Πολυτεχνείο Κρήτης

Σχολή Ηλεκτρολόγων Μηχανικών & Μηχανικών Υπολογιστών

Diploma Thesis

Comparison of Artificial Intelligence systems for the detection of objects on UAV-based images.

TRIMAS CHRISTOS

CHANIA, JUNE 2021

Committee

Professor Zervakis Michalis(Supervisor)

Associate Professor Lagoudakis Michalis

Professor Petrakis Euripidis

Acknowledgements

First, I would like to thank my family who supported me all those years and never made a discount in my studies.

Next, I would like to thank Professor Michalis Zervakis and Dr. Marios Antonakakis who as my advisors guided me through the whole process of this thesis.

Last, but certainly not least, my dear friends who supported and helped me, through the difficult years of Electrical and Computers Engineering studies.

Περίληψη

Abstract

Contents:

1. Introduction

1.1) Unmanned Aerial Vehicles

Unmanned systems are typically known as powered vehicles that do not carry a human operator, can be operated autonomously or remotely and can carry a variety of payloads depending on their type, functionality and mission objectives.

Unmanned Aerial Systems, also known as a drone, have experienced the greatest growth. As of 2020, seventeen countries have armed UAVs, and more than 100 countries use UAVs in a military capacity. The global military UAV market is dominated by companies based in the United States and China. With extensive cost reduction in electronics, the defense forces around the globe are utilizing UAVs for applications such as logistics, communications, attack and combat, while commercial applications include aerial photography and filmmaking, cargo transport and detection of disasters [1].

Whether it comes to the detection of objects of interest (refugee waves, tracking and exterminating target), prison surveillance or information gathering of a battlefield, UAVs have proven their usefulness. A significant contribution to this development, played the evolution of cameras. The cameras on-board UAVs are a rich source of information that can be processed in order to extract meaningful information. Besides the cameras, the development of other advanced hardware and software technologies allow drones to carry out their missions without human intervention, such as computer vision, object detection, machine learning, thermal sensors and deep neural networks.

1.2) Object Detection

Object detection is a computer technology related to computer vision and image processing that deals with localization and identification of semantic objects of a certain class, in digital images and videos. In other words, given an image or a video stream, an object detector can identify-classify objects of interest and provide information about their positions within the image.

With the evolution of cameras and the oversimplification of data gathering and processing, object detection can be used in the following military and commercial areas:

- Surveillance.
- Search and Rescue missions.
- Anomaly detection.
- Autonomous driving

The basic idea of object detection, is that every object class has its own special features that helps in classifying the class- for example all circles are round. Object detection models learn those special features and create patterns on the objects properties. Features may be specific structures in the image such as points or edges. More broadly a feature is any piece of information which is relevant for solving the computational task related in computer vision applications. The feature extraction process can be a computational expensive and many times due to time constraints, a higher level algorithm may be used to guide the feature detection stage, so that only certain parts of image are searched for features.

There are two kinds of object detection methods:

1) Neural Network approaches.
2) Non-Neural approaches.

Non-Neural approaches use one of the following techniques for feature extraction and an algorithm such as Support Vector Machines for classification of those features.

- **Viola-Jones object detection framework based on Haar features.**
- **Scale-Invariant feature transform.**
- **Histogram of oriented gradients features.**

Neural Network approaches can be distinguished in to two-stage detectors and single-stage detectors. The first ones use a box proposal algorithm as the first stage, and the second stage classifies those proposals, while the second ones detect objects and classify them in the image in one pass through the network.

The most known detectors are:

- **Regional Proposal Networks such as R-CNN.**
- **Single Shot MultiBox Detector.**
- **You Only Look Once (YOLO).**
- **Retina-Net.**

2. Artificial-Intelligence, Machine-Learning, Deep-Learning

Nowadays, the word Artificial Intelligence or A.I. sounds everywhere and it is used increasingly. A.I. refers to the simulation of human intelligence in machines that are programmed to think like humans and mimic actions. The term may also be applied to any machine that exhibits traits associated with a human mind such as learning and problem-solving.

The most common applications of A.I. are: autonomous cars, voice and face recognition, data analysis, virtual assistance and other applications in various industries. Subfields of Artificial Intelligence are machine learning and deep learning.

