DEEP LEARNING FOR COMPUTER VISSION WITH PYTHON

# **Basics of Machine Learning & Neural Networks**

**Volume 1: Starter Bundle**

Deep Learning for image classification

1. Machine Learning
2. Neural Networks
3. Convolutional Neural Networks
4. How to work with own datasets

**Volume 2: Practitioner Bundle**

To learn deep-learning in depth

**Volume 3: ImageNet Bundle**

Ways to train large scale neural networks on massive ImageNet dataset and few other topics like;

* Real world case studies
* Age + gender prediction
* Vehicle make + model identification
* Facial expression recognition

[**http://www.pyimagesearch.com/contact/**](http://www.pyimagesearch.com/contact/)(Link for different bundles)

**PYTHON, KERAS & MXNET**

1. Python: Best way to work with deep learning algorithms
2. Keras: Primary deep learning library. Maintained by **Francois Chollet** (Deep learning researcher and an engineer at Google). It’s a modular network library that can either use **Theano** or **TensorFlow** as backend.

It’s a deep learning framework that provides a well designed API to facilitate building deep neural networks with ease.

1. Mxnet: Second deep learning library (ImageNet bundle only). It’s a lightweight, portable and flexible deep learning library. It provides bindings to python programming language and specializes in distributed, multi-machine learning – training deep neural networks on massive datasets.

We’ll use Computer vision, image processing and ML Libraries like OpenCV, scikit-learn etc. (1,2 and 3 together build a powerful deep learning environment which can be used to learn DL for visual recognition)

**TENSERFLOW**

**TensorFlow and Theano are libraries used for defining abstract, general-purpose computation graphs. They are not deep-learning frameworks. These are computational backends. We can always integrate TensorFlow code directly into keras model.**

**OPEN CV**

**OpenCV is used mainly to facilitate basic image processing operations like loading an image from the disc, displaying it to screen etc.**

AGENDA OF THIS BOOK!

1. Load image datasets from disk, store them in memory, or write them to an optimized database format.
2. Preprocess images such that they are suitable for training a Convolutional Neural Network.
3. Create a blueprint class that can be used to build our own custom implementations of Convolutional Neural Networks
4. Implement popular CNN architectures by hand, such as AlexNet, VGGNet, GoogLeNet, ResNet, and SqueezeNet (and train them from scratch)

# **Deeper into Deep Learning**

DL methods are representational learning methods with multiple levels of representation, obtained by composing simple but non-linear modules that transform the representation at one level into higher level. These are not done by human engineers, rather they are learnt from data using general purpose learning procedure.

DL (belongs to family of ANN Algo)is a subfield of ML which is in-turn a subfield of AI.

