result. There are many types of neurons, for introduction purposes I am just gonna mention two here.

- Perceptrons
- Sigmoids

The more commonly used artificial neuron in Neural Network models is the Sigmoid. The reasons for that will be mentioned in the chapters belowe.

1.1 Percpetrons

Perceptrons were developed in 1950 to 1960 by the scientist Frank Rosenblatt, inspired by earlier works from Waren McCulloch and Walter Pitts.

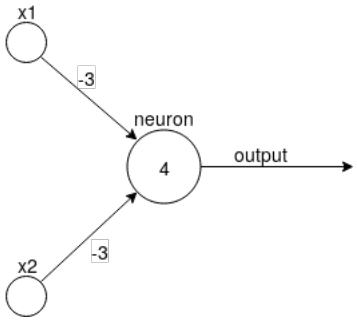


In this example our neuron has 2 input variables and one output, in theory it can have more or less. Further more Frank Rosenblatt proposes the concept of weights for each input variable defining how important a variable is compared to another.

$$output = \begin{cases} 0 \text{ if } \sum_j w_j x_j \leq threshold \\ 1 \text{ if } \sum_j w_j x_j > threshold \end{cases}$$

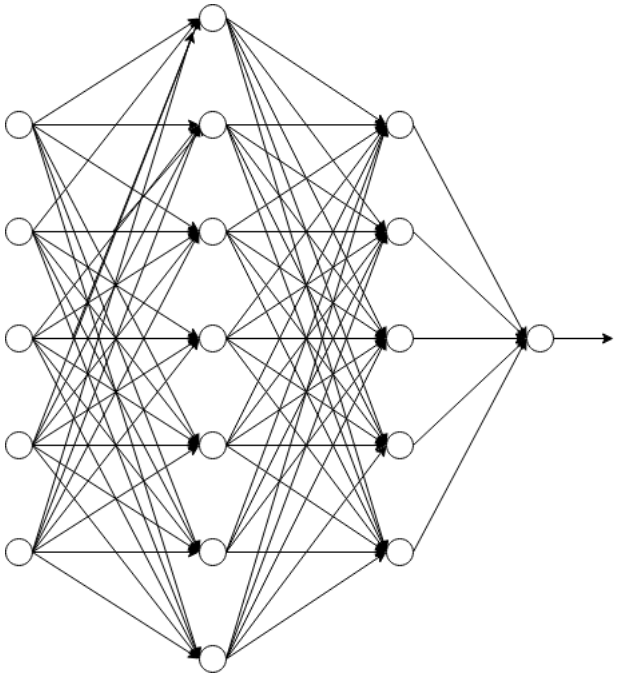
A single neuron is by no means comparable to a decision making possibilities of a human being, but it can weigh variables and make decisions

accordingly by defining a threshold for when a true or false gets output. Another way to create a conditional behavior is to use a bias instead of a threshold.



$$output = \begin{cases} 0 \text{ if } \sum_j w_j x_j + b \leq 0 \\ 1 \text{ if } \sum_j w_j x_j + b > 0 \end{cases}$$

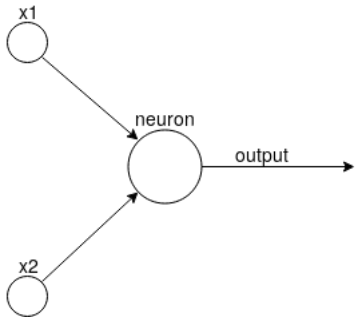
Making it obvious that they are able to perform more complex decisions in layers.



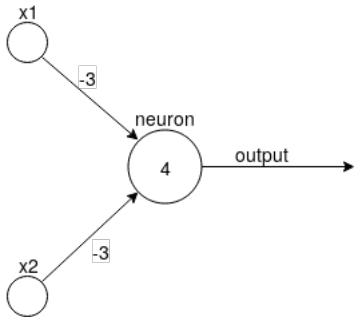
The columns in this network are called layers, the first column is refered to as the first layers and the last one as the output one all the ones in between are called hidden layers.
Since the result of a perceptron can only be one or zero (true or false), one major problem someone faces when using perceptrons in more complex neural networks is that biases and weights can have weird and unwanted effects. Making perceptrons trigger where they should not, thats where sigmoids come in handy and build on top of the concept of perceptrons.

