People‘s Democratic Republic of Algeria
MINISTRY OF HIGH EDUCATION AND SCIENTIFIC RESEARCH

Ibn Khadoun university – Tiaret

Thesis

Introduced to :

Mathematics and Computer Science faculty
Computer Science department

For obtaining :
MASTER

Specialty: Computer Engineering

By:

Chedad Soumia
Kertel Chahrazad

Sur le thème

Recognition of house numbers by artificial neural networks

Publicly supported on .. / .. / 2021 in Tiaret before the jury composed of:

Mr BENDAOUD Mebarek	MCA	Tiaret University	President
Mr MEZZOUG Karim	MAA	Tiaret University	Encadreur
Mr BAGHDADI Mohamed	MCB	Tiaret University	Examiner

2020-2021

PEOPLES DEMOCRATIC AND REPUBLIC OF ALGERIA
MINISTRY OF HIGH EDUCATION AND SCIENTIFIC RESEARCH

University of Ibn Khaldoun Tiaret
faculty of mathematics and
computer science
department of computer science

Theme :

Recognition of house numbers by artificial neural networks

For obtaining Masters degree

Specialty: Software Engineering

Presented by:

Chedad Soumia
Kertel Chahrazad

Supervised by: Mr. Karim MEZZOUG MAA University of Ibn-Khaldoun-Tiaret

The jury is composed of :

Mr BENDAOUD Mebarek MCA University of Ibn-Khaldoun-Tiaret President .
Mr BAGHDADI Mohamed MCB University of Ibn-Khaldoun-Tiaret Examiner.

Academic year 2020-2021

Acknowledgments

First and foremost , I would like to thank Allah for giving me the will, health,patience and strength to complete that humble work.

Definitely ,I would like to thank my supervisor, **Professor Mezzoug Karim**, whose expertise was invaluable. his insightful feedback pushed me to sharpen my thinking and brought my work to a higher level As much as possible.

I would like to thank **Professor BENDAOUD Mebarek** and **Professor BAGHDADI Mohamed** for considering to evaluate my final year thesis.

In addition, I would like to thank my **parents Mohamed** and **Roqia** for their wise counsel and sympathetic ear.They provided me with the tools that I needed to choose the right direction and successfully complete my dissertation., They are always there for me. Finally, I could not have completed this dissertation without the support of My brothers and sisters, who provided happy distractions to rest my mind outside of my research.

Chedad Soumia

Acknowledgments

First of all, I would like to thank **Ms. Mayzouk Karim**, wholed this work. Receive my gratitude, sir, forYour valuable help and advice over the years.

I also extend my sincere thanks to **Professor BENDAOUD Mebarek** and **Professor BAGHDADI Mohamed**, who prefers to serve and do humble work The honor of the jury.

I thank my dear parents, as I owe them everything, their encouragement and Constant support for them, and I thank my dear brothers for their constant encouragement as well my lovely family.

Finally, I would like to express my sincere thanks and appreciation to you I salute the colleagues from Ibn Khaldun University.

Kertel Chahrazad

Abstract

Digital image processing is usually done using convolutional neural networks that reduce the size of images with increasing their depth .as it uses different designs of multilayer perceptron. with different architectures , including street view house number datasets.

This thesis we try to obtain similar results to the state of the art by appling some deep convolutional neural network to the problem of classifying single digits in street view images of house numbers datasets format 2 ,with also introduce three class(class 1,2 and 3) to the SVHN dataset background to aid in the problem of image classification, and are implemented in TensorFlow using Google Colaboratory that help us to minimize time of execution As much as possible.

Keywords: Digital image processing,convolutional neural networks CNNs , Multi-layer perceptron , Street view house number SVHN,.

Contents

List of Figures

List of Tables

Glossary of Acronyms and Abbreviations

AI	Artificial Intelligence
ML	Machine learning
DL	Deep learning
MLP	Multi-layer perceptrons
GD	Gradient Descent
SVM	Support Vector Machines
SGD	Stochastic Gradient Descent
CNN	Convolutional Neural Network
ANN	Artificial Neural Network
RNN	Recurrent Neural Network
GPU	Graphical processing unite
KNN	K-nearest Neighbors
ReLU	Rectified Linear Unit
SVHN	Street View House Number
Resnet	Residual Neural Network

General initiation

Recently, Convolutional neural network has been developed to mightily advanced the state of the art accuracies of object recognition and detection , also used successfully to make representations in computer vision , by extracting and learning good feature representations from unkown huge input data for high level tasks such as classification .

Reading digital numbers from photographs is a kind of a difficult computer vision problem that is important for a range of real world applications. For example, an application is the problem of recognizing house numbers posted on the fronts of build.

The problem of recognizing digit number in images has been extensively studied in the last few years, In our work , We try to obtain similar results to the state of the art by appling some deep convolutional neural network to the problem of classifying single digits in street view images of house numbers datasets.

1 Problematic

The objective of the thesis is to implement a several simple image classification models based on the convolutional neural network and a deep neural network , with different architecture and parameter to recognize the number in pictures not in the street view house number datasets . certainly, Images may be suffer from motion blur ,sizing issues, awkward angles,and becoming dirty. Also there are many types of number . For instance , it ables to be multiple digit classification, single digit classification, mixed classification (combine numbers and letters) ect.

2 Plans

Chapter N1 we have presented the fundamentals of image processing and some definitions and we have seen some methods which help in image processing.

Chapter N2 , we explain most important techniques in recurrent years , Airtificiel intiligence , machine learning and deep learning , By all means , we discut about Classification and neural network , and show varouis type ot it then take a look at some common cnn architecture.

Chapter N3 in this chapter we explain the state of the art that talks about some of the work done, as we also talked about a model that was completed by his experience and some of the results obtained.

Chapter N4 is will explain our architectures of the suggested mixed datasets models and describe the tools used in our experiments , with implement simple interface used by Django, and a discussion over the results of models.

Chapter I

Artificial vision

Introduction

Image processing is the operations that can be performed on images , Before processing the images, these images must first be produced ,it is the acquisition stage whitch consists in transforming the real views into digital images . After this step , these images must be transformed into matrices that can be processed subsequently.

1 What is Image Processing

Picture processing is a technique for applying operations on an image in order to improve it or extract relevant information from it. It's a sort of signal processing in which the input is an image and the output is either that image or its characteristics / features. Image processing is one of the most quickly evolving technology today. It is also a critical research field in engineering and computer science.

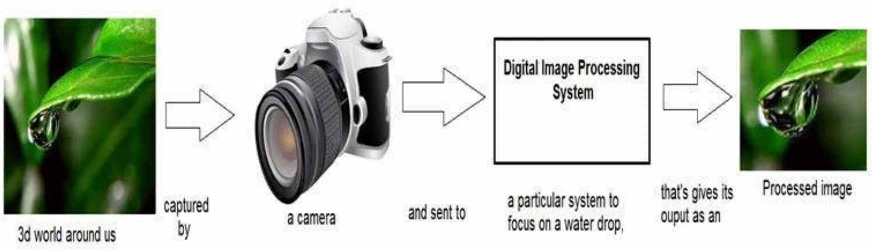

3d world around us
captured by
a camera
and sent to
Digital Image Processing System
a particular system to focus on a water drop,
that's gives its ouput as an
Processed image

Figure I.1: image proceesing

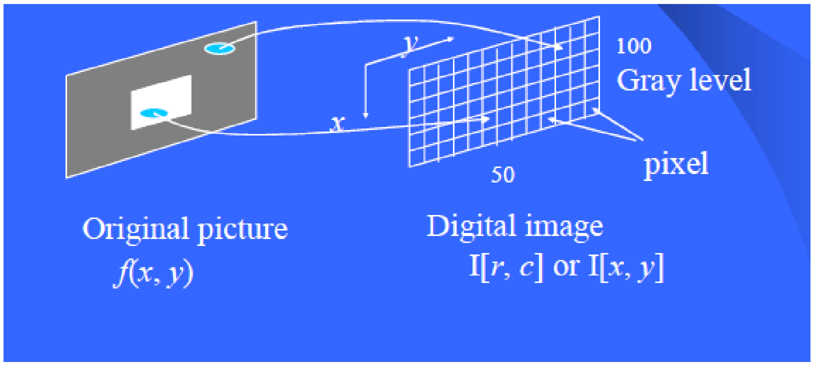


Figure I.2: digital image

1.1 What is the digital image?

1.1.1 Definition of image

A two-dimensional visual picture of a real-life thing (a person or any other object) is known as an image. A photograph is nothing more than a collection of pixels in several color spaces.

1.1.1.1 Numeration It is transforming images into digital images. The digital images when can transform into matrix .

1.1.1.2 Pixel is the term most widely used to denote the elements of a digital image We can read and convert it to matrix as in the figure.

1.1.1.3 Gray Level The grey level or grey value indicates the brightness of a pixel. These values are usally represent in 8-bit.So the range of values from 0 to 255 [00000000-11111111].the values near to 0 indicates darker regions and the values near 255 represent brighter regions.

1.1.1.4 Pixel and graylevel The pixel taking on a value is called the intensity.

$$I(i,j) = N \tag{I.1}$$

where N :gray level value $N \in (N_{min}, N_{max})$

$N \in (N_{min}, N_{max})$ = gray level value.

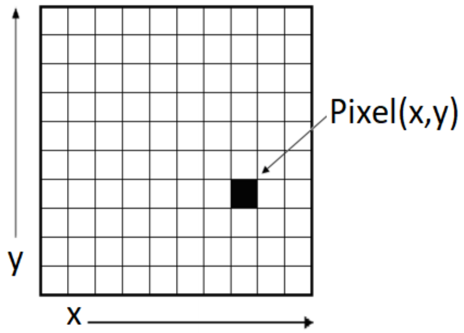


Figure I.3: Digital image

$\log_2(N \in (N_{min}, N_{max})) = \text{dynamic}$.

1.1.1.5 Sampling and Quantification In order to become suitable for digital processing, an image function f(x,y) must be digitized both spatially and in amplitude. Typically, a frame grabber or digitizer is used to sample and quantize the analogue video signal. Hence in order to create an image which is digital, we need to covert continuous data into digital form. There are two steps in which it is done:

Sampling The process of spatial discretization of an image consisting in associating with each rectangular zone R (x, y) of a contune image a single value I (x, y).

Quantification denotes the limitation of the number of different values that can take I (x, y).

1.1.2 Statistical model

1.1.2.1 Histogram The histogram of an image measures the distribution of gray levels in the image. For a gray level x, the histogram allows you to know the probability of finding a pixel of value x by drawing a pixel at random in the image.

Normalization of the histogram Calculate the function:

$$I'(x,y) = \frac{(I(x,y) - Min(I))255}{Max(I) - Min(I)} \tag{I.2}$$

Histogram Equalization The steps to calculate the equalize :

- Calculation of the histogram h(k) with k ∈ 0,255

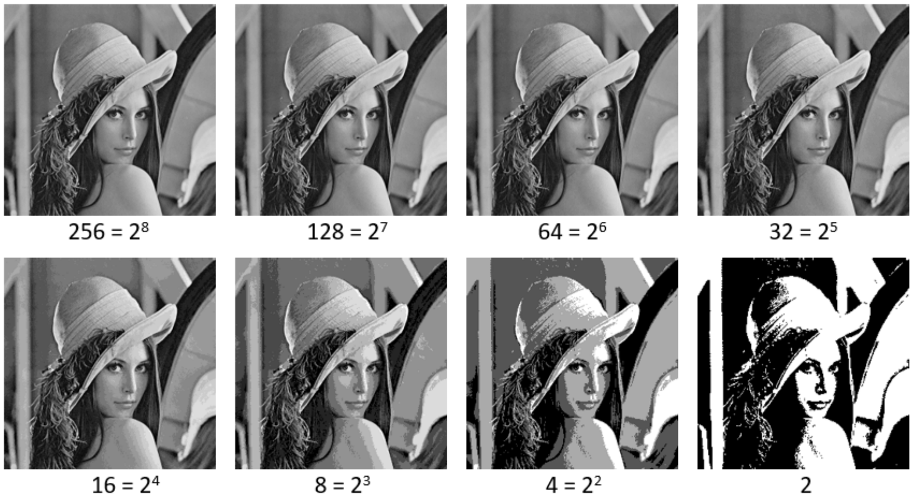


Figure I.4: : Image diffrents de niveau de quantification (nombre de niveau de gris indiqu sous chaque figure).

A digital image is a sampled and quantified image

- Normalization of the histogram

$$h_n(k) = \frac{h(k)}{N} \tag{I.3}$$

  where k $\in$ 0 ,255

- cumulative probability density

$$C(k) = \sum_{i=0}^{i} h_n(i) \tag{I.4}$$

1.1.3 color image processing

Color in imag e processing is motivated by two factors:

- Color is a power ful descriptor that often simplifies object identifications and extractionfrom the scene.
- Human can discern thousands of color shades and intensities,compared to about only two dozen shades of gray.

1.1.4 Type of color

1.1.4.1 binary image (B&W) : Black and white contains only two levels (0,1) 1Bit per pixel.

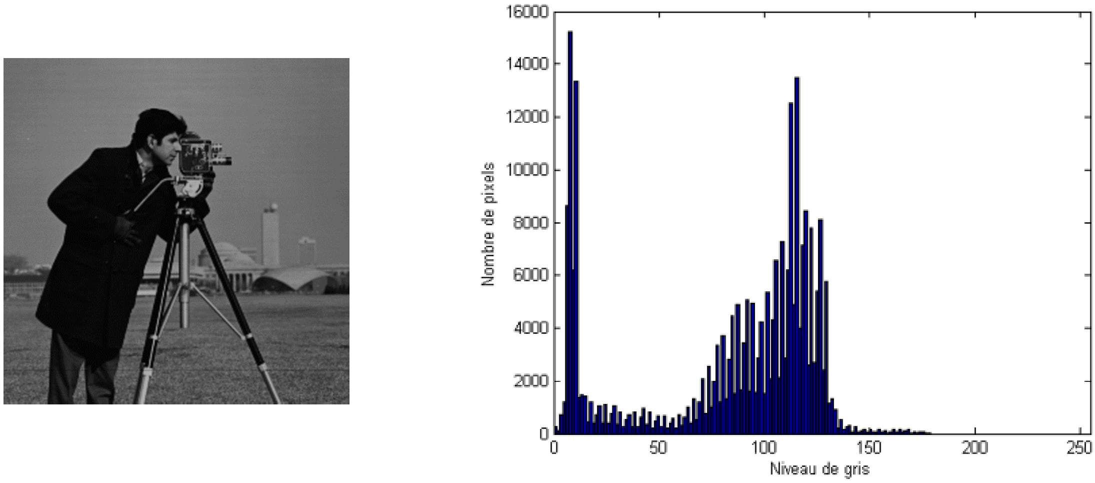


Figure I.5: histogram of image

1.1.4.2 Gray scale image

Gray image represent by black and white shades of gray without apparent color or combination of levels for exemple :(8-bit) gray image means total (2^8). Levels from black to white 0=black and 255=white .

1.1.4.3 RGB Image(red,Green,and Blue)

RGB (red, green and blue) refers to a system for reproducing colors for use in a computer screen, red, green and blue can be mutual in various proportions to achieve any color in the spectrum visible. For each color channel: red (R), green (G) and blue (B), each element of the matrix contains an 8-bit value, returns to the amount of red, green or blue

1.1.4.4 RGBA Image (Red ,Green,and Blue ,+ Alpha or Opacity)

RGBA stands for Red ,Green,and Blue Alpha .Here A in this prooerty name stands for Alpha .Alpha indicates how opaque each pixel is and allows an image to be combined over others using alpha compositing ,with transparent areas and anti-aliasing of the edges of at that point a [0-255] scale

1.2 Digital image Fundamentals

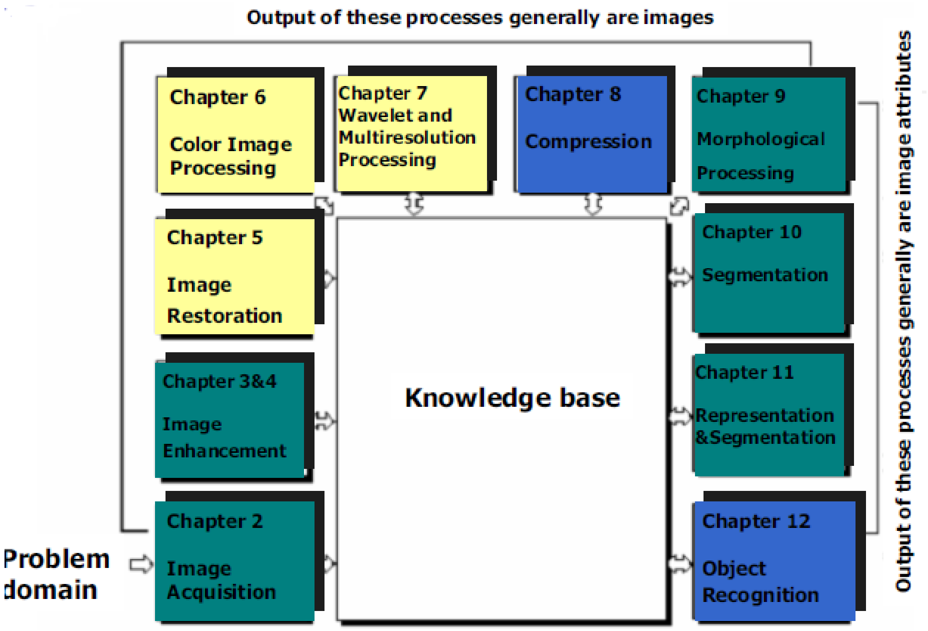


Figure I.6: fundamental steep

There are different algorithms for different tasks and problems, and we often want to distinguish the nature of these algorithms. The different image processing algorithms fall into broad sub-categories.