Deep Learning

Machine Learning

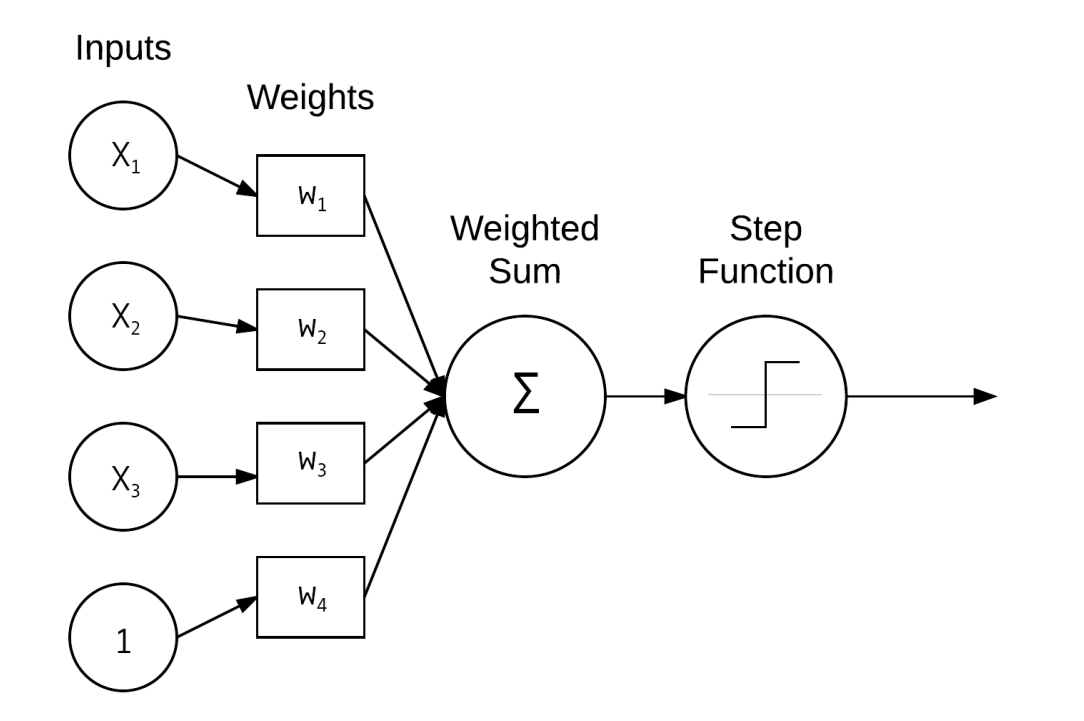
Artificial Intelligence

**Goal of AI**: to provide set of algorithm and technique that can be used to solve problem that humans perform intuitively and near automatically. Ex:- interpreting and understanding concept of image.

**Goal of ML:** specified to pattern recognition and learning from data. **Artificial Neural Networks(ANNs)** are class of ML algo that learn from data and specialize in pattern recognition(Inspired by structure and function of brain). “*Computers should learn from experience of problem they are trying to solve”*

DL has existed from 1940’s. Was named *cybernetics, connectionism, Artificial Neural Networks.* ANN model is inspired by human brain and how neurons interact with each other. Not a proper model of brain, it’s instead allows us to draw similarities between basic brain and how we can mimic some of it’s behaviour.

* **McCulloch and Pitts** (1943): 1ST Neural network model. It was a binary classifier, which was able to recognise 2 different categories based on some input. But this model had human intervention for adjusting weights, so it was not feasible.
* **Seminal Perceptron algo** (1950’s): published by **Rosenblatt** this model could automatically learn the weights required to classify the inputs hence deleting human intervention. The automatic training procedure formed the basic of **S**tochastic **G**radient **D**escent (SGD) which is used to train very deep neural networks today.
* **Publication by Minsky & Papert** (1969): stopped neural network research for 10years, they demonstrated that Perceptron with a linear activation function was a linear classifier and could not be used for non-linear problems.

(Example of Nonlinear problem)

* **Back Propagation algo**  and research by **Werbos**(1974), **rumelhart**(1986) and **LeCun**(1998) was able to restart(resuscitate) the studies on neural networks. It enabled “Multi-layered feed forward” neural network to be trained. They declared that neural networks are universal approximators which are capable of approximating any continuous functions. Modern day NN allowing us to efficiently train NN and teach them to **learn from its mistakes.** (due to slow computer, lack of large training sets the model then became infeasible).
* **Deep Learning:** Latest incarnation of NN. It’s different from previous models due to
* Faster speed
* Specialized hardware with more available training data

By this it is possible to train network with many more hidden layers. Hierarchical learning is possible where simple concept are learned in lower layer, abstract pattern are learnt in higher layer of network.

* **Convolutional Neural Network (CNN):** LeCun(1988) applied handwritten character recognition which learns to **filter** image by sequentially stacking layers on top of each other. Filters in low level of network represent edges and corner, at higher level these E&C are used learn more abstract concepts useful for filtering between image class.

**CNN** is considered as the most powerful image classifier and are currently responsible for pushing “state -of-the-art” forward in CV subfields that lever-age machine learning.

**HIERARCHICAL FEATURE LEARNING**

ML algorithm fall into 3 sectors:

1. Supervised
2. Unsupervised
3. Semi-supervised

**SUPERVISED CASE:** (similar to having a teacher watch you take a test)

In this case ML algo is given **both,** set of inputs and target outputs.Later the algo tries to learn patterns that can be used to automatically map input data points to their correct target output. Based on previously learnt knowledge, you try to answer. However, if you go wrong the teacher will educate you to give a more precise answer the next time.

**UNSUPERVISED CASE:**

ML algo try to automatically discover filter feature without any hint as to what the inputs are. Eg:- students learning without teacher’s guidance, they gather similar questions and find solution. This case is more challenging than supervised.