1.2 Sigmoids

The core principle of a sigmoid neuron is the same as the perceptrons. Just like perceptrons, sigmoids have input and output variables but instead of just being 0 or 1 it can be anything in between 0 and 1 so 0.5234 is a valid output of a sigmoid.



It can also have weights and biases similiar to the perceptron:



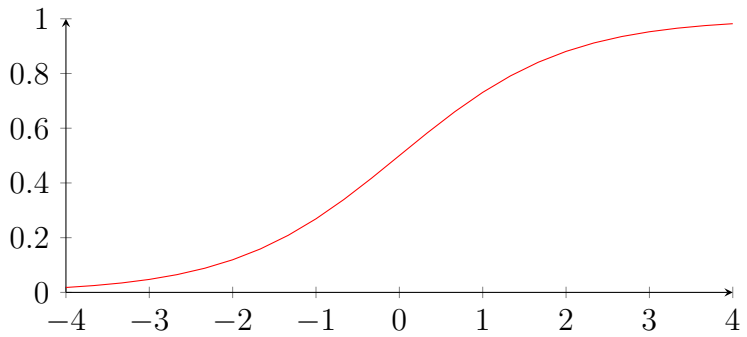
The arithmetic function on the other looks different and is called the sigmoid function or sometimes the logistic function.

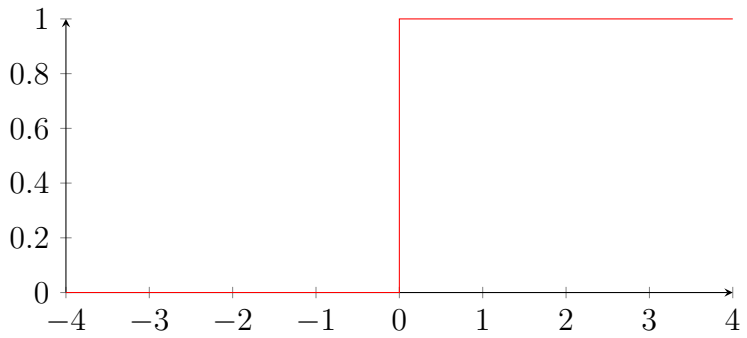
$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

To put it more explicit with input variables, weights and bias:

$$\frac{1}{1 + exp(-\sum_j w_j x_j - b)}$$

Since the algebraic form of σ a is smoothed and not a step function this difference is minimal but makes a huge difference in the way how weights and bias influence the result of a sigmoid compared to a percpetron.





2 Current state

Object Detection in dealing with two major problems and therefore can be split into two processes, these are object localization and object classification of multiple objects in one image. There are many approaches in current Neural Network models to perform these actions. There are models which have a multi layer architecture, each performing complex tasks such as analyzing, localizing and classifying each sections individually, making the cost high and the network in general slow but accurate. There models that combine functionality in one layer so that one layer for instance localizes and classifies at the same time, is making the model more flexible, smaller and therefore faster but less accurate. The trade off between accuracy and speed is something that all networks deal with nowadays especially if the goal is to have a near real life object detection system which is still reliable and accurate when doing so.
There are multiple algorithms covering various forms of this formula, called RCNN, F-RCNN, SSD, YOLO but first we will cover the approach of detecting interesting regions in an image because this is a key problem in object detection.

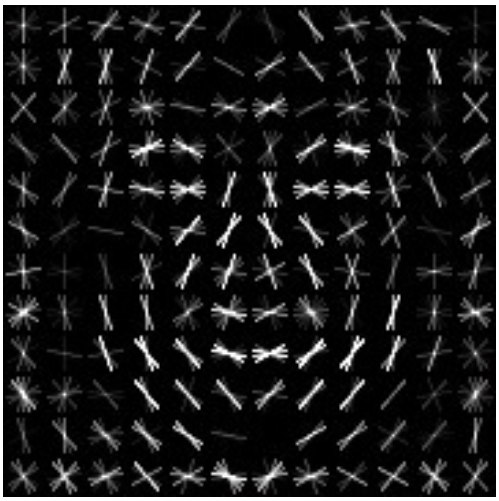
2.1 Approaches to Object Localization

The segmentation of an image is the key approach to Object Localization, since images are always hierarchical the segmentation as well has to be hierarchical. It has to be able to cover different criteria and forms, where maybe objects are part of a bigger total or just single parts inside another one. Since is not feasible to compute every possibility inside an image there has to be some kind of separation through grids and scales.
As there are many approaches to algorithms how to analyze and classify regions, there are many approaches for algorithms to detect interesting sections of an image which may contain objects.