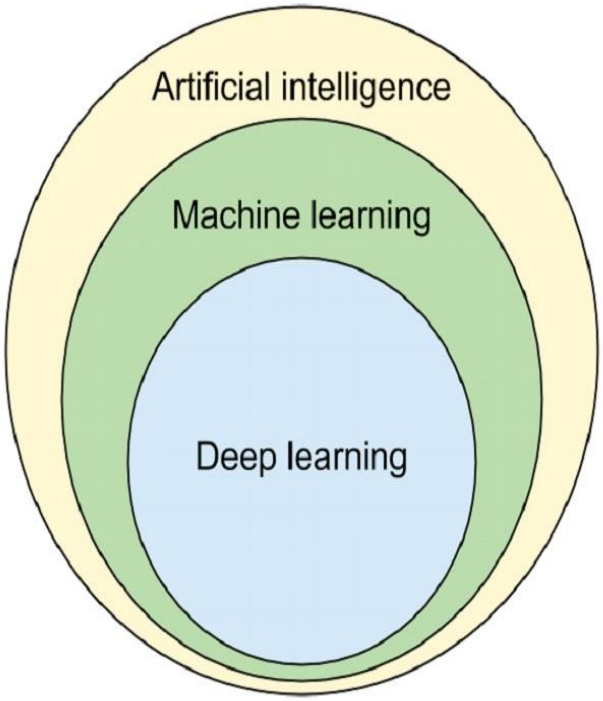

Image [1]: A.I., M.L. and D.L.

2.1) Machine Learning

The concept of machine learning dramatically changes the way of how classical programming works. In the classical method, someone provides the data and defines the rules of the program to obtain an answer. In machine learning or ML, someone give the data with the answers and demands from the machine to create the rules. The rules can then be applied to a new data to confirm the results and to generate new

answers. In other words, ML consists of algorithms that improve automatically through experience and by the use of data.

A subset of Machine Learning is Deep Learning.

2.2) Deep Learning

From Machine Learning Deep Learning was born. D.L. is part of a broader family of machine learning methods based on artificial neural networks with feature learning. Deep-learning architecture such as deep neural networks and convolutional neural networks have been applied to fields including computer vision and image analysis.

A Deep Neural Network (DNN) is an artificial neural network with multiple layers between the input and the output layers. In computer vision the most used class of deep neural networks are Convolutional Neural Networks or CNNs.

2.3) Convolutional Neural Networks

A Convolutional Neural Network is a Deep Learning algorithm which can take in an input image, assign importance to various objects in the image and be able to differentiate one from the other. ConvNets require less pre-processing compared to other classification algorithms.

The architecture of a CNN is analogous to that of the connectivity pattern of Neurons in the Human Brain and was inspired by the organization of the visual cortex. Individual neurons respond to stimulations only in a restricted region of the visual field known as the Receptive Field. A collection of such overlap cover the entire visual area.

An image is a matrix of pixel values. A lot of times images contain objects that have pixel dependencies throughout the image. A CNN is able to successfully capture spatial dependencies in an image though the application of relevant filters and the network can be trained to understand the sophistication of the image better.

In image [2], an RGB image, which has been separated by its three color planes, is represented. Although this example has small dimensions, real images can reach higher dimensions, for example an 8K image has 7680x4320x3 dimensions, making object detection in such dimensions a computational intensive procedure. The role of CNN is to reduce the image into a form which is easier to process the image, but at the same time without losing features which are critical for getting good predictions.

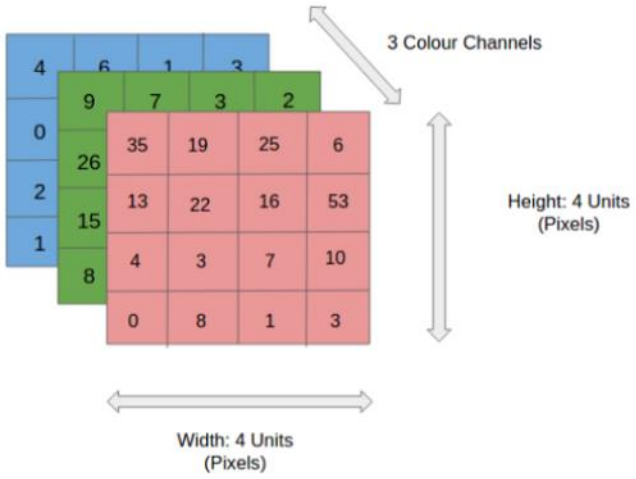

Image [2]: An RGB image

ConvNets, usually, are divided into two parts, the convolutional and the densely connected. The first one applies various layers such as Convolution and Pooling to reduce the dimensions and retain the important features of the image, while the second one is responsible for classification. In the following image [3], an example of a CNN architecture is shown.