1.2.1 image acquisition

An image is captured by a sensor (such as a monochrome or color TV camera) and digitized. If the output of this camera or sensor is not already in digital form, an analog-digital converter digitizes it.

1.2.2 image Enchanement

This refers to the processing of an image so that the result is more suitable for a particular application. To highlight certain characteristics of interest in an image. Activating an image provides better contrast and a more detailed image compared to an unimproved image. Examples include:

- Emphasizing or eliminating the blur of a blurred image
- Highlighting edges
- Improving the contrast of the image or brightening an image
- Noise suppression

Enchanting techniques are divided into 2 types:

1. Spatial domain: direct manipulation of pixels in the image plane.
2. Frequency domain: modification of the frequency spectrum of the image.

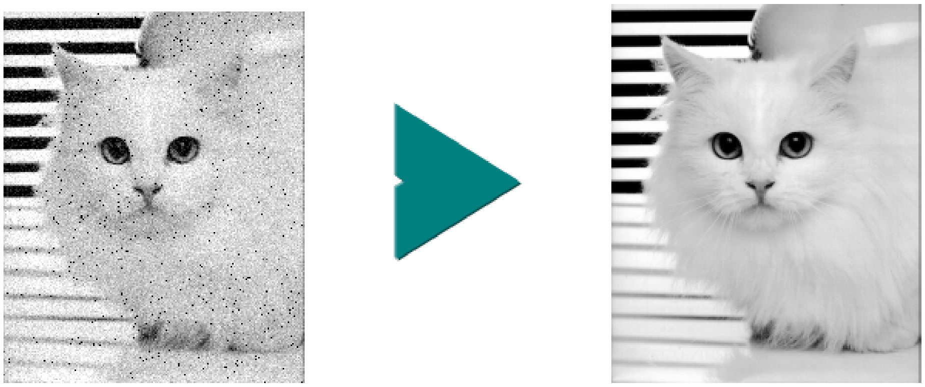


Figure I.7: Image enchanemnet

1.2.3 Image restoration

This may be considered as reversing the damage done to an image by a known cause, for example:

- Removing of blur caused by linear motion
- Removal of optical distortions
- Removing periodic interference

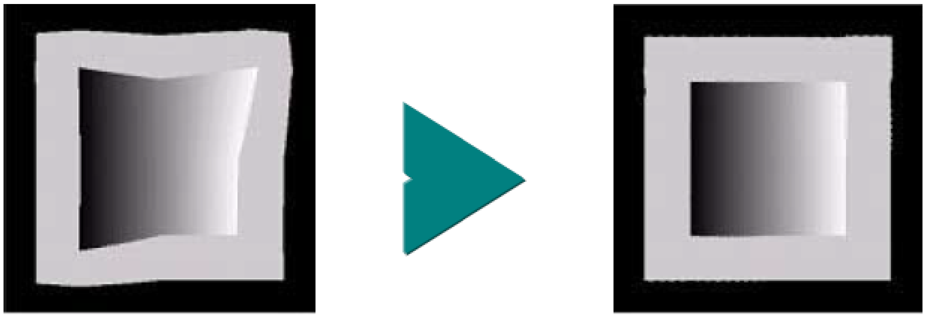


Figure I.8: Image restoration

1.2.4 Morphological processing

Tools for extracting image components that are useful in the representation and description of shape.

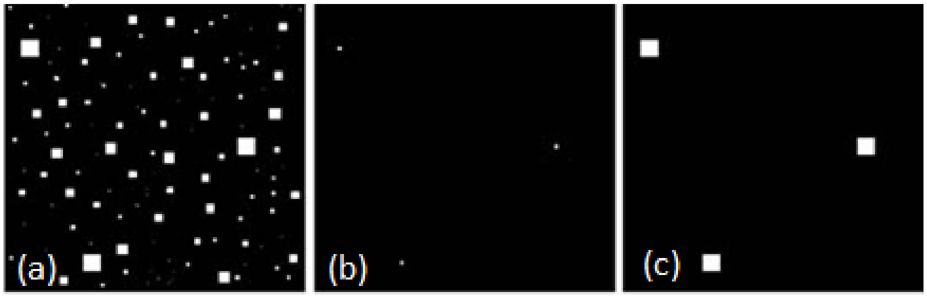


Figure I.9: (a) Image of sqares of size 1.3.5.7.9 and 15 pixels on the side.(b) Erosion of (a) with square structuring element of 1's.13 pixels on th side.(c) Dilation of (b) with the same structuring element.

1.2.5 Image segmentation

This entails breaking down a picture into its component elements or isolating specific aspects of an image:

- Recognizing lines, circles, or certain shapes in a picture.
- Recognizing automobiles, trees, buildings, or roadways in an aerial image.

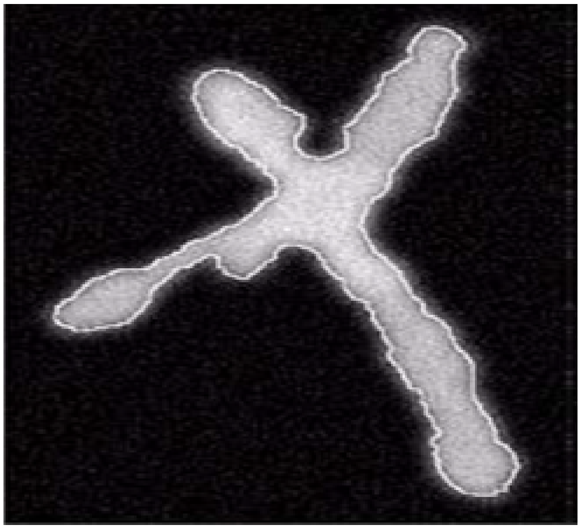


Figure I.10: Image segmentation

1.2.6 Representation & description

Representation make a decision whether the data should be represented as a border or as a complete region.

- Representation of limits focus on shape features, such as corners and inflections.
- Representation of the region focus on properties, such as texture or skeleton shape.

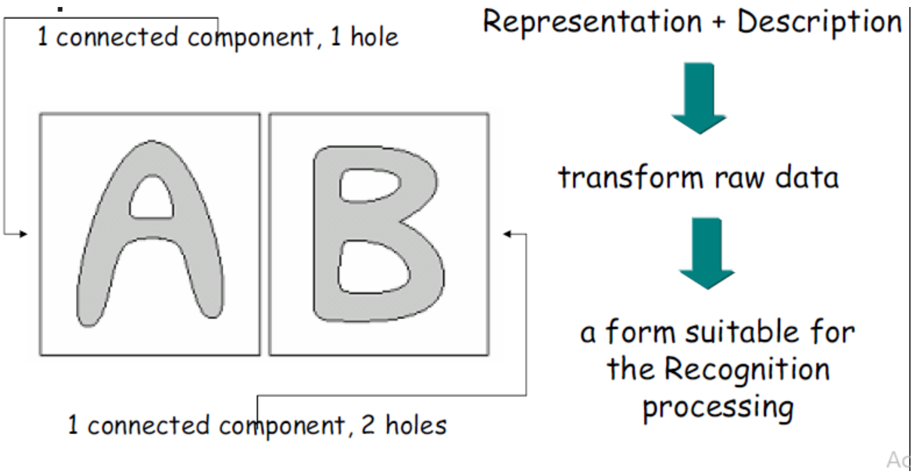

1 connected component, 1 hole
Representation + Description
transform raw data
a form suitable for the Recognition processing
1 connected component, 2 holes

Figure I.11: Representation & description

1.2.7 Object recognition

Recognition is the process that awards a label to an object based on information provided by its descriptors.

1.2.8 Image compression

Reduced the amount of storage or bandwidth needed to record and transmit an image. The image compression standard JPEG (Joint Photographic Experts Group) is an example.

1.2.9 Color image processing

Get in importance due to the significant increase in digital use images on the Internet.

1.2.10 Image registration

This entails matching different photos in order to compare them, or processed at the same time To share the same coordinate system, all of the initial images must be connected together. Although registration as a whole is not described in this article, several activities that are critical to registration, such as corner detection.

These classes are not mutually exclusive; a single algorithm can be used for both picture and text processing.for image enhancement or restoration We should, however, be entitled to choose what it is.is what we're attempting to achieve with our image:merely improve the appearance (enhancement)or repairing the harm (restoration).

1.3 Filter

1.3.1 definition of filter

Is ,decrease the change amount of color /color intensity between pixel and the Neighborhood .usually used to remove the effects of bugs in the image .

The most interested methods for image processing that related to :

- Remove noises from the image
- Correct the image

1.3.1.1 Spatial frequencies Numbers of changes in the brightness values per unit distance for any particular part of the image. Image composed of :

- Low frequency details: few changes in brightness valueover a given area.
- High frequency details: brightness values change dramatically over short distances.

1.3.1.2 Spatial filter is an image processing technique for changing the intensities of a pixel according to the intensitier of the neughboring pixels . g(x,y)=T[f(x,y)]

1.3.1.3 Filter type

Low-pass filters

- obstruct high-frequency information.
- Gives photos a smoothing appearance.
- It's a noise-cancelling device.

- The salt and pepper sounds has been removed.
- o Image blurring, especially around the corners the others.

High pass filters

- Preserves high frequencies while removing components that change slowly.
- Pays special attention to small things.
- Detection and enhancement of edges.
- Transitions - points where one category gives way to another Occurs.

To do this ,there are many types of filters, like:

- Mean filter
- Median filter
- Gaussian filter

Mean filter

- Used to remove large bugs(noise) in the image
- The idea of this filter is :
  - Change the color value for each pixel by calculate the arithmetic mean(average) for this pixel and neighborhood in the form of matrix.

Disavantages:

- Reducting the severity of contrast.

Median filter

- Median filter used as alternative for Mean filter
- Used to remove dot noises with littlie space :
  - That cover one or two pixels.
  - Name it (salt and pepper)

Gaussian filter

- This technique give a greater weight for the selected pixel, then to nearest the center ,and so on.
- Use to:
  - Remove unregularly bug in the image
  - In calculation gives a greatest value to selected pixel

2 Computer Vision

2.1 historique

In recent years, the largest international companies (Google, Facebook, Amazon, Apple) have invested heavily in deep learning and computer vision. In the automotive sector, the autonomous vehicle manufacturer Tesla has focused for several years on computer vision, more than on IoT. The premise that justifies this price position: connected cameras can process information in real time, offering greater reliability than various electronic sensors. In the energy sector, Suez uses computer vision in water and waste, in particular to detect objects that are not intended to enter the incinerator. Another example in the industry, where the start-up Prophesee intends to use images to ensure predictive maintenance. In addition, with the coronavirus crisis, security solutions specialist Dahua Technology readjusted its cameras to detect people with fever by computer vision.

2.2 what is computer vision? (IBM)

Computer vision is an artificial intelligence (AI) area that allows computers and systems to extract useful information from digital pictures, movies, and other media. Using visual inputs, perform actions or create recommendations depending on the data. If artificial intelligence allows computers to think, computer vision allows them to see, watch, and learn. Computer vision operates similarly to human vision, with the exception that humans have a retina.get a head start Human vision has the benefit of lifetimes of context to teach how to detect the difference. objects apart, how far apart they are, if they are moving, and if there is a gap between them In a photograph, there is a flaw.

2.3 How does computer vision work?

Computer vision necessitates a large amount of data. It goes through the data analysis process again and again until it recognizes and distinguishes between photos. To teach a computer to recognize automotive tires, for example, it must be fed a large number of tire photos and tire-related materials in order for it to understand the differences and recognize a tire, particularly one with no faults.

Deep learning, which is a type of machine learning, and a convolutional neural network are the two main technologies used (CNN).

Machine learning is a technique that allows a computer to train itself about the context of visual input using algorithmic models. If enough data is supplied into the model, the computer will learn to distinguish between images by ”looking” at the data. Instead of someone training the machine to recognize an image, algorithms allow it to learn on its own.

By breaking images down into pixels and assigning tags or labels, a CNN aids a machine learning or deep learning model's ”look.” It creates predictions about what it is ”seeing” by using the labels to do convolutions (a mathematical operation on two functions to produce a third function). In a series of iterations, the neural network executes convolutions and assesses the accuracy of its predictions until the predictions start to come true. It then recognizes or sees images in a human-like manner.

A CNN, like a human recognizing a picture from a distance, detects hard edges and simple forms first, then fills in the details as it runs iterations of its predictions. To comprehend single images, a CNN is employed. In video applications, a recurrent neural network (RNN) is used in a similar way to help computers grasp how visuals in a sequence of frames are related to each other.

2.4 Image processing Vs Computer Vision

Between six and seven million cone cells make up a human eye, each of which contains one of three color-sensitive proteins known as opsins.When photons of light strike these opsins, they change form, starting a chain reaction that results in electrical impulses, which are then sent to the brain for interpretation. This is a highly intricate process, and creating a machine that can comprehend it at a human level has always been difficult. The aim behind today's machine vision systems is to emulate human vision for pattern recognition, face recognition, and converting 2D pictures from a 3D world into 3D. On a conceptual and linguistic level, there is a lot of overlap between image processing and computer vision.Here we present some differences :

Image processing	Computer vision
- Image processing focus on processing images. - Input and the output are both images. - Transform images in many ways : smoothing ,sharpening, contrest,highlighting the edges ,and so on.	- Computer vision focuses on marking sense of what a machine sees. - Inputs an image and outputs task-specific knowledge ,such as object labels.

Table I.1: Image processing Vs Computer Vision

Conclusion

Image processing is a subfield of computer vision .The image processing techniques of a computer vision system are used to try to emulate human vision at a large scale. Picture processing, for example, might be used to improve an image for later use. And it's called computer vision if the aim is to recognize things, such as defects for automated driving.

Chapter II

Inteligence artificial

Introduction

In todays world , technology is quickly growing ,and we are getting in touch with various brand new technologies day by day, one of the emerging technologies of computer science is Artificial Intelligence which has grown to be too popular that can provide humans a great relief from doing various repetitive tasks,it is currently working with different subfields,and You can find it used in a great many applications, so, What is artificial intelligence [4]? and what are its most famous subsets ? there are infinitely questions of that technology.All these questions are meaningful when trying to understand artificial intelligence.

In this chapter we will try to explain some answers to common questions about that technology.

1 What is Artificial Intelligence ?

Artificial Intelligence , You probably use it dozens of times a day without even knowing it. Each time you do a web search on Google or Youtube, that works so well because their Artificial Intelligence software has figured out how to rank what pages. When Facebook or Apple's photo application recognizes and detection your friends in posted pictures, that's also Artificial Intelligence . whenever you read your email and a spam filter saves you from having to wade through tons of spam, again, that's because your computer has learned to distinguish spam from non-spam email. that's Artificial Intelligence.

Artificial Intelligence , as known as AI , is a board field of computer science where we try to make machines imitate human behavior , such as reasoning, learning, and problem solving ,that goal of it to develop them that behave as though they were intelligent.

Artificial Intelligence has significantly evolved over the past few years and we have found its applications in almost deferent business sector. such as Speech recognition , Machine translation systems,Robotic vehicles,Autonomous planning and scheduling,Game playing,Spam filtering,Vision Systems ,Robotics and more .

Artificial Intelligence consists of many Subsets whose common goal is to make a decision based on data and provide output. like Machine learning, Deep learning, Natural language programming and Virtual assistance like chatbots. and in this chapter, We will discuss two main important Subsets of artificial intelligence : the first one is the ML (machine learning), and the second is the concept of DL (Deep learning).

2 Machine learning

Machine learning is seen as a branch of artificial intelligence (AI) that gives systems the capability to automatically learn and improve their self through experience without being explicitly programmed. The process of learning begins with observations or feeding data and information,Machine learning focuses on the development of computer systems that can access data and use it to learn for themselves.

Growing sizes and varieties of accessible data, computational processing that is cheap and more powerful, and affordable data storage. all of these things are the reason of resurging interest in machine learning and Artificial Intelligence ,That mean it's possible to speedily and automatically produce models that can analyze bigger, more complex data and deliver faster, more accurate results even on a extremely big scale. And by making precise models.

Before we get into types of Machine learning , we have to look at the three basic concept of Machine Learning technique ,Regression , Classification and Clustering.

2.1 Regression vs Classification vs Clustering

2.1.1 Regression

Regression is one of the common well known and well understood algorithms in machine learning,it is a process of finding the correlations between dependent and independent variables. It helps in predicting the continuous variables, for instance, a model that assumes a linear relationship between the independent input variables (x) and the single dependent outcome variable (y). most specifically, that y can be calculated from a linear series of the input variables (x) , Regression algorithms are used to **predict the continuous values** for example ,predicting a price for your house and the amount you can sell it for.