2.1.1 Exhaustive Search

Since an object can be located anywhere in an image the scope to look for it is enormous making the computational cost for such search very expensive. To compensate for this cost most exhaustive search approaches such as the sliding window approach, constraint the size and aspect ratio of the window over the grid, also using weak classifiers and economic image features such as histogram of oriented gradients (HOG). [3]
HOG approaches the problem of recognizing and detecting objects in an unknown image through detecting corners. It analyzes the distribution and their orientation through which he is able to separate the image in multiple partitions or detect similar regions in different images. Robert K. McConnell described the concept in a patent in 1986 but it became

known in 2005 through a publication of Navneet Dalal and Bill Triggs. [**?**]

 [**?**]

2.1.2 Selective Search

A selective search algorithm is subjected into 3 design ideas:

- Capture All Scales

  The idea is similar to the one of the Exhaustive Search that all objects have to be found. Therefore all scales have to be considered as a potential region. The best approach is a hierarchical grouping, where a initial sub segmentation has to be performed. [3]

   [4]

  After the segmentation a grouping of regions is done by similarity. After each grouping a similarity between resulting regions and its neighbors are being calculated. This process is done until the whole image becomes a single region. [3]

 [5]

- Diversification

  There is no single strategy for grouping regions together therefore multiple aspects have to be considered. Selective search diversifies by using a variety of color spaces with different invariance properties, by using different similarity measures and by varying the starting regions. By using different color spaces, selective search is able to account for different scene and light conditions. To account for these variances selective search is performing the hierarchical grouping algorithm with different color spaces which have different invariance properties.

- Fast to Compute

  The fast computation is being achieved by four complementary similarity measures. All these measures are in range of [0, 1] which facilitates combinations of these measures. [3]

  – S$color$

    Measures color similarity specifically for each region where a color histogram of 25 bins is being created and normalized with the L1 norm. Similarity is measured using the histogram intersection: [3]

$$S_{color}(r_i, r_j) = \sum_{n=1}^{n} min(c_i^k, c_j^k)$$

[3]

The histogram can be efficienlty propagated through the hierarchy by:

$$C_t = \frac{size(r_j) * C_i + size(r_j) * C_j}{size(r_i) + size(r_j)}$$

[3]

– S*texture*

Selective Search measures texture similarity through the usage of the SIFT algorithm. SIFT stands for scale invariant feature transform and is a algorithm to detect and describe features in a region. Further more it takes Gaussian derivatives in eight orientations for each color channel, where a 10 bin histogram is being extracted. Similarity is being measured again using the histogram intersections and the propagation is function is the same as for the color. [3]

$$S_{texture}(r_i, r_j) = \sum_{n=1}^{n} min(t_i^k, t_j^k)$$

[3]

– S*size*

– S*fill*

2.1.3 Region Proposal Network

2.2 Approaches to Object Classification

Tensorflow

Application

Conclusion

References

[1] Michael Nielson. Neural networks and deep learning. http://neuralnetworksanddeeplearning.com, 2018.

[2] Ross Girshick Shaoqing Ren, Kaiming He and Jian Sun. Faster r-cnn: Towards real-time objectdetection with region proposal networks. https://arxiv.org/pdf/1506.01497.pdf, 2015.

[3] T. Gevers J.R.R. Uijlings, K.E.A. van de Sande and A.W.M. Smeulders. Selective search for object recognition. http://www.huppelen.nl/publications/selectiveSearchDraft.pdf, 2012.

[4] image with segmentation. https://www.learnopencv.com/wp-content/uploads/2017/09/breakfast_fnh.jpg.

[5] image with bounding boxes. https://www.learnopencv.com/wp-content/uploads/2017/09/breakfast-top-200-proposals.jpg.