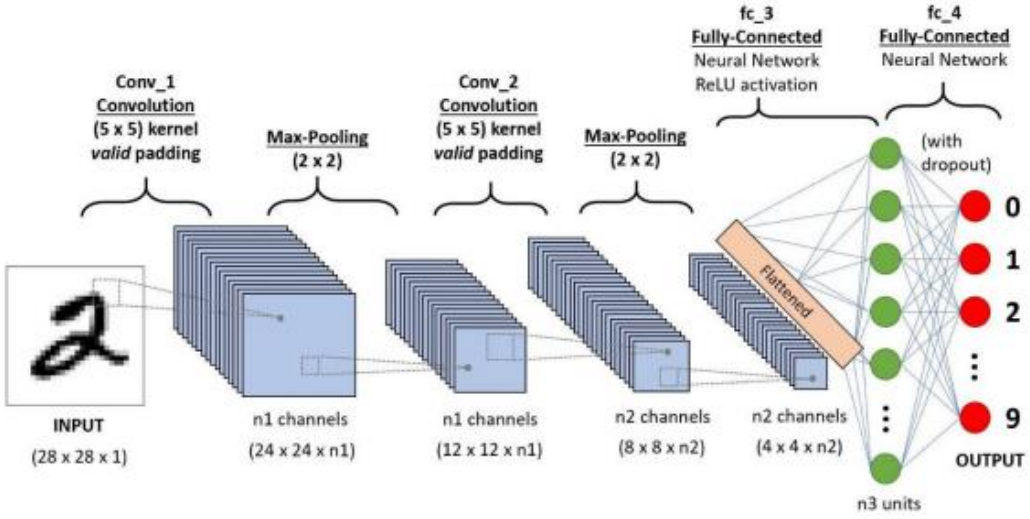

Image [3]: A 4 layer CNN.

2.3.1) Convolution Layer

In a ConvNet, the input is an image (tensor) with a shape: (H) x (W) x (C), representing height, width and number of channels respectively. After passing the input through the convolutional layer, the image becomes abstracted to a feature map, with new shape: (Feature Map Height) x (Feature Map Width) x (Feature Map Channels).

Generally, a convolutional layer has the following attributes/hyperparameters:

- Convolutional filters, also known as kernels.
- The number of input and output channels.
- Padding (augmentation of the kernel) and Stride (size of the step the kernel parses an image).

A convolutional kernel is basically a matrix that is applied throughout the image. Each filter is convolved across the width and height of the input image, computing the dot product between the filter entries and the input, resulting to a feature map of that filter. The network learns filters that activate when it detects some specific type of feature at some spatial position in the input.

Given a two-dimensional Image **I** as input and a two-dimensional kernel **K** the convolution operation can be described [2]:

$$\boldsymbol{S(i,j) = (I * K)(i,j) = \sum_{m}\sum_{n} I(m,n)K(i-m,j-n)}$$

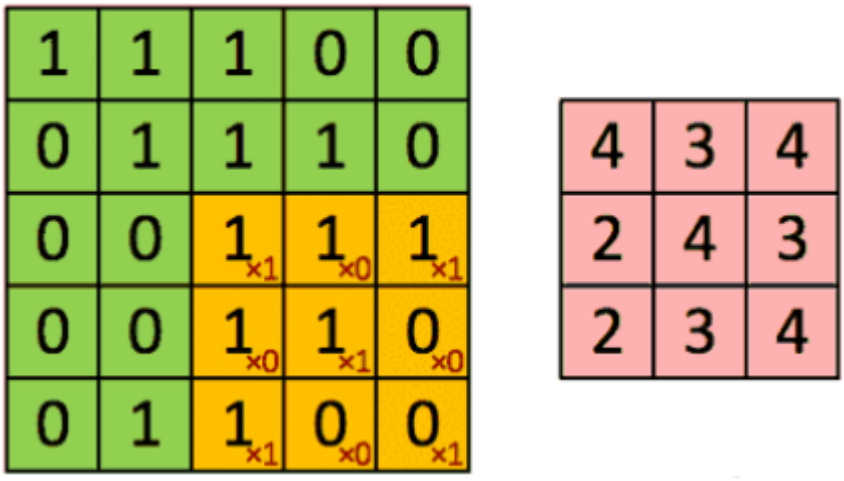

Image [4]: Original and Convolved Image.