When there is an independent only input variable (x), the method is referred to a simple regression. When there are independent multiple input variables, literature from statistics often refers to the method a multiple regression.

2.1.2 Clustering

Clustering consists of grouping similar instances together into clusters and, this is a great tool for recommender systems,search engines, image segmentation ,data analysis, customer segmentation , semi-supervised learning, dimensionality reduction,unsupervised learning , and more. for instance finding group of customers with similar behavior given a large database of customer data containing their demographics and data bying records

2.1.3 Classification

Classification algorithms are used to **predict/Classify the discrete values**, The best example to understand the Classification problem is The spam filte. The model is trained with many example emails along with their class (spam or not), and whenever it receives a new email, it identifies whether the email is spam or not. If the email is spam, then it is moved to the Spam folder.
There are multiple types of classifications like binary classification, multi-class classification

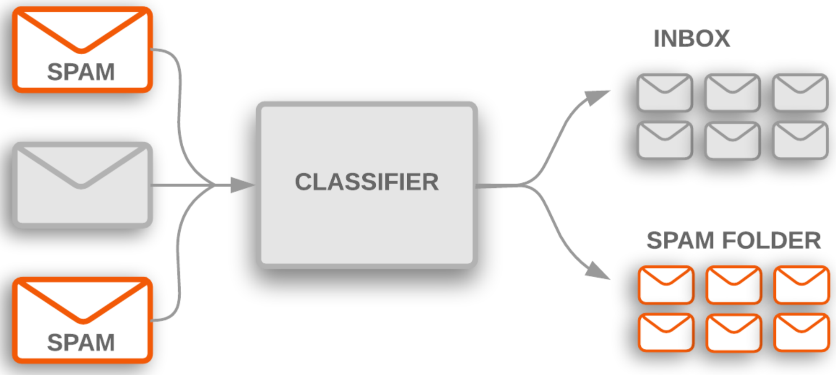


Figure II.1: Spam Detection [1]

2.1.3.1 Binary classification Perhaps Binary classification is the most widely applied type of machine-learning problem, is the special case of distinguishing between exactly two classes[5], the output of your model should be a scalar between 0 and 1.

2.1.3.2 Multi-class classification We use binary classifiers to distinguish between any two classes[6],and multiclass classifiers or multinomial classifiers can distinguish between more than two classes[7], for instance, classifying handwritten digits[8].

[1] This Picture was taken from (`https://medium.com/@naveeen.kumar.k/naive-bayes-spam-detection-7d087cc96d9d`) accessed 02 June 2021.

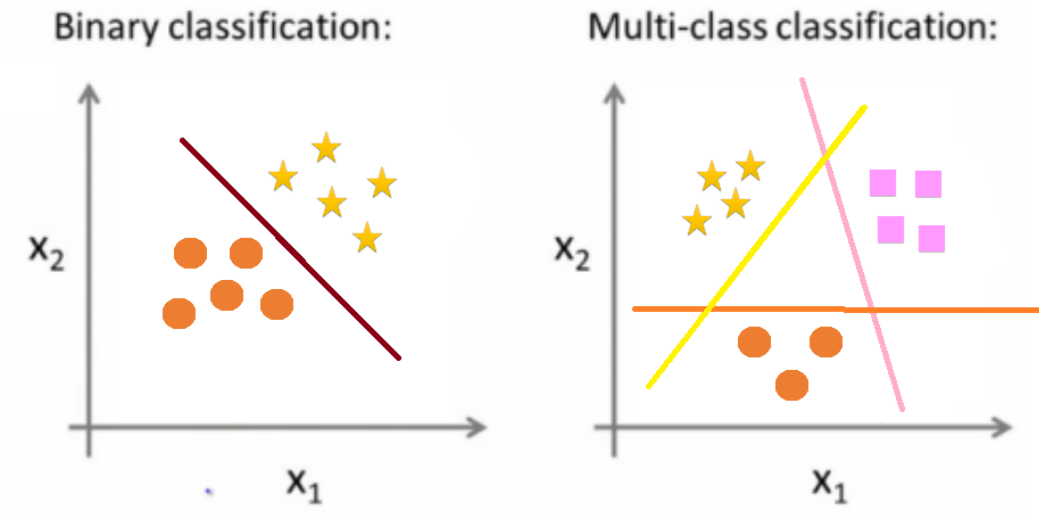


Figure II.2: Binary classification vs. Multi-class classification

2.1.4 Classification vs Clustering

Both Classification and Clustering are two effective machine learning techniques used for the categorisation of objects into one or more categories based on the features. They seem to be a similar process as the basic. The table II.1 shows us difference between Classification and Clustering

parameter for comparison	clasification	Clustering
Involved in	Supervised learning	Unsupervised learning
Basic	This model function classifies the data into one of classes already defined labels.	This function maps the data into one of the multiple class where the arrangement of data items is relies on the similarities between them.
Training sample	Labeled data is provided.	Unlabeled data is provided.
Examples Algorithms	Support Vector Machines , Logistic Regression , Nave Bayes ect.	k-means clustering , Fuzzy c-means clustering ect.

Table II.1: Difference between Classification and Clustering

2.1.5 Image classification

Image Classification is a supervised and unsupervised learning problem which is a fundamental task that tries to comprehend an entire image as a whole. The aim is to classify various classes of images(define a set of target classes), by assigning it to a specific label. For instance , you may train a model to recognize photos representing three different types of animals: Cats, bears, and dogs , you must to make 3 classes of a ton of images of them and train your model to recognize them.

2.2 type of Machine learning problems

There is simply not enough room in this thesis to cover all the machine learning problems! Instead, its better to focus on the most common ,Supervised learning ,unsupervised learning and rieforcement learning .

2.2.1 Supervised learning

Supervised learning is one of the most commonly used and successful types of machine learning. In this type, the machine learning algorithm is trained on labeled data, look at the picture II.3,it means it is a process of providing input data as well as correct output data to the machine learning model, the spam filter is a good example of this learning.
In another hand, Supervised learning is where you have input variable (x) and output variable (Y) and you use an algorithm to learn the mapping function from the input to the output based on example input-output pairs, as equation $Y = f(x)$.

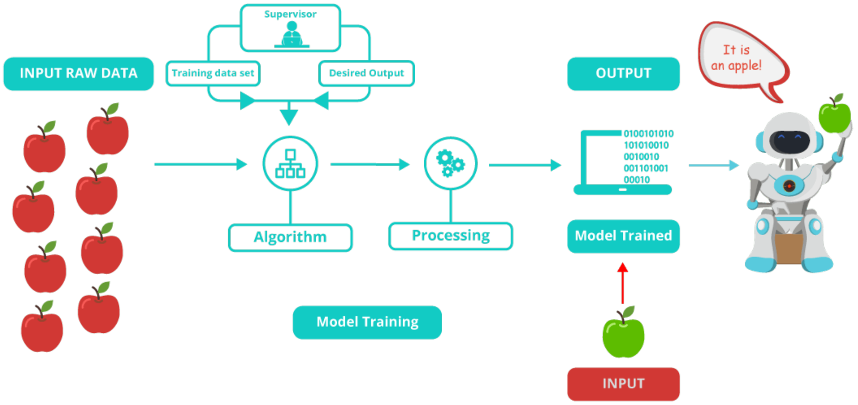


Figure II.3: Supervised learning [2]

[2] This Picture was taken from (https://www.edureka.co/blog/machine-learning-tutorial/) accessed 26 May 2021.

2.2.1.1 Common Supervised Learning Algorithms

Support Vector Machine (SVM) is currently the most popular approach for supervised learning algorithms , it can be used for both classification or regression problems, However, it is mostly used in classification problems.it use a hyperplane[3] to separate the feature from one dimension to high dimensional space, this concept is best when all the features are tightly bounded to work correctly.

k- Nearest Neighbors is a lazy algorithm that can be used for Regression and Classification but mostly it is used for the Classification problems.it stores all instances corresponding to training data in n-dimentional space , and at the time of classification, it performs an action on the dataset.

Decision tree algorithm builds the classification or regression model in the form of a tree structure. Decision trees are flowchart-like structures that lets you classify input data points or predict output values given inputs[8].

2.2.2 Unsupervised learning

Unsupervised learning is the training of a model using dqtq that is neither classified nor labelled. this model can be used to cluster the input data in categories on the basis of their statistical properties .

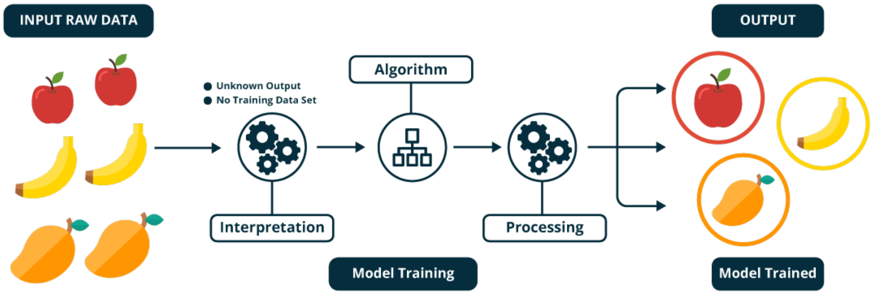


Figure II.4: Unsupervised learning [4]

[3] In an N-dimensional space. Hyperplanes are decision boundaries that aid in classifying the data(features) points.

[4] This Picture was taken from (`https://www.edureka.co/blog/machine-learning-tutorial/`) accessed 26 May 2021.

2.2.2.1 Common unsupervised Learning Algorithms

K-Means Clustering is used to solve the clustering problems in machine learning or data science.It allows us to cluster the data into various groups and a great way to discover the categories of groups in the unlabeled dataset on its own without the need for any training.

Fuzzy C-means is a method of clustering which works by assigning to each data point corresponding to each class center on the basis of distance between the cluster center and the data point.

2.2.3 Reinforcement learning

In simplest form, Reinforcement learning is learning by interacting with a space or an environment, For instance , An RL agent learns from the consequences of its action, rather than from being taught explicitly. it selects its actions on basis of its past experience(exploitation) and also by new choices(exploration).

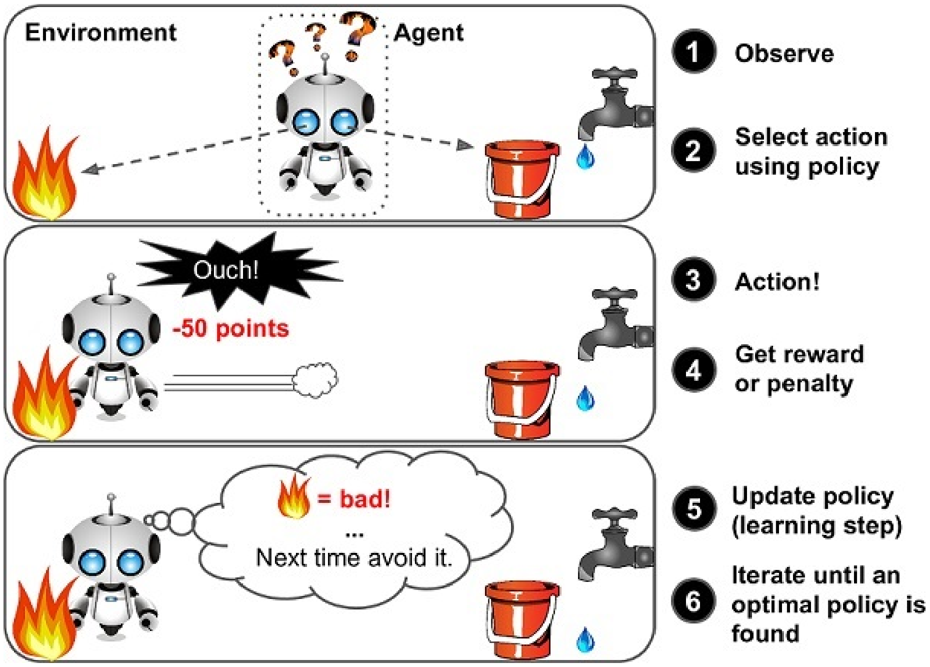

Environment
Agent
1 Observe
2 Select action using policy
Ouch!
-50 points
3 Action!
4 Get reward or penalty
= bad!
...
Next time avoid it.
5 Update policy (learning step)
6 Iterate until an optimal policy is found

Figure II.5: Reinforcement learning [5]

as shown in figureII.5 the learning agent gaine knowledge from observe the environment , the action give him a more maximum reward or give him negative

[5] This Picture was taken from (`https://marutitech.com/businesses-reinforcement-learning/`) accessed 26 May 2021.

reward. if the action correct or net ,This process of performing an action and learning lets him to learn from himself.

2.3 Limitation of machine learning

Machine learning algorithms are not useful while with high dimensional data The second major challenges with traditional Machine learning models is a process called feature extraction, For complex problems such as object recognition or handwriting recognition, this is a huge challenge. Deep learning models are capable to focus on the right features by themselves, these models can also partially solve the dimensionality problem. deep learning also can skip the manual steps of extracting features, you can directly feed images to the dl algorithm, which then predicts the object.

3 Deep learning

Humans and computers are inherently suited to different types of tasks[9]. for instance , The human is able to recognise a familiar object speedy, but it is extremely difficult for computer.It is only in recent years that deep learning has shown an accuracy on some of these tasks that exceeds that of a human[9].

Deep learning is an ensemble of statistical machine learning techniques used to learn feature hierarchies based on the concept of artificial neural network (ANNs) .

4 Artificial neural networks

An artificial neural network or perceptron is a model of reasoning which are inspired by biological neural networks,is a linear model used for binary classification. It models a neuron which has a collection of inputs, each of which is given a specific weight. The neuron computes some function on these weighted inputs and given the output. Through research of intelligent systems we can try to understand how the human brain works and then model or simulate it on the computer[4]. Many ideas and principles in the field of Artificial neural networks.

Artificial neural network is computational model which is inspired by biological neural network ,so let's take a look how biological neuron works

4.1 The biological neuron

A biological neuron network (represented in Figure II.6) consists of a number of very simple and highly interconnected processors[10], also called **neurons**,Most **neurons** receive many input signals throughout their **dendritic** ,and Each biological neuron is connected to several thousands of other neurons ,The processing of this information happens in **soma** or **cell body** , Towards its end, the **axon** is separated into many branches and develops bulbous swellings known as **axon terminals** (or **nerve terminals**). These **axon** terminals make connections on target cells.

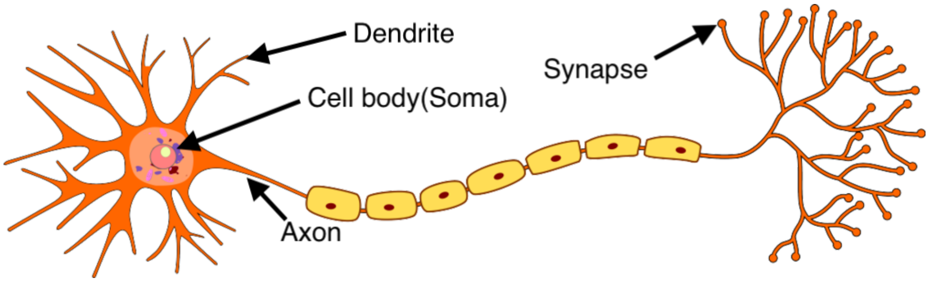


Figure II.6: Biological neuron [6]

4.1.0.1 Soma It is the body of a biological neuron[10]

4.1.0.2 Dendrites A branch of a biological neuron that transfers information from one part of a cell to another. Dendrites typically serve an input function for the cell, although many dendrites also have output functions[10]

4.1.0.3 Axon is responsible for transmitting signals to other neurons and may therefore be considered the neuron output[11].

4.2 The Perceptron model

The perceptron is a very simple classification algorithm. It consists of a single node or neuron that takes a row of data as input and predicts a class label. This is achieved by calculating the weighted sum of the inputs and a bias. The weighted sum of the input of the model is called the activation. As shown in figure II.7, each node gets many inputs. It sums up all the weights expressed as the following equation:

$$z(x,w) = \sum_{j=1}^{n} w_j x_j \tag{II.1}$$

Taking into account that the signal will always move from left to right. After sum , the result of sum passes it on as output, via a activation function as the following equation:

$$y_p = a(z(x,w) + \beta) \tag{II.2}$$

[6] This Picture was taken from (`https://smhatre59.medium.com/what-is-the-relation-between-artificial-and-biological-neuron-18b05831036`) accessed 7 June 2021.