ML applied to image classification, goal of ML algo is to take these sets of images and identify pattern that can be used to filter out one image from another.

**Hand-Engineering Feature**: used to quantify the contents of an image. For each image in the data set we generally perform

1. **feature extraction** (process of taking input image, quantifying it according to some algo which is called **feature extractor** or **image descriptor**)
2. **return a vector:** list of numbers. These numbers aim to quantify the contents of an image.

METHOD-1: Hand Engineering features attempt to encode

* **texture:** Local binary pattern, Harlik texture
* **shape:** Hu Moments, Zernike Moments
* **color:** color moments, color histograms, color correlograms

METOHD-2: these describe the Salient(interesting) regions of the image

* + - Keypoint Detectors (FAST, Harris, DoG)
    - Local Invariant Descriptors: (SIFT, SURF, BRIEF, ORB)

METHOD-3: Histogram of Oriented Gradients (HOG)

This method is proved to be good at detecting objects in image. Ex:- detecting “STOP” board from an image, Here image may contain road, tree etc and a STOP board(object), the model will be trained to identify this object.

After the introduction of DL and CNN, hand-defining a set of rules and algo to extract features from an image have been discarded. Now **these features are automatically learnt from training process.**

Using DL we try to analyse the problem in a hierarchical manner, where each concept is built on top of other. Concept in Low level deal with basic representation of the problem, at higher level they use basic layers to form more abstract concepts. Hierarchical method completely removes hand-design feature extraction process and treat CNN as end-to-end.

**Major advantage of using DL and CNN is it skips FEATURE EXTRACTION and focuses on process on training the network to learn these filters.**

Given an image,

* Pixel intensity- input to CNN
* Series of **hidden layers** (Built upon each other in hierarchical manner) used to extract feature from input image.
* Initially edge region are detected in lower layer, these region are later used to define corners and outlines of object. Combining corner and outlines can lead to abstract “object part” in next layer.
* Output layer, used to classify image and object class label. It is directly or indirectly influenced by every other node in network.

**Traditional Feature Deep Learning**

**Extraction & ML**

Input Images Input Images

Hand Crafted Simple Features

Feature Extraction (edges)

Algorithm

Intermediate

ML Classifier features(corners)

Abstract Features

Output (object parts)

Output

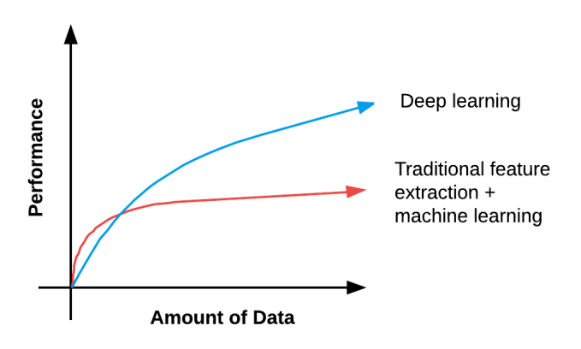
**“DEEP”** refers to number of layers. DL refers to large NN where layers built on top of each other gradually increasing the depth.

Any network with more than 2 hidden layers can be considered deep. Reasoning is as follows.

* Previously ANN’s were not feasible due to these reasons we could not train model with more than 2 hidden layers:
* Lack of large, labelled dataset for training
* Computers being too slow to train large NN
* Inadequate Activation functions
* In current times, due to enhancements in DL we now have
* Faster computers
* Highly optimised hardware
* Large, labelled datasets
* Understanding od weight initialization function

As the depth of network increases, even classification accuracy increases.

As the amount of data available to deep learning algorithms increases, accuracy does as well, substantially outperforming traditional feature extraction + machine learning approaches.



As the amount of training data increases, NN algo obtain high classification accuracy. Due to this relation we tend to associate DL with Large data sets.

**Rules to determine if given NN is deep:**

* If specialized network architecture like CNN, Recurrent NN, Long Short-Term Memory (LSTM) are used then we are performing DL.
* If network has depth > 2, DL is being performed
* If network has depth > 10, very DL

# **Image Fundamentals**

**PIXELS**

Pixels are the raw building block of images

Each image has set of pixels. Generally pixel is considered as a **color** or **intensity** of light that appears in a given place in our image.

Pixels are represented in 2 ways:

* Color pixels: are represented in RGB color space. These are not scalar values instead they have one Red component, one green and one blue.