In image [4], the kernel shifts 9 times in the orginal image, performing every time a matrix multiplication operation between the kernel and the portion of the image over which the kernel is hovering at the time. In this example, the filter parses the image with a stride of 1.

In cases of images with multiple channels such as RGB (Image [5]), the kernel has the same depth as that of the input image, and matrix multiplication is performed between the Kernels and each Channel. The results then are summed to give a squashed one-depth channel Convolved Feature Map.

The goal of Convolution operations is to extract high-level semantically rich features from the input image. One layer is not enough to achieve this, therefore

CNNs need not to be limited to only one convolutional layer. The first layers are responsible for Low-Level features such as edges. With more depth in the network, the architecture adapts to the High-Level features as well, providing a network which understands the whole image.

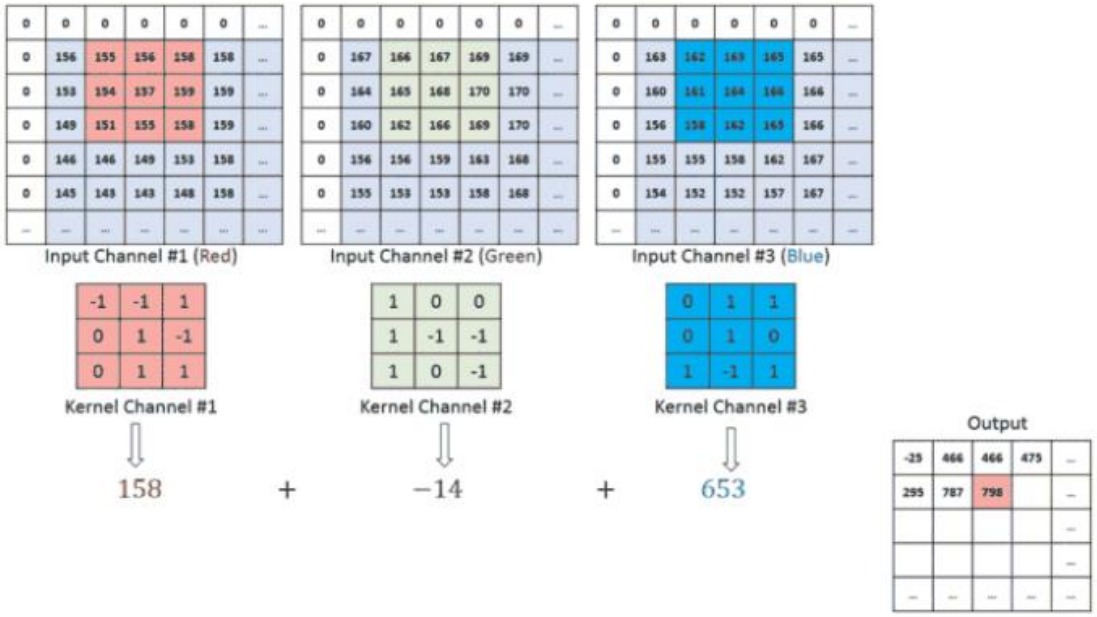

Image [5]: RGB example of convolution.

To add more layers (depth) to the network, there are two types of operation. One in which the convolved feature is reduced in dimensionality (Valid padding) compared to the input, and the other in which the dimensionality remains the same or it is increased (Same padding).

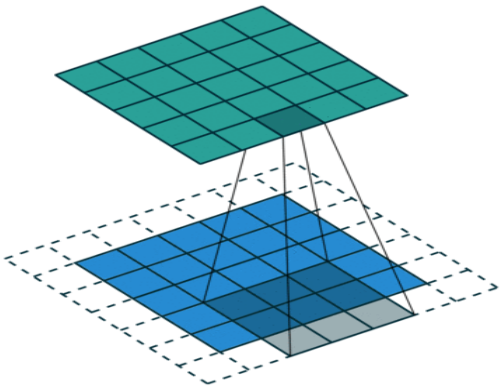

Image [6]: Same padding with zeros.

The same padding operation that is shown in Image [6] has been achieved by augmenting the input image from 5x5x1 to 6x6x1 and then applying the 3x3x1 kernel over the augmented image. If the valid padding operations was performed, the convolved matrix will have the same dimensions with the kernel.

2.3.2) Pooling Layer

Similar to the convolutional layer, the Pooling layer is responsible for reducing the spatial size of the convolved feature. The goal of pooling layer is to decrease the computational power required to process the data and to extract dominant features through dimensionality reduction.