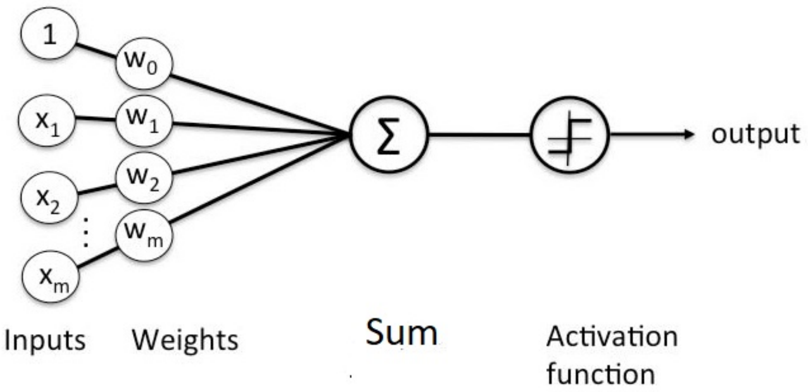


Figure II.7: perceptron mathematical model

Once all the nodes have followed the procedure, the final output will be given.Heres what the equation of a node looks like.

$$y_p = a(\sum_{j=1}^{n} w_j x_j + \beta) \tag{II.3}$$

4.2.1 Activation function

Activation function is the most crucial part of any neural network, it decides whether a neuron should be activated or not by calculating sum and further adding bias[7].The purpose of the activation function is to introduce non-linearity into the output of a neuron.

there can be many activation functions like :

4.2.1.1 sigmoid or Logistic The **sigmoid function** transforms the input, which can have any value between plus and minus infinity, into a reasonable value in the range between[10] 0 and 1 .

$$\sigma = \frac{1}{1+e^{-z}} \tag{II.4}$$

4.2.1.2 Tanh just like the sigmoid function, **Tanh function** is also used to predict or to differentiate between two classes but it plots the negative input into negative quantity only and its output value ranges in from -1 to 1.

$$\tanh = \frac{1}{1+e^{-z}} \tag{II.5}$$

[7] A bias is a learned offset , The bias can be seen as adding a threshold to the difficulty of activating the perceptron

4.2.1.3 Rectified Linear unit (ReLU) is a piecewise linear function , it is the most popular used activation function in artificiel neural networks, especially in CNN , it is defined as II.6

$$f(x) = \max(0, x) \tag{II.6}$$

4.2.1.4 softmax **Softmax** is used mainly at the last layer i.e output layer for decision making the same as sigmoid activation works, the **softmax** basically gives value to the input variable according to their weight and the sum of these weights is eventually one.

$$f(x_j) = \frac{\exp(x_i)}{\sum_{j=1}^{k} x_j} \tag{II.7}$$

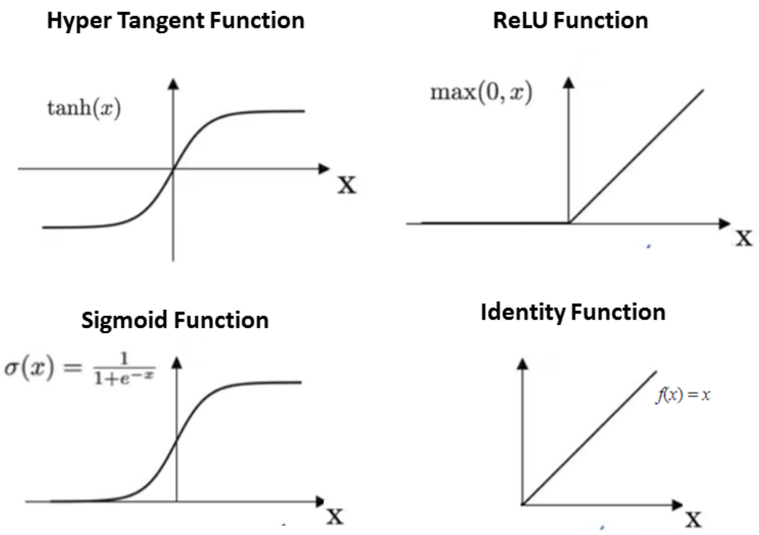


Figure II.8: Some activation function [8]

4.3 The Cost function

A cost function is a measure of error between what value your model predicts and what the value actually is , it represented by following equation:

$$J(w, b) = \frac{1}{m} \sum_{j=1}^{m} (y_j - \widehat{y_j})^2 \tag{II.8}$$

Where y_j is the actual value and $\widehat{y_j}$ is the pridect value.

[8] This Picture was taken from (https://medium.com/@zeeshanmulla/cost-activation-loss-function-neural-network-deep-learning-what-are-these-91167825a4de) accessed 16 June 2021.

The goal is to **minimize the cost function** by use the concept of **gradient descent** (there are other **optimizer**), we want to find the best parameters (W) for our learning algorithm. We can apply the same analogy and find the best possible values for that parameter

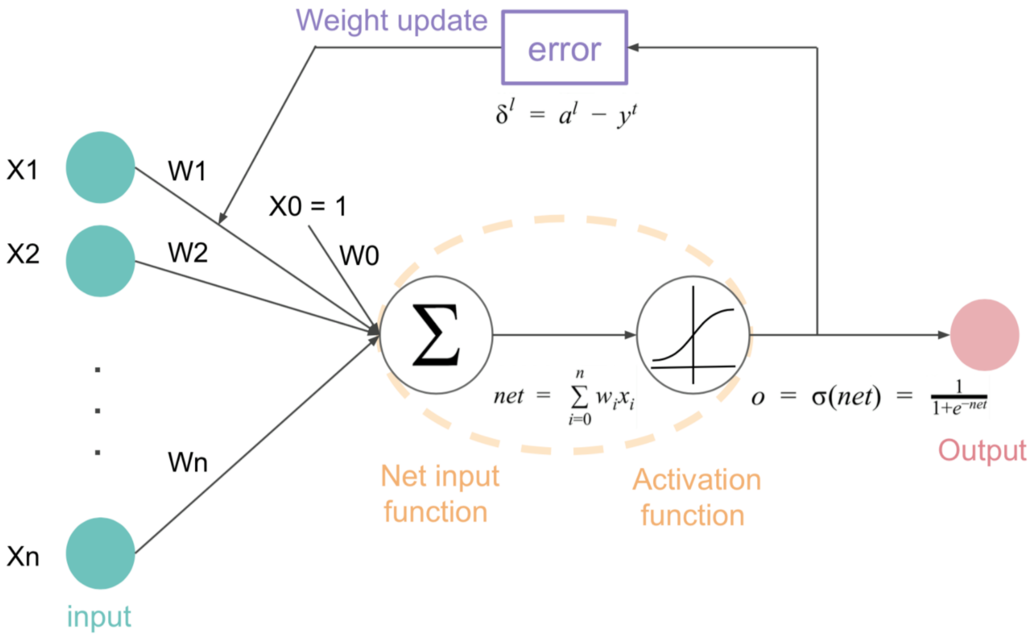


Figure II.9: Backpropagation [9]

4.4 Optimizer and Backpropagation

Optimizers are algorithms used to change the parameters of the artificial neural network such as weights and learning rate to reduce the losses values in backpropagation phase by update the parameters of our model. Parameters refer to weights and bias ,as described in equations II.9.(Changed weights) and equation II.10 (changed bias).

$$w_{n+1} = w_{(n)} - \alpha \frac{\partial J(w_n, b_n)}{\partial w} \tag{II.9}$$

$$b_{n+1} = b_{(n)} - \alpha \frac{\partial J(w_n, b_n)}{\partial b} \tag{II.10}$$

[9] This Picture was taken from (https://medium.com/@zeeshanmulla/cost-activation-loss-function-neural-network-deep-learning-what-are-these-91167825a4de) accessed 16 June 2021.

4.4.1 Type of optimizer

4.4.1.1 Gradient descent : is a first-order iterative optimization algorithm used to minimize some function(in our case, is the loss function of the neural network) by iteratively moving in the direction of steepest descent as defined by the negative of the gradient. In neural networks, we use gradient descent to update the parameters of our model. Parameters refer to weights and bias There are three variants of gradient descent algorithm :

- **Batch gradient descent :** is the very basic gradient descent algorithm ,it calculates the weights and bias variations for each observation but performs the learning (weights and bias update) only after all observations have been evaluated, or, in other words, after a so-called epoch[12].
- **Stochastic gradient descent :** calculates the gradient of the cost function and then updates weights and biases for each observation in the dataset[12].

$$w_{t+1} = w_{(t)} - \alpha J(0; x^{(i)}; y^{(i)}) \tag{II.11}$$

- **Mini-batch gradient descent :** In thid term , datasets are split into a fixed number of small groups of observations called batches, and weights and biases are updated only after each batch has been fed to the model[12]. . This is by far the method most commonly used in the field of deep learning[12].

4.4.1.2 Momentum : The idea behind the Momentum optimizer is to use exponentially weighted averages of the corrections of the gradient and then use them for the weights updates[12] . The updete rule is following :

$$w_{n+1} = w_{(n)} - \alpha V_{dw} \tag{II.12}$$

$$b_{n+1} = b_{(n)} - \alpha V_{db} \tag{II.13}$$

where

$$V_{dw} = \beta V_{dw} + (1 - \beta) dw \tag{II.14}$$

$$V_{db} = \beta V_{db} + (1 - \beta) db \tag{II.15}$$

4.4.1.3 RMSprop : Root Mean Square Propagation optimizer restricts the oscillations in the vertical trend . So, we can rise our learning rate and our algorithm abled to take larger steps in the horizontal trend converging faster.

$$w_{n+1} = w_{(n)} - \alpha \frac{V_{dw}^{corrected}}{S_{dw}^{corrected} + \epsilon} \tag{II.16}$$

$$b_{n+1} = b_{(n)} - \alpha \frac{V_{db}^{corrected}}{S_{db}^{corrected} + \epsilon} \tag{II.17}$$

where

$$S_{dw} = \beta_2 S_{dw} + (1 - \beta_2) dw^2 \tag{II.18}$$

$$S_{db} = \beta_2 S_{db} + (1 - \beta_2) db^2 \tag{II.19}$$

4.4.1.4 Adaptive Moment estimation (Adam) :

It combines the ideas of RMSProp and Momentum in one optimizer [12].Like Momentum, it uses an exponential weighted averages of past derivatives[12], and like RMSProp, it uses the exponentially weighted averages of past squared derivatives [12]. You will have to calculate the same quantities that you need for Momentum and for RMSProp[12].

$$V_{dw} = \beta_1 V_{dw} + (1 - \beta_1) dw \tag{II.20}$$

$$V_{db} = \beta_1 V_{db} + (1 - \beta_1) db \tag{II.21}$$

$$S_{dw} = \beta_2 S_{dw} + (1 - \beta_2) dw^2 \tag{II.22}$$

$$S_{db} = \beta_2 S_{db} + (1 - \beta_2) db^2 \tag{II.23}$$

with initiale $S_{db} = 0, S_{dw} = 0, V_{db} = 0, V_{dw} = 0$

Values are often fixed with $\beta_1 = 0.9 \leftarrow (dw)$, $\beta_2 = 0.9 \leftarrow (dw^2)$, $\epsilon = 10^{-8}$

Applying the chain rule to the computation of the gradient values of a neural network brings on to an algorithm called **Backpropagation**[8].

In the simplest way possible, the idea is to pass the training set through the hidden layers of the neural network. after that , we need to update the parameters of the layers(weights) (if there an error) by computing the optimizers using the training selected from the original training dataset.

4.5 Limitation of Single-Layer perceptron

After all , the perceptron is simply devided data into two categories , one side of the line are classified into one category, inputs on the other side are classified into another. for instance , it can learn the operation like AND , OR with it. But,Single-Layer perceptron cannot deal with non-linearly separable data,in otherwords , When we have multi categories , perceptron cant not seperates that with single line , sample example XOR function(as shown in figure II.10).

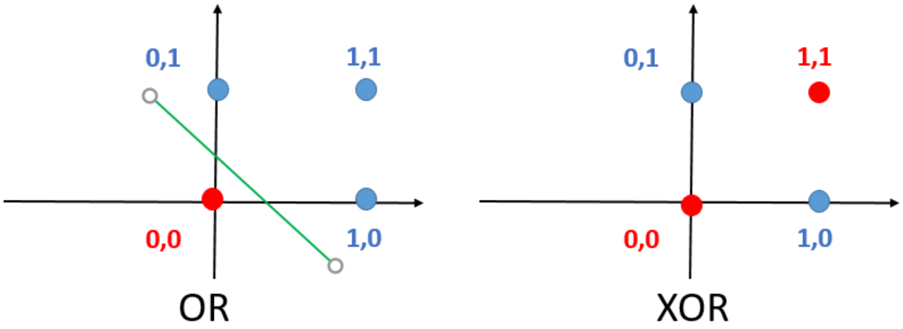


Figure II.10: A CNN sequence to classify handwritten digits

For solving this problem, a multi perceptron can be used.

4.6 Multilayer Perceptron

A multilayer perceptron is a deep feedforward artificiel neural network with one or more than single layer perceptron,also its Learning proceeds the same way as for a perceptron, The are composed of an input layer to receive the signal,an output layer that makes a decision or prediction about the input , and in middle of those two,a number of hidden layers(shown in Figure in II.11) ,Each layer in a multilayer neural network has its own specific function[12] ,the weights of the neurons represent the features hidden in the input . These features are then used by the output layer in determining the output target .if there is an error , in other words in anther words if there a difference between the actual value and the predected one , we must reduce that by change the value of weights.

4.7 The Problem of Overfitting

The problem of overfitting refers to the fact that fitting a model to a particular training data set does not guarantee that it will provide good prediction performance on unseen test data, even if the model predicts the targets on the training data perfectly. In other words, there is always a gap between the training and test data performance, which is particularly large when the models are complex and the data set is small[9].

[10] This Picture was taken from (`https://towardsdatascience.com/its-deep-learning-times-a-new-frontier-of-data-a1e9ef9fe9a8`) accessed 16 June 2021.

[11] This Picture was taken from (`https://www.quora.com/Is-overfitting-okay-if-the-test-accuracy-is-hi` accessed 16 June 2021.

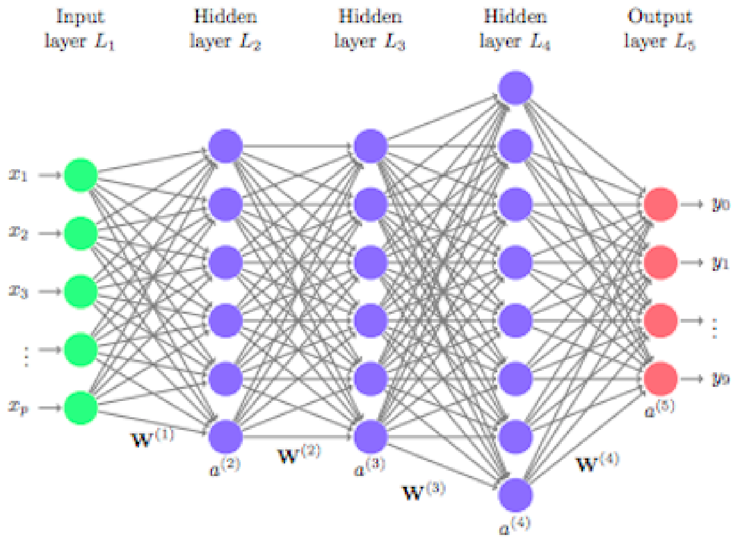


Figure II.11: A multi-layer perceptron[10]

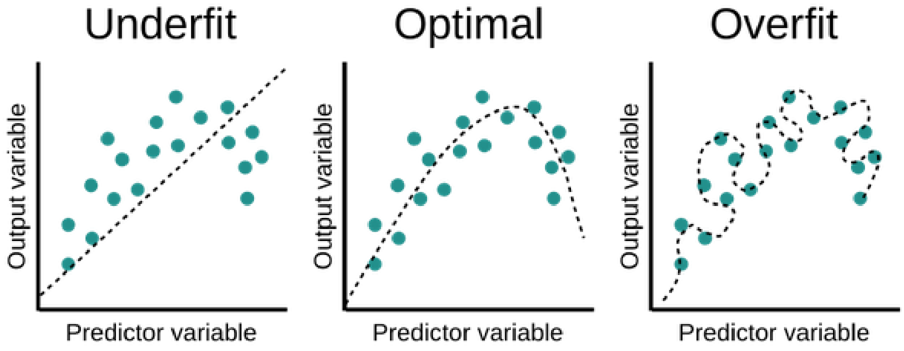


Figure II.12: The Problem of Overfitting [11]

4.8 Regularization

Regularization is a process of avoiding the problem of overfitting by apply their techniques to the cost function ,For instance , data augmentation and dropout .

4.8.1 Data Augmentation

Data augmentation artificially increases the size of the training set by generating many realistic variants of each training instance[7]. This reduces overfitting, making this a regularization technique. The generated instances should be as realistic as possible[7]. We going to talk about droupout in section 4.9.2.

4.9 Type of ANN

4.9.1 Recurrent neural networks

Recurrent neural networks (RNNs) are a variant of neural networks in which units are connected along a sequence[13], it can scale to much longer sequences than would be practical for networks without sequence-based [14].

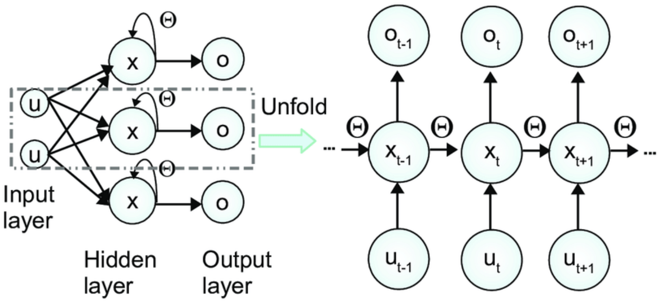


Figure II.13: A CNN sequence to classify handwritten digits [12]

4.9.2 convolutional neural network

convolutional neural network (CNN) are an extension of multi-layer perceptron to effectively avoid the major flaws of MLPs. They are designed to automatically extract the features of the input images and to classify these features.