Each RGB have values defined in the range [0,255] for a total of 256 shades where 0 is no representation 255 is full representation. (8bit unsigned int is used to represent the intensity).

* Grayscale/single channel: here each pixel is a scalar value between 0 (black) and 255 (white). Values between 0 to 255 are varying shades of Gray. Ones near to 0 are darker, ones near to 255 are lighter.

**Images as NumPy Arrays**

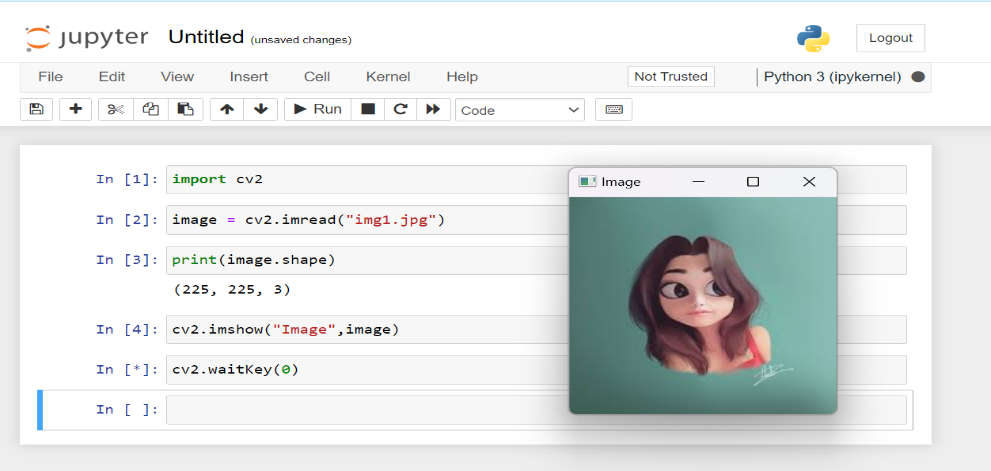
Image processing libraries like OpenCV, scikit-image represent RGB image as multi-dimensional NumPy array with shape **(height, weight, depth)** due to matrix notation, height comes before width in the array.

Dimension is always written as rows \* column.

Number of rows = height, number of columns = width.

<http://localhost:8888/notebooks/Untitled.ipynb> //Link for Juputer Notebook

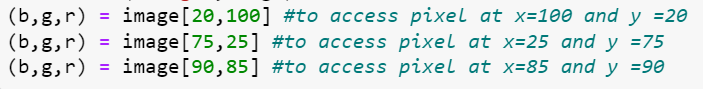
RESULT



The image has height – 255pixel //Num of row in array hence mentioned first

Width – 255pixel //Num of column in array hence mentioned second

Depth – 3pixel (number of channel)



y values are entered before x.

* import cv2 #Pre-built CPU-only OpenCV packages for python
* image = cv2.imread("img1.jpg"): imread() function in OpenCV is used to read image file from filesystem and load it into memory as a matrix representation.

Syntax**: image = cv2.imread(path, flags)**

**Path-** represent path to the image file

**Flags-** (optional) Integer flag that specifies how the image should be read.

**imread() –** returns NumPy represents the loaded image. And BGR color format is used.

* print(image.shape): after reading an image using imread() function in OpenCV, it will display the shape of loaded image in the **(height,width,depth)** format.
* cv2.imshow("Image",image): imshow() function is used to display an image in a window.

SYNTAX: cv2.imshow(window\_name, image)

**window\_name -** a string specifying the name of the window to display

**image –** the image to be displayed

* cv2.waitKey(0): function is used to wait for a keyboard event. It waits until a key is pressed and returns the key code of the pressed key. The argument 0 passed to waitKey() indicates that it will wait indefinitely until a key is pressed.

**RGB and BGR Ordering**

OpenCV stores RGB channels in reverse order. It stores in Blue | Green | Red

This was made because, early camera manufacturers and other s/w developers were using this

**Scaling and Aspect Ratio**

Scaling – resizing, process of increasing or decreasing the size of an image in terms of width and height.

While resizing image, aspect ratio needs to be considered. It is ratio of width to height.

While working with CNN applied to task of image classification assume a fixed input, i.e., dimension of all image we pass must be same.