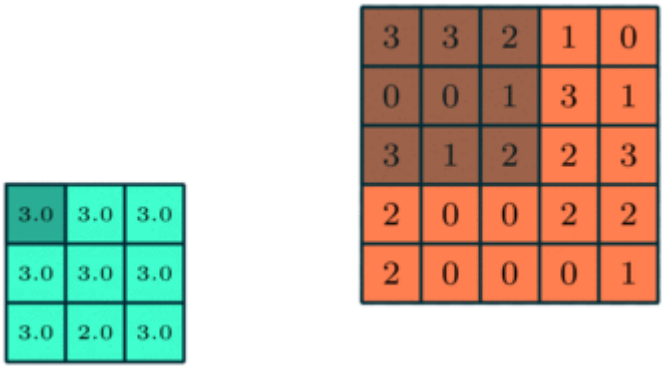

Image [7]: An example of pooling.

There are two types of Pooling: Max Pooling and Average Pooling.

Max pooling returns the maximum value from a portion of the image I covered by a kernel K. It can be used as a Noise Suppressant, discarding the boisterous activations altogether and hence performing de-noising and dimensionality reduction at the same time.

Average pooling returns the average of all the values from the portion of the image I covered by a kernel K and as result performs dimensionality reduction.

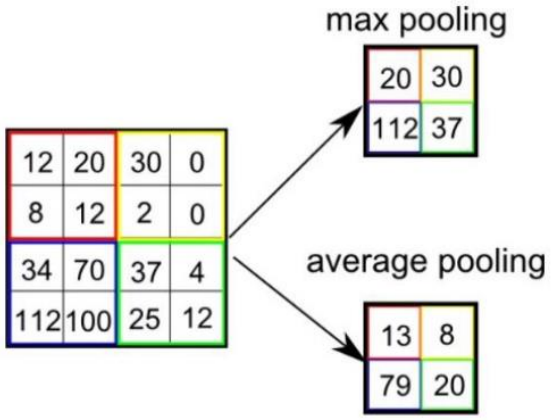

Image [8]: Avg and Max pooling.

After repeating convolution and pooling layers for several times, the model will successfully understand low and high level features. The final step is to feed those features to either an Artificial Neural Network or use another technique to perform classification.

2.3.3) Fully Connected Layer

Adding a Fully-Connected (FC) layer is a cheap way of learning non-linear combinations of the high-level features as represented by the output of the convolution and pooling layers. In more details, the input to the FC layer is the output from the final convolutional or pooling layer, which is flattened and then fed into the fully connected layer.

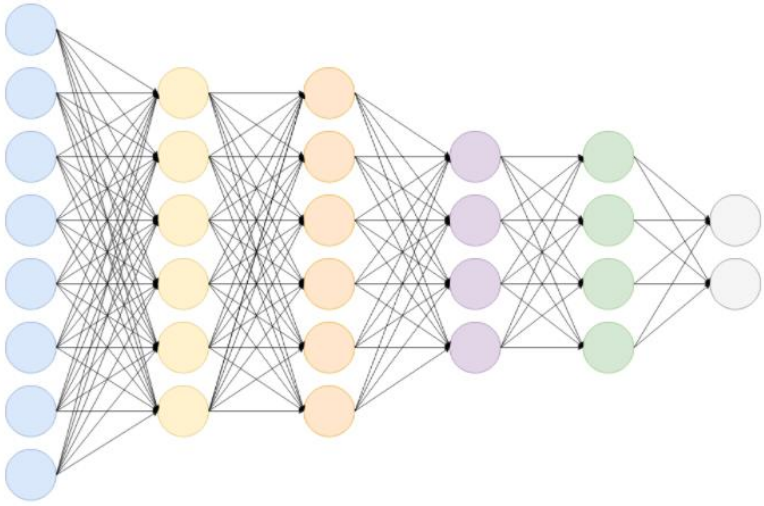

Image [9]: A Fully Connected Network.

The output after performing convolutions and pooling layers is a 3-d matrix. To flatten the output, each value of the matrix is stacked and the result is huge vector. The flattened vector is then connected to fully connected layers which are Artificial Neural Nets. Each layer of the ANN applies the following function:

$$\boldsymbol{g(Wx + b)}$$

Where,

x is the input vector with dimension: ***d1 = (number of neurons, 1)***.