There are three types of layers in a convolutional network:

- Convolution Layer The Kernel
- Pooling Layer
- Classification Fully Connected Layer

[12] This Picture was taken from (https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53) accessed 16 June 2021.

[13] This Picture was taken from (https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53) accessed 16 June 2021.

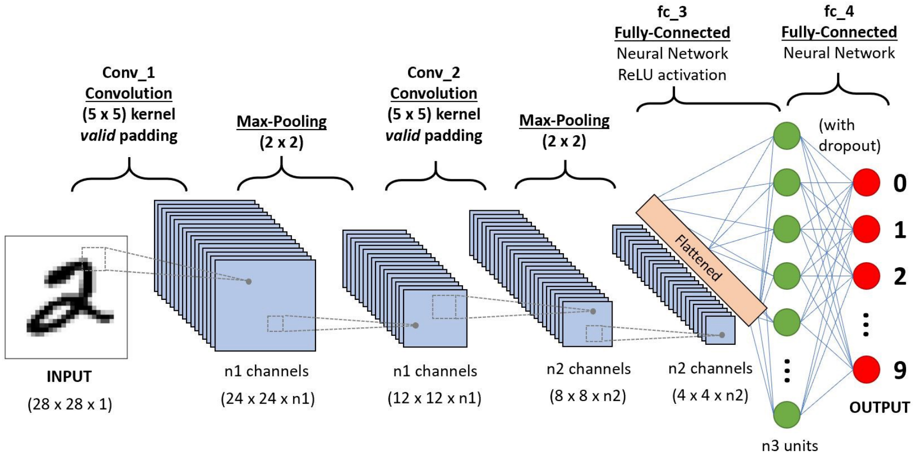
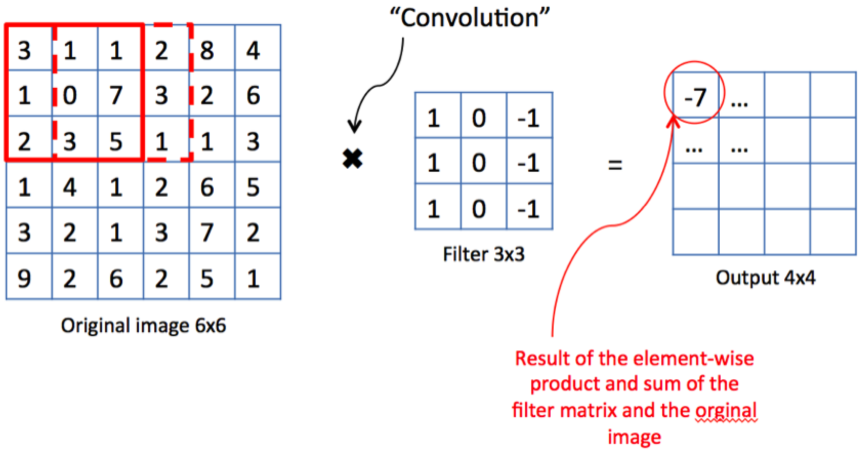


Figure II.14: A CNN sequence to classify handwritten digits [13]

4.9.2.1 Convolution Layer The Kernel

The convolution layer is nothing other than something that transforms the input into an output tensor[12].


"Convolution"
✖
=
Result of the element-wise product and sum of the filter matrix and the orginal image

3	1	1	2	8	4
1	0	7	3	2	6
2	3	5	1	1	3
1	4	1	2	6	5
3	2	1	3	7	2
9	2	6	2	5	1

Original image 6x6

1	0	-1
1	0	-1
1	0	-1

Filter 3x3

-7	...		
...	...		

Output 4x4

Figure II.15: A CNN sequence to classify handwritten digits [14]

[14] This Picture was taken from (`https://medium.com/machine-learning-bites/deeplearning-series-convolutional-neural-networks-a9c2f2ee1524`) accessed 16 June 2021.

4.9.2.2 Pooling Layer the pooling layers are extremely easy to understand than convolution layer. Their goal is to subsample the input image in order to reduce the computational load[7], the memory usage[7], and the number of parameters (thereby limiting the risk of overfitting)[7].

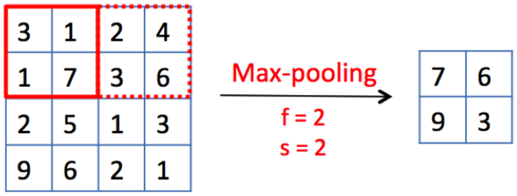


Figure II.16: Average pooling[15]

4.9.2.3 Classification Fully Connected Layer A layer in which neurons are connected to all neurons of previous and subsequent layers[12].

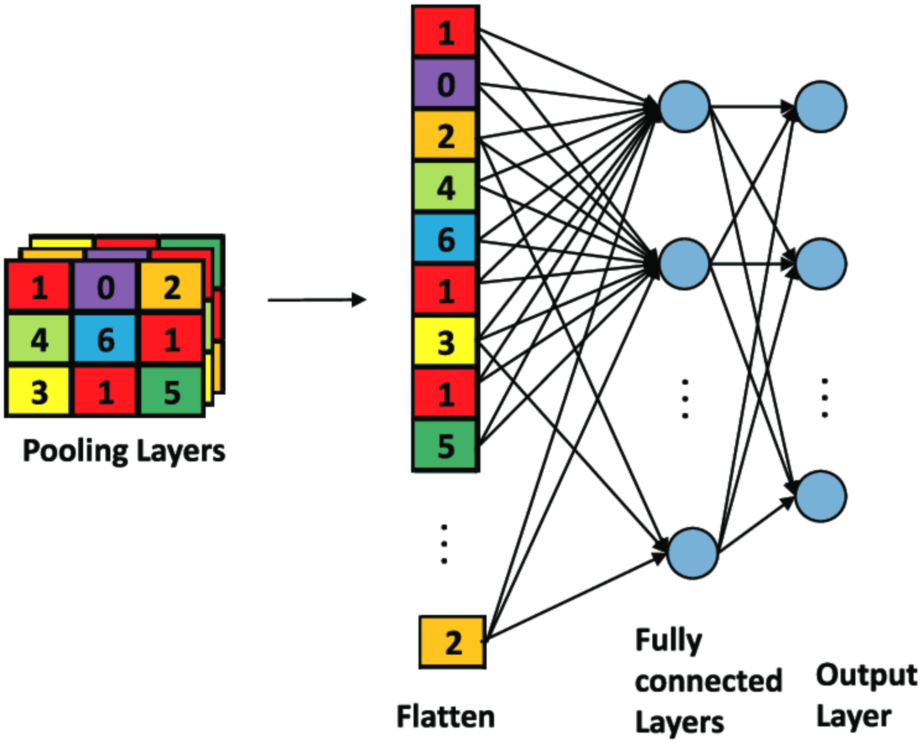


Figure II.17: Fully Connected Layer[16]

[15] This Picture was taken from (https://medium.com/machine-learning-bites/deeplearning-series-convolutional-neural-networks-a9c2f2ee1524) accessed 16 June 2021.

4.9.2.4 convolutional neural network known models

ResNet : It was developed by Alex Krizhevsky (hence the name), Ilya Sutskever, andGeoffrey Hinton in 2012[7] [15]. They enter an identity x to the output function as shown in a equation II.24 , where function F(x) and input x have a different dimensionality because a convolution operation usually reduces the spatial resolution of an image.and figure II.18 shows us how it allows for the identity(input) x and F(x) to be combined as input to the next layer.

$$y = f(x) + x \tag{II.24}$$

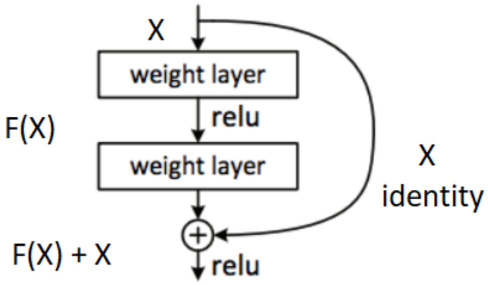


Figure II.18: Example of a classic ResBlock

LeNet : LeNet was one of the earliest convolutional neural networks models, it consists of seven layers. In addition to input layer , In the figure II.19 , every convolutional layer includes three parts: convolution, pooling, and activation functions .

AlexNet : AlexNet contained eight layers; the first five were convolutional layers, where the fist , second and last followed by max-pooling layers , and the last three were fully connected layers. It used the ReLU activation function.

[16] This Picture was taken from (https://medium.com/analytics-vidhya/from-convolutional-neural-network-to-variational-auto-encoder-97694e86bb51) accessed 16 June 2021.

[17] This Picture was taken from (https://en.m.wikipedia.org/wiki/File:Comparison_image_neural_networks.svg) accessed 01 July 2021.

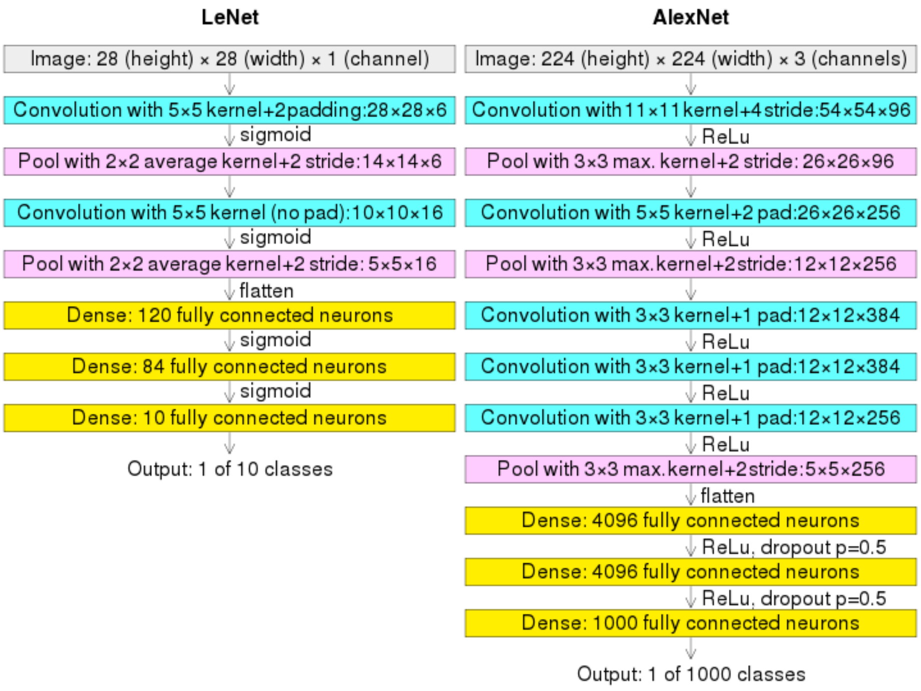

LeNet
Image: 28 (height) × 28 (width) × 1 (channel)
Convolution with 5×5 kernel+2padding:28×28×6
sigmoid
Pool with 2×2 average kernel+2 stride:14×14×6
Convolution with 5×5 kernel (no pad):10×10×16
sigmoid
Pool with 2×2 average kernel+2 stride: 5×5×16
flatten
Dense: 120 fully connected neurons
sigmoid
Dense: 84 fully connected neurons
sigmoid
Dense: 10 fully connected neurons
Output: 1 of 10 classes
AlexNet
Image: 224 (height) × 224 (width) × 3 (channels)
Convolution with 11×11 kernel+4 stride:54×54×96
ReLu
Pool with 3×3 max. kernel+2 stride: 26×26×96
Convolution with 5×5 kernel+2 pad:26×26×256
ReLu
Pool with 3×3 max.kernel+2stride:12×12×256
Convolution with 3×3 kernel+1 pad:12×12×384
ReLu
Convolution with 3×3 kernel+1 pad:12×12×384
ReLu
Convolution with 3×3 kernel+1 pad:12×12×256
ReLu
Pool with 3×3 max.kernel+2stride:5×5×256
flatten
Dense: 4096 fully connected neurons
ReLu, dropout p=0.5
Dense: 4096 fully connected neurons
ReLu, dropout p=0.5
Dense: 1000 fully connected neurons
Output: 1 of 1000 classes

Figure II.19: LeNet versus AlexNet [17]

4.10 Autoencoder

An autoencoder is a type of artificial neural network used to learn efficient codings of unlabeled data (unsupervised learning) [16] As shown in figure II.20 , Autoencoders have an encoder-decoder structure to learning, the fist one **encoder** transforms the input to a latent space vector.and the second one **decoder** takes in these encodings to produce output .

[18] This Picture was taken from (https://towardsdatascience.com/building-a-carbon-molecule-autoencoder-21973e5f88b6) accessed 01 July 2021.

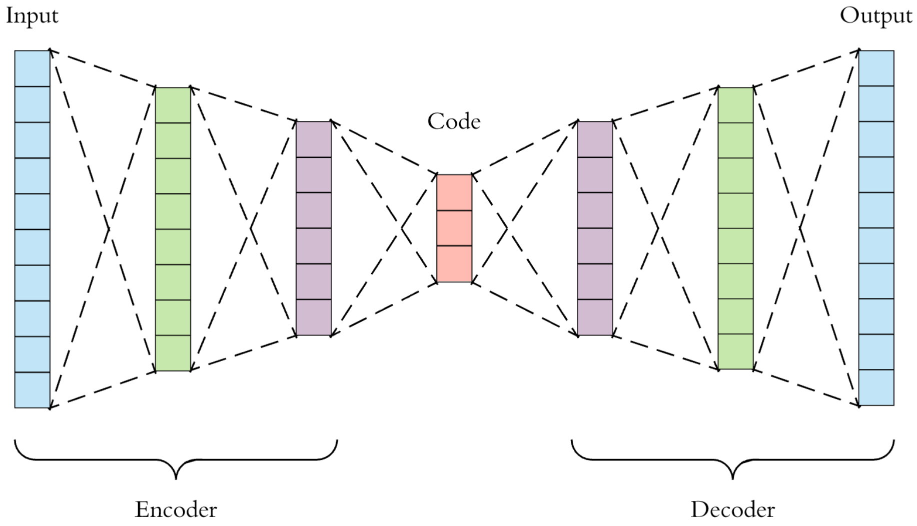


Figure II.20: Autoencoder architecture. [18]

Conclusion

In this chapter, we have presented Artificial intelligence , Machine learning in general and deep learning In particular , In deep learning , we have shown an overview of the perceptron and his type with how it works , and the most popular optimisers for create a optimale model .

Chapter III

State of the art

Introduction

SVHN is a new and popular dataset, it is like MNIST dataset to some extent but more complex ,and complement to other popular datasets.used for developing machine learning, object recognition and detection algorithms.in this chapter we take an overview of the common preprocessing techniques used and the best performance measures.

1 Related work

1.1 Netzer and al. paper

There are many papers studying the recognize and detection digit number with SVHN dataset, with both two problems of machine learning , Supervised learning and unsupervised learning . Netzer and al. In original paper that introduced the SVHN dataset [1], they used too minimal preprocessing by converting the images to grayscale , 32X32 pixel size of image, with using unsupervised learning algotithms (like K-Means clustering algorithm and Support Vector Machines). in their paper , they focused on a restricted instance of the scene text problem: reading digits from house-number signs in street level images[1]. For [1] application,they used two different feature learning schemes applied to a large corpus of digit data and compare them to hand-crafted representations [1].Each of their model , their used very minimal preprocessing by converting the images to grayscale with input images 32-by-32 pixels , they extract features and then trained a linear SVM classifier from the labeled training data using these features as input, and test the classifier on the test set[1].they also trained with the K-Means clustering algorithm and other algorithms. After that , they present their experimental results,they selected the best classifiers for each of the different models[1] as shown in table III.1

ALGORITHM	SVHN-TEST (ACCURACY)
HOG	85.0%
BINARY FEATURES (WDCH)	63.3%
K-MEANS	90.6%
STACKED SPARSE AUTO-ENCODERS	89.7%
HUMAN PERFORMANCE	98.0%

Table III.1: Accuracies on SVHN-test.[1]

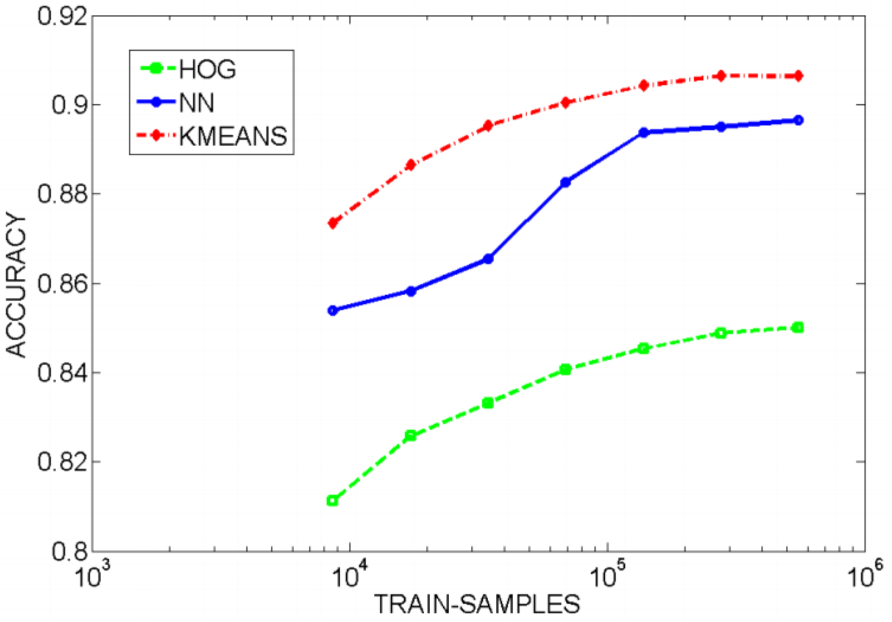


Figure III.1: Accuracies vs. number of training samples. Notice the log scale on the x-axis[1].