W is the weight matrix with dimensions:

d2 = (number of neurons in previous layer, number of neurons in the current layer).

b is the bias vector with dimensions: ***d3 = (number of neurons in current layer, 1)***.

g is the activation function.

After passing through the FC layers, the final layer uses an activation function (see next subsection) to get the probabilities of the input and classify them into a particular class.

2.3.4) Activation Functions

Also known as Transfer Function, is a way to extract the output of a node in an Artificial Neural Network. It maps the resulting values in a well-defined space, depending the function.

Activation functions can be basically divided into 2 types:

1. Linear Activation Functions.
2. Non-Linear Activation Functions.

Linear of Identity Activation Function:

The function is a line ranging between (-∞, ∞).

Equation: $f(x) = x$.

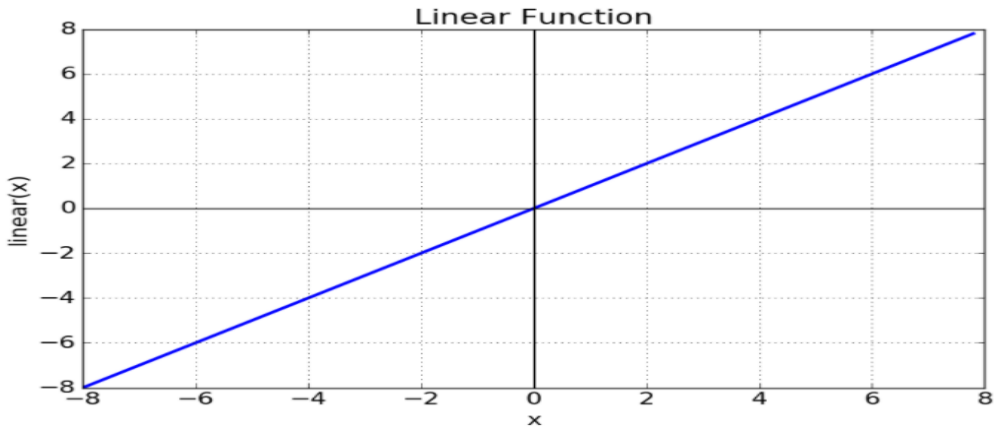

Image [10]: Linear Activation Function.

Sigmoid or Logistic Activation Function:

A sigmoid function ranges between (0, 1), therefore making it useful in models that predict probabilities as an output. The function is differentiable and monotonic.

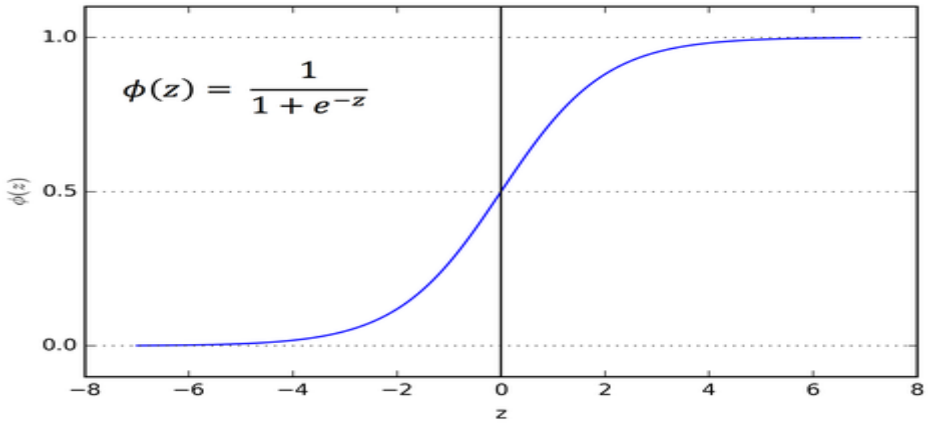

Image [11]: Sigmoid Function.

A huge disadvantage is that the logistic function can cause a neural network to get stuck during the training phase. This is because, if a strongly-negative input is provided, it outputs the value very near to zero. This behavior slowdowns the update of the learnable parameters, such as weights and bias.

Equation: $\boldsymbol{\varphi(z) = 1/(e^{-z} + 1)}$

Rectified Linear Unit (ReLU) Activation Function:

This function maps every negative value immediately to zero. It ranges from zero to infinity, and the function and its derivative are monotonic.

Equation: $\boldsymbol{R(z) = max(0, z)}$

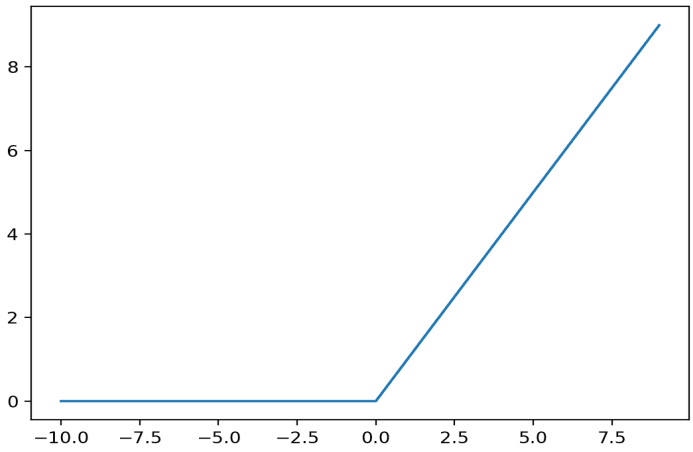

Image [12]: ReLU Activation Function.

Softmax Activation Function:

Softmax maps the output in range between [0, 1]. Furthermore, the total sum of the mapped output is 1. Therefore, the output of Softmax is a probability distribution.

Equation:

$$\boldsymbol{\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}}$$

For j = 1,…,K.

2.3.5) Loss Functions

In Neural Networks, the term Loss refers to the prediction error of the Neural Network. The calculation of the loss with the use of a function is called Loss Function. Loss function is responsible for the update of weights of the Neural Network.

There are various Loss functions and the selection of the one that fits the best to a model depends on various factors, such as the type of problem (Regression or Classification), the model architecture and many more.

Few of the most known Loss Functions are:

Cross Entropy. One of the most known loss functions, it measures the performance of a classification model whose output is a probability value.

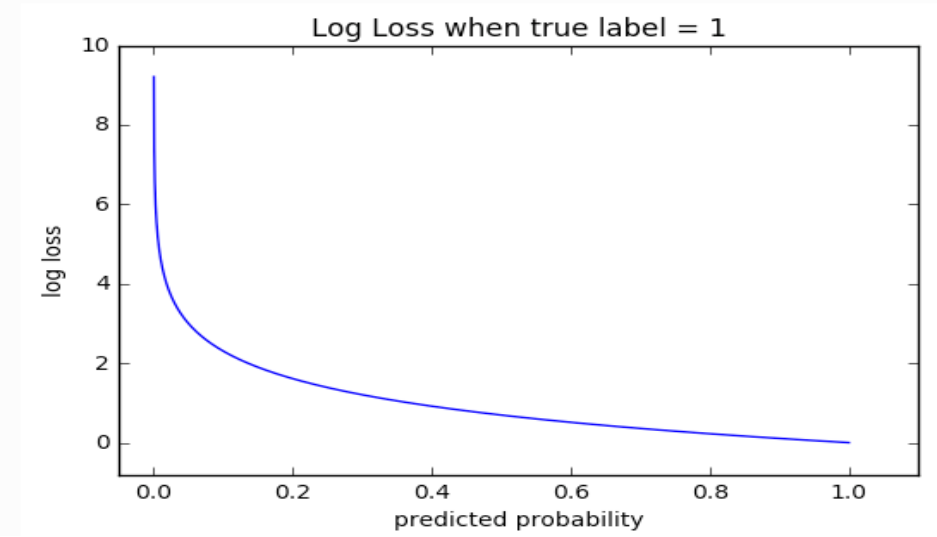

Image [13]: Cross-entropy loss.

Cross-entropy can be described by the following equation

$$L(\boldsymbol{\Theta}) = -\sum_{i=1}^{M} y_i \log(\hat{y}_i)$$

where, M is the number of the classes, log the natural logarithm, y the binary indicator (0 or 1) if class label i is the correct classification and y_hat is the predicted label.

Kullback-Leibler divergence. Also called *relative entropy*, it is the gain or loss of entropy when switching from distribution one to distribution two – and it allows to compare the two distributions.

KL divergence is primarily used in Variational Autoencoders or in multiclass classification scenarios. Mathematically can be described by the following equation:

$$KL(P||Q) = \sum p(X) log(\frac{p(X)}{q(X)})$$

Smooth L1. Mostly used in object detectors for bounding box regression.