Large training set is another key to achieving high performance in addition to the use of learned feature representations[1].

In phase to evaluate how well their models perform in comparison to humans they sampled images from the set of images that their best model has misclassified[1]. they then evaluated the estimated accuracy of a few of the authors on the original resolution[1].

1.2 Lan Goodfellow paper

And in 2013, Goodfellow and others proposed a new activation function called maxout that is particularly well suited for training with dropout [2],they usesd some datasets include SVHN dataset,they select 400 samples for each class from the training set and 200 samples for each class from the extra set to built the validation set that used it only to find the best hyperparameters.

The rest digits of the train and extra sets are used for training.they did not train on the validation set at all. They used it only to find the best hyperparameters[2] , They applied preprocessing the same way as [4].

Model was composed of three convolutional maxout hidden layers and a densely connected maxout layer followed by a densely connected softmax layer. they got a test error rate of 2.47%, . A summary of comparable methods is provided in table III.2.

METHOD	TEST ERROR
Different pooling methods[17]	4.90%
Stochastic pooling[18]	2.80%
Rectifiers + dropout [19]	2.78%
Rectifiers + dropout + synthetic translation [19]	2.68%
Conv. maxout + dropout	2.68%

Table III.2: Test set misclassification rates for the best method on the SVHN dataset[2].

1.3 Makhanzi & Frey paper

In addition , Makhanzi & Frey proposed convolutional winner-take-all autoencoder (Stacked Conv-WTA Autoencoder) that learn to do fully-connected and convolutional sparse coding in [3] which combines the benefits of autoencoders and convolutional neural network architecture for learning shift-invariant spars representations [3].

SVM was used for classifying the learned representation in a similar fashion to the original paper [1] and in [20] . Training sparse autoencoders has been well studied in the literature[3].The learnt dictionary of a FC-WTA autoencoder trained on SVHN, MNIST, CIFAR-10 and Toronto Face datasets[3], They used about 600K in training points and 26K in test points in The SVHN dataset. First, they trained a shallow and stacked CONV-WTA on whole 600K training cases to learn the unsupervised features and then performed two sets of experiments,In the first experiment, they used whole the N=600K available labels to train an SVM on top of the CONV-WTA features, and compared the result with convolutional k-means [1] [3] , they could see that the stacked CONV-WTA achieves a dramatic improvement over the shallow CONV-WTA as well as k-means [1].In the second experiment,we trained an SVM by using only N = 1000 labeled data points and compared the result with deep variational autoencoders [20] trained in a same semi-supervised fashion [3]. The table III.3 reports the classification results of CONV-WTA autoencoder on the SVHN dataset.

	ACCURACY
Convolutional Triangle k-means[1]	90.6%
CONV-WTA Autoencoder, 256 maps(N=600K)	88.5%
Stacked CONV-WTA Autoencoder, 256 and 1024 maps(N=600K)	93.1%
Deep Variational Autoencoders (non-convolutional) [20] (N=1000)	63.9%
Stacked CONV-WTA Autoencoder, 256 and 1024 maps (N=1000)	76.2%
Supervised Maxout Network[2] (N=600K)	97.5%

Table III.3: CONV-WTA autoencoder trained on the Street View House Numbers (SVHN) dataset.[3]

,

1.4 Liang and Hu paper

Liang and Hu proposed a Recurrent Convolutional Neural Network (RCNN) model in [21],They also used some datasets comprise SVHN,The basic idea was to add recurrent connections within every convolutional layer of the feed-forward CNN [21], Thay followed the training procedure described in[2], Local contrast normalization was suggested to be an effective preprocessing step [18] and was adopted by many models including the maxout networks, NIN, DSN and DropConnect[21] , They subtracted the mean value from each pixel. With this preprocessing step, RCNN-128 outperformed the state-of-the-art models without data augmentation and two models with data augmentation[21] ,for more details take a look at table III.4 .

Model	No. of Param.	Testing Error (%)
Without Data Augmentation		
Maxout[2]	> 5M	2.47
Prob maxout[22]	> 5M	2.39
NIN[23]	1.98M	2.35
DSN[24]	1.98M	1.92
RCNN-128	1.19M	1.87
RCNN-160	1.86M	1.80
RCNN-192	2.67M	1.77
With Data Augmentation		
Multi-digit number Recognition[25]	> 5M	2.16
DropConnect(5 nets)[26]	-	1.94

Table III.4: Comparison with existing models on SVHN[3].

1.5 Lee, Gallagher and Tu paper

Lee, Gallagher and Tu [27] proposed improving deep neural networks with using some datasets including svhn datatset , they by generalize the pooling operations which are popular in convolutional neural network architectures.
they Our model has 128,128, 320, 320, 384, 384 channels for conv1 to conv6 and 96, 256, 256 channels for mlpconv1 to mlpconv3[27]. They proposed two approaches: merging max and average pooling by a learned pooling function, and learning a pooling function which was composed of a tree-structured fusion of learned pooling filters. They then merged these approaches in a single architecture to achieve new state of the art results on the SVHN dataset with result of 98.32% in acurracy rate.

1.6 Results

The current state of the art result according to those mesures is 1.69% test error produced by [27],which surpasses the 2% human performance estimated by [1]. The best available measure and the best esults achieved by these architectures for SVHN dataset are compared in the table III.5.

Architectures	Accuracy	Error
The Netzer et al. paper in [1] results		
stacked convolutional winner-take-all autoencoder	89.70%	10.30%
Convolutional K-means algorithm	0.90	9.40%
Pierre Sermanet & Yann LeCun paper in [17] results		
Different pooling methods	95.10%	4.90%
Liang & Hu paper in [21] results		
Recurrent Convolutional NN 192	98.23%	1.77%
Lee, Gallagher & Tu in [27] results		
Mixed max average pooling method	98.24%	1.76%
Gated Max verage pooling method	98.26%	1.74%
Tree Pooling method method	98.30%	1.70%
Tree pooling + Max average pooling method	98.32%	1.69%

Table III.5: comparison between best results achieved by these architectures

Conclusion

There are many papers for street view house numbers dataset, Certainly ,Mentioning all of them requires hundreds of pages,In this chaper we took an overview about popular paper with result of accuracy and test .

Chapter IV

Experiments and realization

Introduction

This chapter is devoted to the present the tools and libraries used in our experiments , the realization and the implementation of system and a discussion about the results.

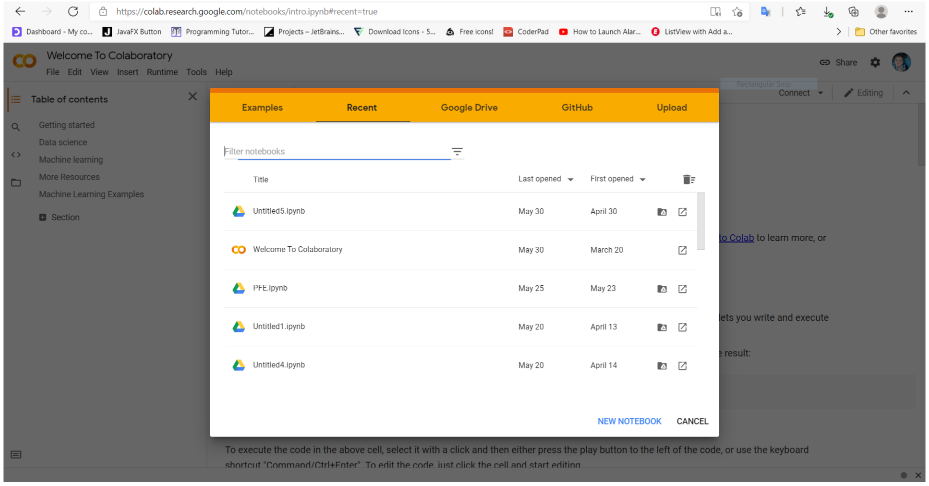
1 Tools and Libraries

1.1 Python

Python is an easy to learn, powerful programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming. Pythons elegant syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms [28].

1.2 Google Colaboratory

Colaboratory, or Colab for short, is a product from Google Research. Colab allows anybody to write and execute arbitrary python code through the browser, and is especially well suited to machine learning, data analysis and education. More technically, Colab is a hosted Jupyter notebook service that requires no setup to use, while providing free access to computing resources including GPUs [29].



Title	Last opened	First opened
Untitled5.ipynb	May 30	April 30
Welcome To Colaboratory	May 30	March 20
PFE.ipynb	May 25	May 23
Untitled1.ipynb	May 20	April 13
Untitled4.ipynb	May 20	April 14

to Colab to learn more, or

lets you write and execute

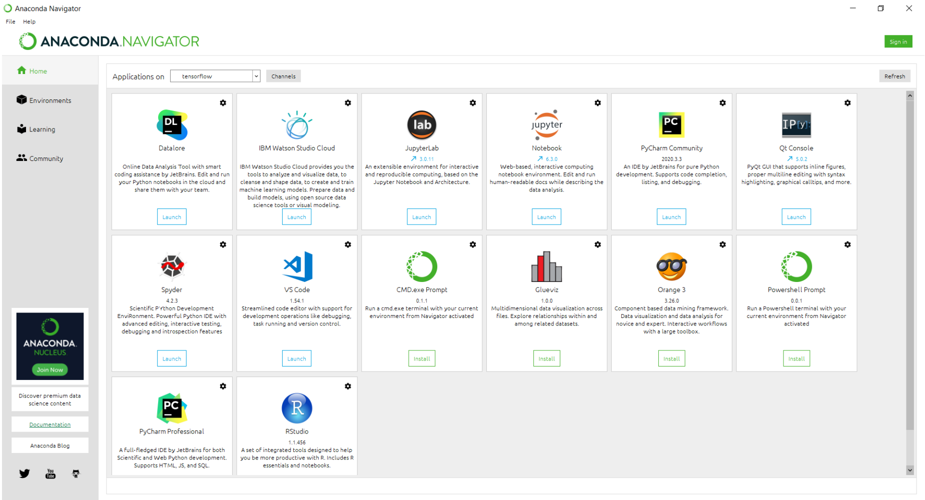
result:

To execute the code in the above cell, select it with a click and then either press the play button to the left of the code, or use the keyboard shortcut "Command/Ctrl+Enter". To edit the code, just click the cell and start editing.

Figure IV.1: Google Colaboratory

1.3 Anaconda navigator

Anaconda Navigator is a desktop graphical user interface (GUI) included in Anaconda distribution that allows you to launch applications and easily manage conda packages, environments, and channels without using command-line commands. Navigator can search for packages on Anaconda Cloud or in a local Anaconda Repository. It is available for Windows, macOS, and Linux [30].



Datalore
Online Data Analysis Tool with smart coding assistance by JetBrains. Edit and run your Python notebooks in the cloud and share them with your team.
Launch

IBM Watson Studio Cloud
IBM Watson Studio Cloud provides you the tools to analyze and visualize data, to cleanse and shape data, to create and train machine learning models. Prepare data and build models, using open source data science tools or visual modeling.
Launch

JupyterLab
↗ 3.0.11
An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.
Launch

Notebook
↗ 6.3.0
Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.
Launch

PyCharm Community
2020.3.3
An IDE by JetBrains for pure Python development. Supports code completion, listing, and debugging.
Launch

Qt Console
↗ 5.0.2
PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.
Launch

Spyder
4.2.3
Scientific PYthon Development EnviRonment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features
Launch

VS Code
1.54.1
Streamlined code editor with support for development operations like debugging, task running and version control.
Launch

CMD.exe Prompt
0.1.1
Run a cmd.exe terminal with your current environment from Navigator activated
Install

Glueviz
1.0.0
Multidimensional data visualization across files. Explore relationships within and among related datasets.
Install

Orange 3
3.26.0
Component based data mining framework. Data visualization and data analysis for novice and expert. Interactive workflows with a large toolbox.
Install

Powershell Prompt
0.0.1
Run a Powershell terminal with your current environment from Navigator activated
Install

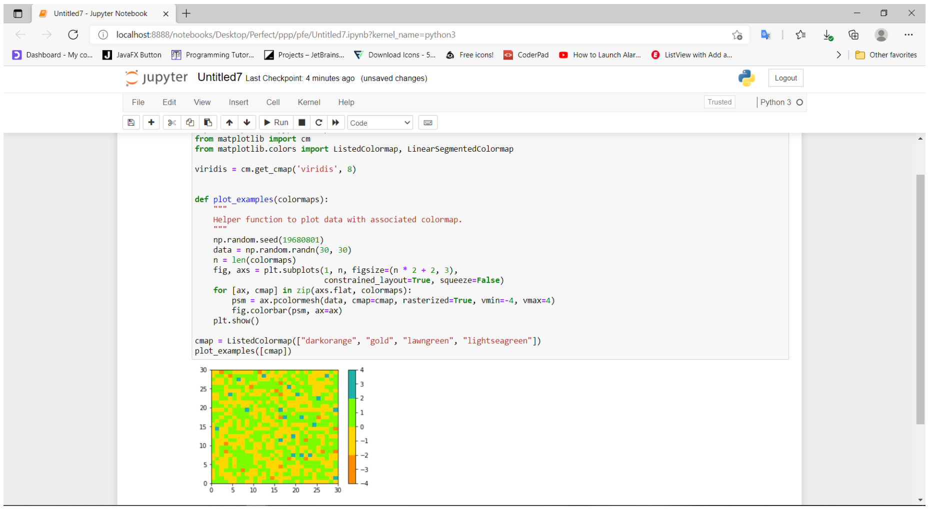
PyCharm Professional
A full-fledged IDE by JetBrains for both Scientific and Web Python development. Supports HTML, JS, and SQL.

RStudio
1.1.456
A set of integrated tools designed to help you be more productive with R. Includes R essentials and notebooks.

Figure IV.2: Anaconda navigator

1.4 Jupyter notebook

The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. Uses include: data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more [31].



```
from matplotlib import cm
from matplotlib.colors import ListedColormap, LinearSegmentedColormap

viridis = cm.get_cmap('viridis', 8)


def plot_examples(colormaps):
    """
    Helper function to plot data with associated colormap.
    """
    np.random.seed(19680801)
    data = np.random.randn(30, 30)
    n = len(colormaps)
    fig, axs = plt.subplots(1, n, figsize=(n * 2 + 2, 3),
                            constrained_layout=True, squeeze=False)
    for [ax, cmap] in zip(axs.flat, colormaps):
        psm = ax.pcolormesh(data, cmap=cmap, rasterized=True, vmin=-4, vmax=4)
        fig.colorbar(psm, ax=ax)
    plt.show()

cmap = ListedColormap(["darkorange", "gold", "lawngreen", "lightseagreen"])
plot_examples([cmap])
```

Figure IV.3: Jupyter notebook

1.5 Tensorflow

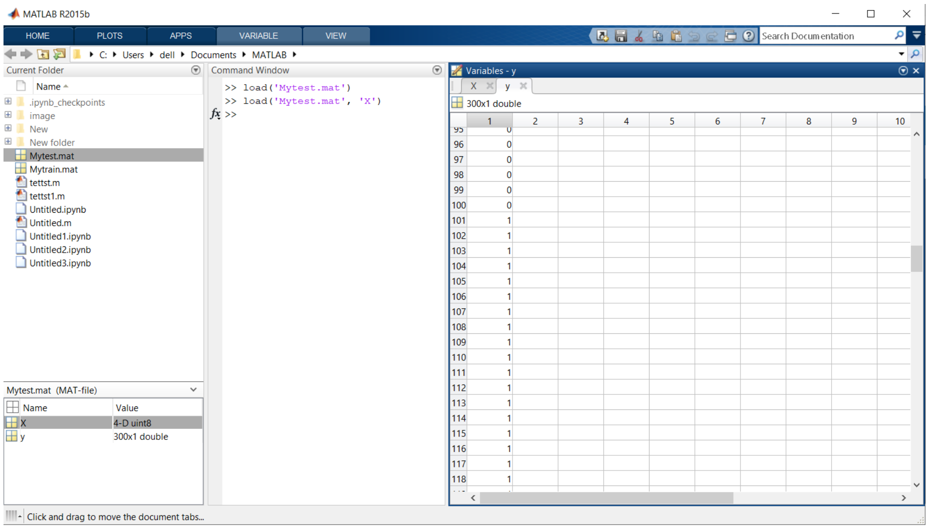
TensorFlow is an end-to-end open source platform for machine learning. It has a comprehensive, flexible ecosystem of tools, libraries and community resources that lets researchers push the state-of-the-art in ML and developers easily build and deploy ML powered applications[33].