After defining L1 or Manhattan distance:

$$L_1 := (a, b, N) \rightarrow \sum_{i=1}^{N} |a[i] - b[i]| \Longrightarrow (a, b, N) \rightarrow \sum_{i=1}^{N} |a_i - b_i|$$

The smooth L1 can be defined:

$$smooth_{L1} := (x) \rightarrow piecewise(|x| < 1, 0.5x^2, |x| - 0.5) \Longrightarrow$$

$$x \rightarrow piecewise(|x| < 1, 0.5x^2, |x| - 0.5$$

where x is the Manhattan distance between 2 vectors.

From the above equation the smooth L1 can be re-written:

$$sL1 = \begin{cases} |x| - 0.5, & if \ \ |x| > 1 \\ 0.5x^2, & else \ if \ \ |x| \leq 1 \end{cases}$$

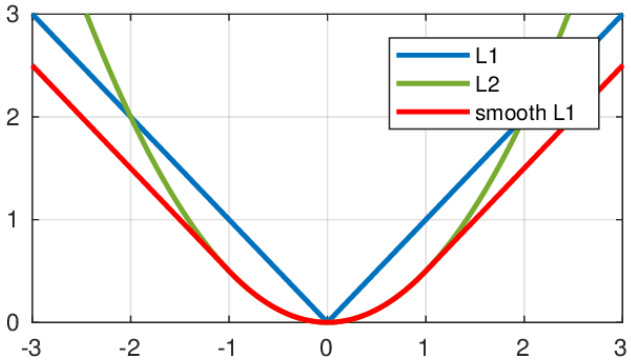

Image [14]: Smooth L1 Loss.

In a more general way, the smooth L1 loss can be re-written:

$$sL1 = \begin{cases} |x - y|, & if \ \ |x - y| > a \\ \frac{1}{|a|}(x - y)^2, & else \ if \ \ |x - y| \leq a \end{cases}$$

In other words, the goal of smooth L1 is to minimize the absolute difference between the target and the estimated value.

Focal Loss [3]. Focal loss is a modification of the cross entropy function, that reduces the contribution from easy examples and increases the importance of correcting misclassified examples.

From the Cross-Entropy function:

$$\boldsymbol{CE(p,y)} = \begin{cases} \boldsymbol{-\log(p)}, & \boldsymbol{y=1} \\ \boldsymbol{-\log(1-p)}, & \boldsymbol{otherwise} \end{cases} \quad (1)$$

Modifying the above loss function in more simplistic terms:

$$\boldsymbol{p_t} = \begin{cases} \boldsymbol{p}, & \boldsymbol{y=1} \\ \boldsymbol{1-p}, & \boldsymbol{otherwise} \end{cases} \quad (2)$$

By applying (2), in equation (1):

$$\boldsymbol{CE(p,y) = CE(p_t) = -\log(p_t)} \quad (\mathbf{3})$$

At this point, Cross-Entropy handles only the weight of positive and negative examples. Positive examples are the target class and negative examples are non-target class or background information. However, in object detection there are samples that were correctly classified as positive or negative example, and there are samples misclassified as negative or positive examples. Those are easy and hard positives/negatives respectively and Cross-Entropy does not handle them at all. Apart from that, usually dataset suffer from class imbalance, making the network biased towards the dominant class.

To solve those problems, Focal Loss adds a modulating factor to the cross-entropy loss, with a tunable hyper-parameter.

$$\boldsymbol{FL(p_t) = (1-p_t)^{\gamma} log(p_t)}$$

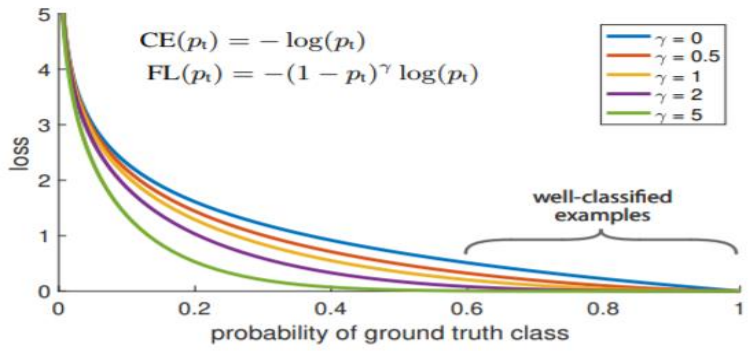

Image [15]: Focal Loss.

If the gamma parameter gets the value zero, then the focal loss becomes cross-entropy.

There are other loss functions, such as categorical cross-entropy, hinge loss, Huber loss, Mean Square Error and many more.

3. Well Established CNNs