1.6 Keras

Keras is a deep learning API written in Python, running on top of the machine learning platform TensorFlow. It was developed with a focus on enabling fast experimentation. Being able to go from idea to result as fast as possible is key to doing good research [34]

1.7 Matlab

MATLAB is a programming and numeric computing platform used by millions of engineers and scientists to analyze data, develop algorithms, and create models[32]. MATLAB combines a desktop environment tuned for iterative analysis and design processes with a programming language that expresses matrix and array mathematics directly. It includes the Live Editor for creating scripts that combine code, output, and formatted text in an executable notebook[32].



```
>> load('Mytest.mat')
>> load('Mytest.mat', 'X')
fx >>
```

Mytest.mat (MAT-file)

Name	Value
X	4-D uint8
y	300x1 double

300x1 double

	1	2	3	4	5	6	7	8	9	10
95	0									
96	0									
97	0									
98	0									
99	0									
100	0									
101	1									
102	1									
103	1									
104	1									
105	1									
106	1									
107	1									
108	1									
109	1									
110	1									
111	1									
112	1									
113	1									
114	1									
115	1									
116	1									
117	1									
118	1									

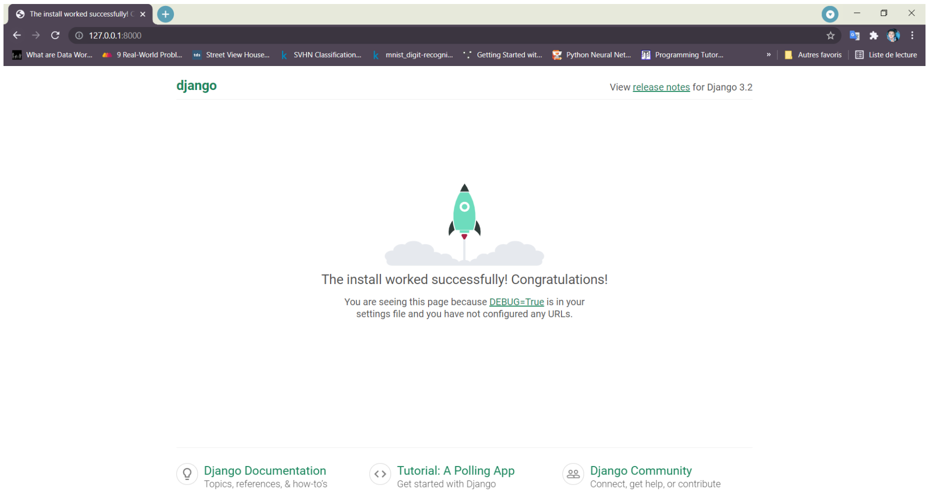
Figure IV.4: Matlab

1.8 OpenCV

OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being a BSD-licensed product, OpenCV makes it easy for businesses to utilize and modify the code[35].
The library has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms. These algorithms can be used to detect and recognize faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to produce a high resolution image of an entire scene, find similar images from an image database, remove red eyes from images taken using flash, follow eye movements, recognize scenery and establish markers to overlay it with augmented reality, etc. OpenCV has more than 47 thousand people of user community and estimated number of downloads exceeding 18 million. The library is used extensively in companies, research groups and by governmental bodies[35].

1.9 Django framework

Django is a high-level Python Web framework that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of Web development, so you can focus on writing your app without needing to reinvent the wheel. Its free and open source.[36]



View release notes for Django 3.2

The install worked successfully! Congratulations!

You are seeing this page because DEBUG=True is in your settings file and you have not configured any URLs.

Django Documentation
Topics, references, & how-to's

Tutorial: A Polling App
Get started with Django

Django Community
Connect, get help, or contribute

Figure IV.5: Django framework

1.10 SVNH dataset

SVHN is a real-world image dataset for developing machine learning and object recognition algorithms with minimal requirement on data preprocessing and formatting. It can be seen as similar in flavor to MNIST (e.g., the images are of small cropped digits), but incorporates an order of magnitude more labeled data (over 600,000 digit images) and comes from a significantly harder, unsolved, real world problem (recognizing digits and numbers in natural scene images). SVHN is obtained from house numbers in Google Street View images[37].

- 10 classes, 1 for each digit. Digit '1' has label 1, '9' has label 9 and '0' has label 10[37].

- 73257 digits for training, 26032 digits for testing, and 531131 additional, somewhat less difficult samples, to use as extra training data[37].

- Comes in two formats:

  1. Original images with character level bounding boxes[37].
  2. MNIST-like 32-by-32 images centered around a single character (many of the images do contain some distractors at the sides)[37].

1.10.1 Format 1 of SVHN

These are the original, variable-resolution, color house-number images with character level bounding boxes, as shown in the examples images above. (The blue bounding boxes here are just for illustration purposes. The bounding box information are stored in digitStruct.mat instead of drawn directly on the images in the dataset.) Each tar.gz file contains the orignal images in png format, together with a digitStruct.mat file, which can be loaded using Matlab. The digitStruct.mat file contains a struct called digitStruct with the same length as the number of original images. Each element in digitStruct has the following fields: name which is a string containing the filename of the corresponding image. bbox which is a struct array that contains the position, size and label of each digit bounding box in the image. Eg: digitStruct(300).bbox(2).height gives height of the 2nd digit bounding box in the 300th image[37].

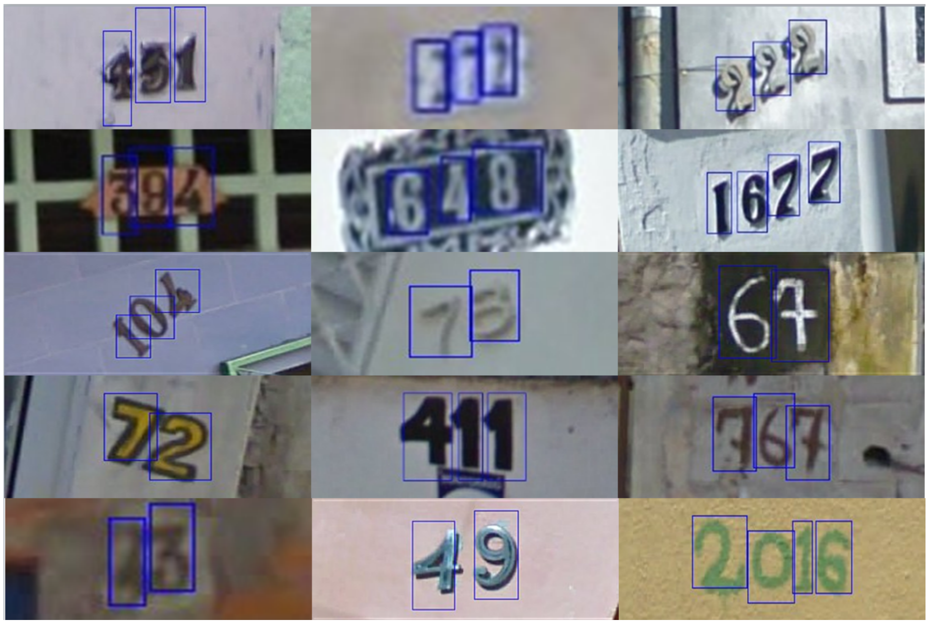


Figure IV.6: Image base of SVHN format 1

1.10.2 Format 2 of SVHN

Character level ground truth in an MNIST-like format. All digits have been resized to a fixed resolution of 32-by-32 pixels. The original character bounding boxes are extended in the appropriate dimension to become square windows, so that resizing them to 32-by-32 pixels does not introduce aspect ratio distortions. Nevertheless this preprocessing introduces some distracting digits to the sides of the digit of interest. Loading the .mat files creates 2 variables: X which is a 4-D matrix containing the images, and y which is a vector of class labels. To access the images, X(:,:,:,i) gives the i-th 32-by-32 RGB image, with class label y(i)[37].

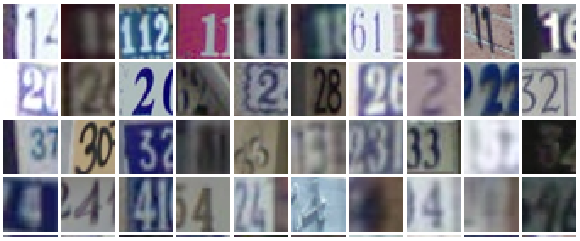


Figure IV.7: Image base of SVHN format 2

Number images in each class in train of SVHN dataset

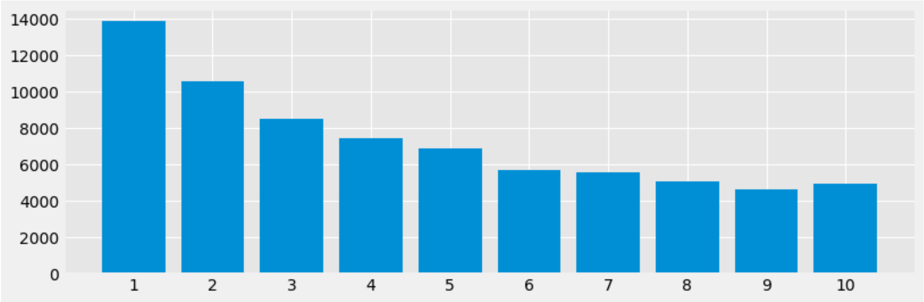


Figure IV.8: Number images in each class in train

Number images in each class in test of SVHN dataset

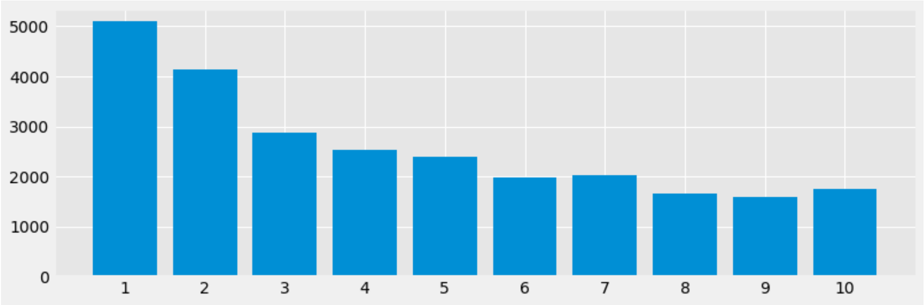


Figure IV.9: Number images per class in test

Table for number images in each class in test of SVHN dataset

Class	1	2	3	4	5	6	7	8	9	10
Train	13861	10585	8497	7458	6882	5727	5595	5045	4659	4948
Test	5099	4149	2882	2523	2384	1977	2019	1660	1595	1744

2 Experiments

The goal of this thesis is going to build a several models composed of several deferent layers of convolutional neural networks to recognize the digit numbers obtained from The Street View House Numbers (SVHN) Dataset format 2 .the model is going to focus on single digit classification.

2.1 Image bases

All the training is conducted using SVHN datasets combine with our datasets collections.

2.1.1 Our datasets

We could have made 2 datasets that contain the same characteristics with street view house number datasets format 2 with a simple code in matlab except that there are 3 classes(1 , 2 and 3) containing 300 images for each class in the first one for learning from images google , and in the second there are 100 images for each class for testing , Each picture comes with a its label (the picture to which it belong. Certainly, we compared between the SVHN datasets and our datasets, There is no similarity between them.

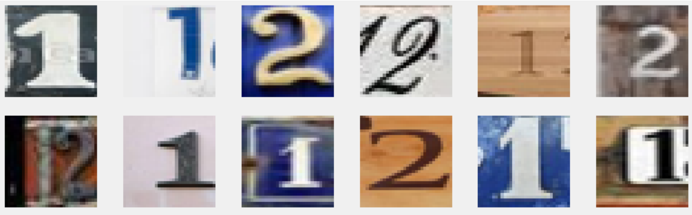


Figure IV.10: Collected Datasets

2.2 Architecture of our network

During our experiments, we created three models (model 1, model 2 and model 3) with different architectures and parameter, where all models were applied on the basis of combine the two datasets(our datasets with their one).

Our models were trained using Google Colab platform . the first model took around 8 seconds for each epoch to train, the second model took 6 seconds for each epoch to train and the last took 40 seconds for each epoch to train. We store the weights of the CNN model and record the accuracy and loss for every epoch to be able to evaluate and compare the results for each model.

Additionally, we have created a web application that allows you to recognize number in images.

In follows we present the architecture of the three models:

2.2.1 Architecture of model 1

The first model that we present in figure IV.11 is composed of eight convolutional layers , four layers of maxpooling ,five layer of dropout , four layer of batch normalization and two layers of fully connected . The input image is 32 * 32 size, the image goes through the first convolution layer . This layer is composed of 16 filters of size 3 * 3, Each of our convolution layers is followed by a ReLU activation function this function forces the neurons to return positive values following by batch normalization layer to keep the mean output close to 0 and the output standard deviation close to 1, after this convolution 16 feature maps of size 32 * 32 will be created.

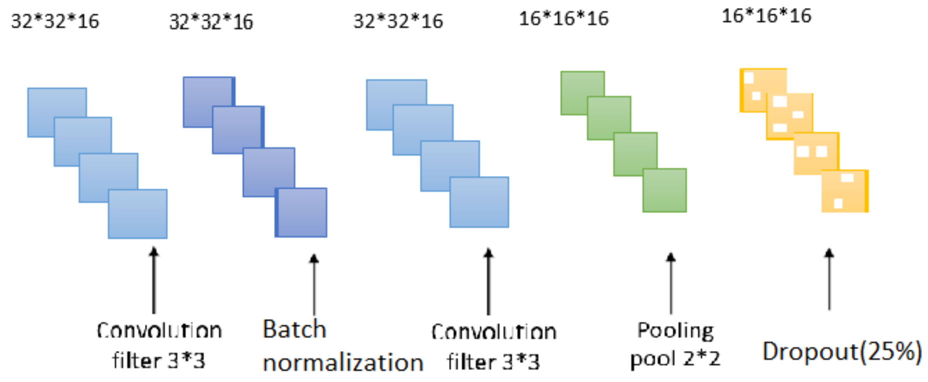


Figure IV.11: architecture of model 1

The 16 feature maps which are obtained before they are given as input of the second layer of convolution which is also composed of 16 filters, a RELU activation function is applied to the convolution layer, then we apply Maxpooling to reduce the size of the image and the quantity parameters and calculation followed by Dropout layer of 25% which randomly selected neurons to be dropped-out during training, As shown in figure IV.11 .

At the exit of this layer, we will have 16 feature maps of size 16 * 16, which are the input of the next layer , We repeated the same things three times with 32 ,64,128 filter of size 3x3.

The vector of characteristics resulting from the last dropout layer has a dimension of 512,that represent the result of 128 multiply by dimension of matrix where 128 is the sequence of matrices with dimension 2X2

After height layers of convolution, we use a neural network composed of two layers fully connected and a layer of dropout .(as shown in figure IV.12). The first layer have 128 neurons where the activation function used is the ReLU with 64 images as in input into each neuron, following by Dropout layer of 25%, and the last layer is a softmax which allows to calculate the distribution of probability of 11 classes (number of classes in the SVHN image base and our classes) with ADAM as an optimizer with learning rate is 0.01.

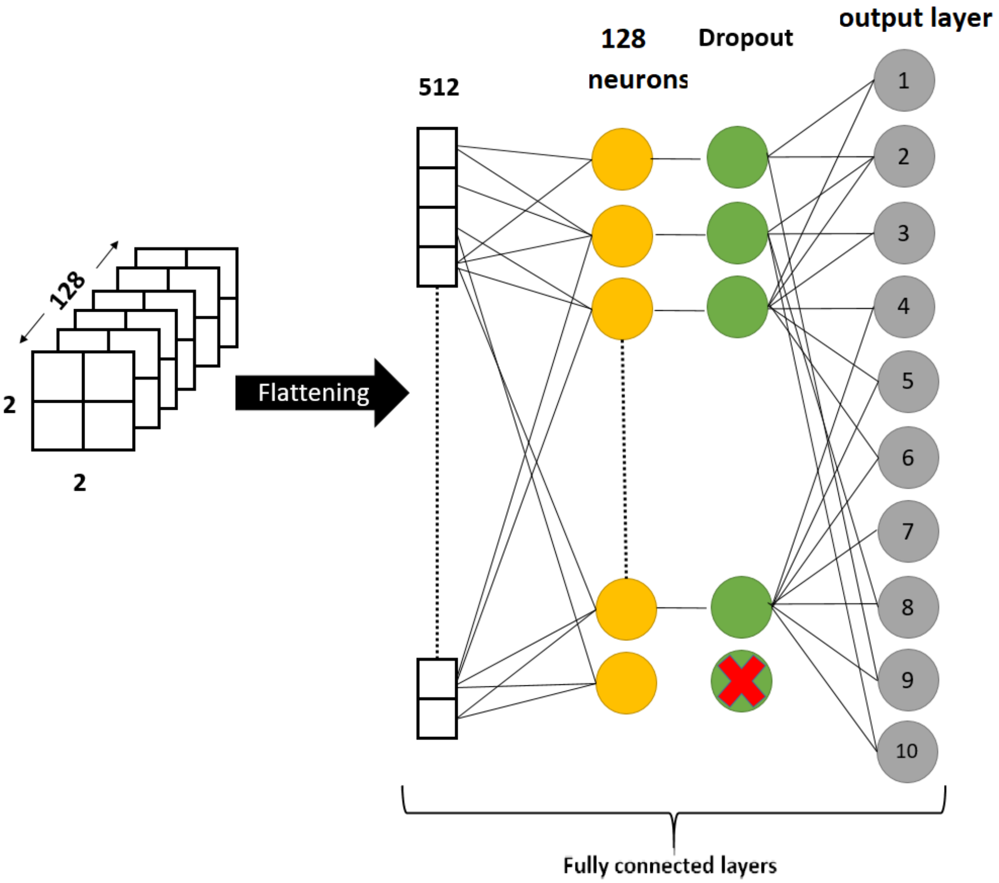


Figure IV.12: Fully connected of model 1

2.2.2 Architecture of model 2

The second model that we present in figure IV.13 is composed of eight convolutional layers and four layers of maxpooling ,six layer of dropout and three layers of fully connected.

The input image is 32 * 32 size, the image goes through same steps of previous model except that the first convolution layer is composed of 32 filters of size 3 * 3 instead of 16 filter . We repeated the same thing three times with 32 ,64,128 filters of size 3x3. At the end of last layer, we will have 128 feature maps of size 2*2.

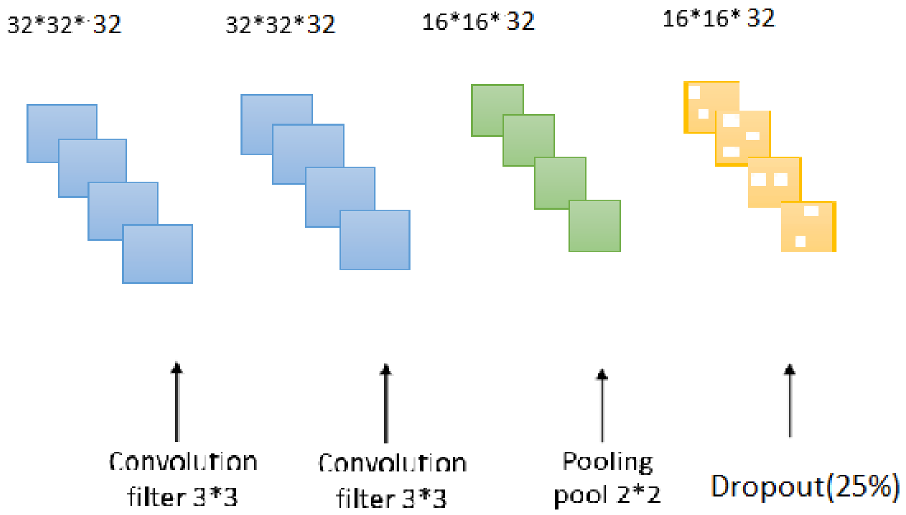


Figure IV.13: architecture of model 2

The vector of characteristics resulting from the ast dropout layer has a dimension of 512,that represent the result of 128 multiply by dimension of matrix where 128 is the sequence of matrices with dimension 2X2 (as shown in figure IV.14).

After height layers of convolution, we use a neural network composed of three layers fully connected and two layers of dropout (as shown in figure IV.14). The first layer have 1024 neurons where the activation function used is the ReLU with 128 images as in input into each neuron, following by dropout layer of 25%,And the third layer have 128 neurons with same activation function ,Also following by dropout layer of 25%, and the last layer is a softmax which allows to calculate the distribution of probability of 11 classes (number of classes in the SVHN image base and our classes) with ADAM as an optimizer with learning rate is 0.001.

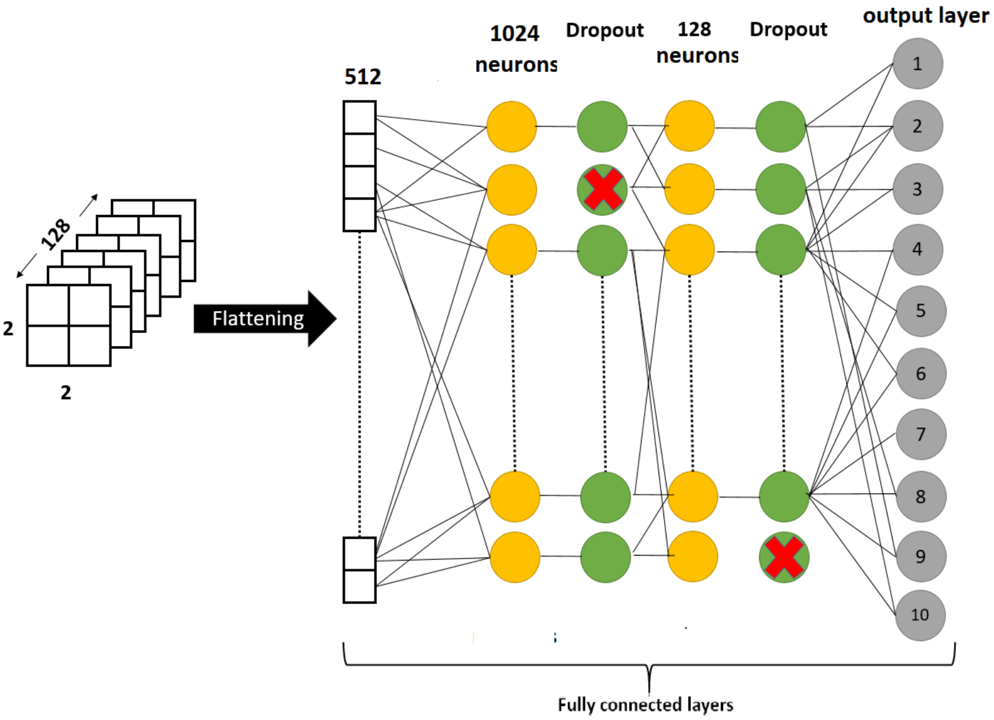


Figure IV.14: Fully connected of model 2

2.2.3 Architecture of model 3

The last model is composed of ten convolutional layers , five layers of max-pooling, seven layer of dropout , six layer of batch normalization and three layers of fully connected.

This model was trained using data augmentation (rotation range: 8 of degree for random rotations ,width shift range shift the image to the left or right,horizontal shifts. and height shift range to shift the image to up or down,vertically shifts) with keras module, . The input image is 32 * 32 size, the image goes through same steps of first model (as shown as in model 1 in fugure IV.11),except that the Drop out layer of 20% .

We repeated the same thing three times with 32,64,128,256 filters.

At the end of last layers, we will have 256 feature maps of size 1*1. The vector of characteristics resulting from the convolutions has a dimension of 256,that represent the result of 256 multiply by dimension of matrix where 256 is the sequence of matrices with dimension 1X1 (That comes from last dropout layer) (as shown in figure IV.15).

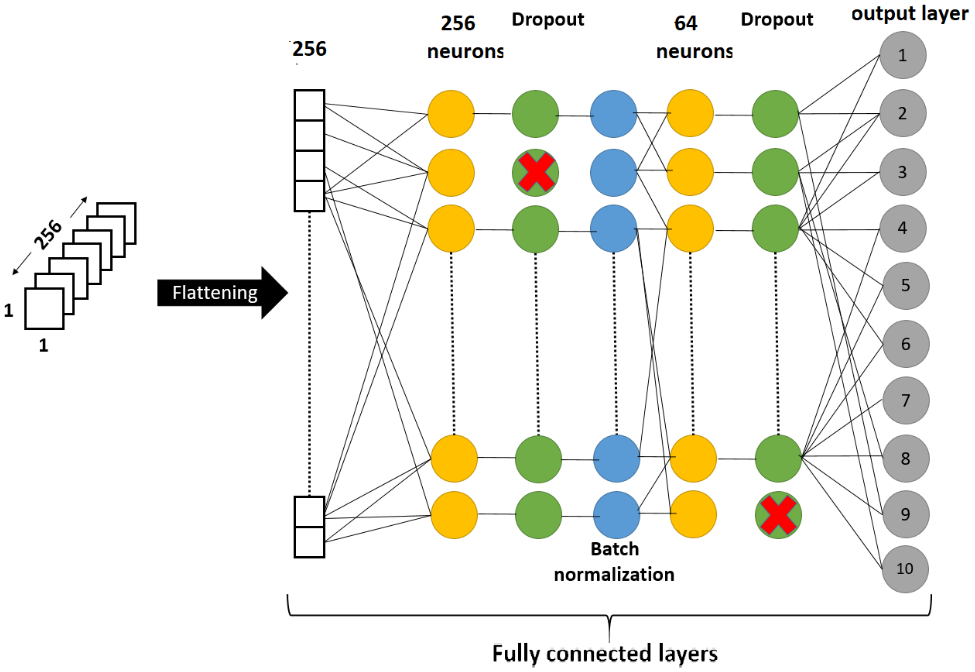


Figure IV.15: Fully connected of model 3

After height layers of convolution, we use a neural network composed of three layers fully connected ,a layer of batch normalization and two dropout layers of 10% ,the first layer has 256 neurons where the activation function used is the ReLU with 32 images as in input into each neuron following by dropout layer ant then batch normalization layer, and the fourth has 64 neurons followed by dropout layer, and the last layer is a softmax which allows to calculate the distribution of probability of 11 classes (number of classes in the SVHN image base and our classes) with ADAM as an optimizer with learning rate is 0.001.

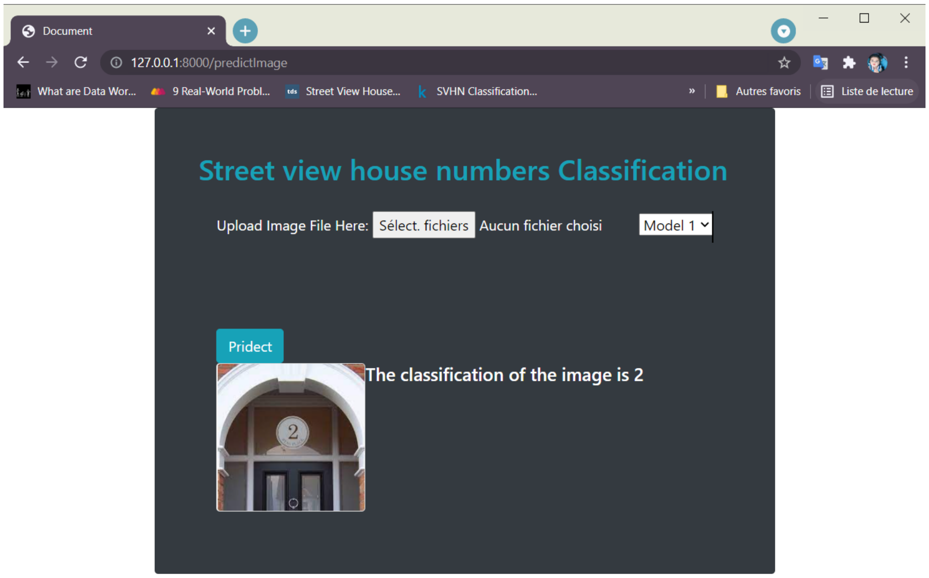
Model	Architecture			
	Conv	Maxout Pooling	Dropout	batch normalization
Model 1	8	4	5	4
Model 2	8	4	6	-
Model 3	10	5	7	6

Table IV.1: Layers of each model

3 Implementation

Our model architectures are based on [38] they were adjusted to change the accuracy value and recognize the various changed in each model for recognize number in images, and browser user interface was implemented using Django.

After the models have been well tuned by learning information from thousands of images and their corresponding numbers, the model could then be used to recognize the number in pictures not in the datasets bases which means it would be a generalized model for extracting numbers from picture used in many areas.



Street view house numbers Classification

Upload Image File Here: Sélect. fichiers Aucun fichier choisi Model 1

The classification of the image is 2

Figure IV.16: Browser user interface

4 Result and discussion

4.1 Results obtained for model 1

After analyzing the results obtained, the following remarks are noted:

From Figure IV.17 The accuracy of training and testing increases with the number of epochs, this reflects that with each epoch the model learns more informations. If the accuracy is reduced then we will need more informations to teach our model and therefore we must increase the number of epochs . Likewise, the learning and validation error decreases with the number of epochs.

We got an error rate of 22.63% in train and 25.45% in validation,and an accuracy rate of 94.51% in train and 93.66% in validation.

In confusion matrix IV.18, we must add 1 to labels to represents the classes of datasets for example , 0 is the 1 and 1 is the 2 and 9 is 10 and so on.

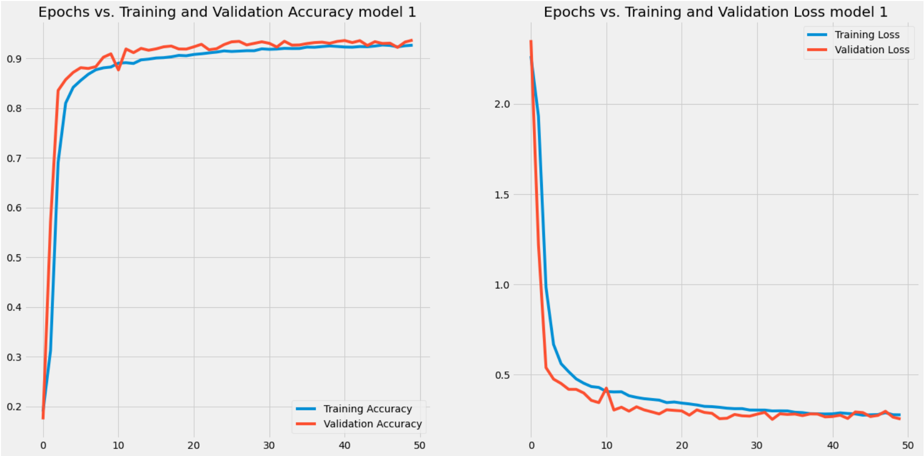
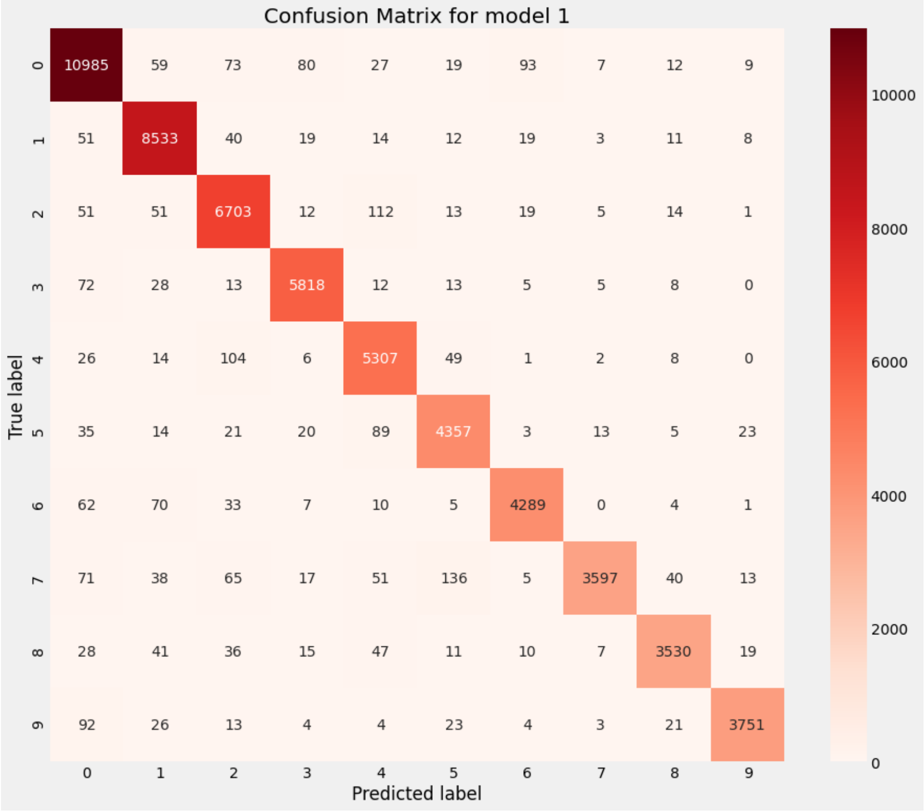


Figure IV.17: Accuracy and loss plots model 1



	0	1	2	3	4	5	6	7	8	9
0	10985	59	73	80	27	19	93	7	12	9
1	51	8533	40	19	14	12	19	3	11	8
2	51	51	6703	12	112	13	19	5	14	1
3	72	28	13	5818	12	13	5	5	8	0
4	26	14	104	6	5307	49	1	2	8	0
5	35	14	21	20	89	4357	3	13	5	23
6	62	70	33	7	10	5	4289	0	4	1
7	71	38	65	17	51	136	5	3597	40	13
8	28	41	36	15	47	11	10	7	3530	19
9	92	26	13	4	4	23	4	3	21	3751

Figure IV.18: Confusion matrix for model 1

The confusion matrix allows us to estimate the performance of our model, since it reflects the metrics of True Positive, True Negative, False Positive and False Negative. Figure IV.18 closely illustrates the position of these metrics for each class. For example, the model has correctly classified the images of number 1 with 10985 images .

4.2 Results obtained for model 2

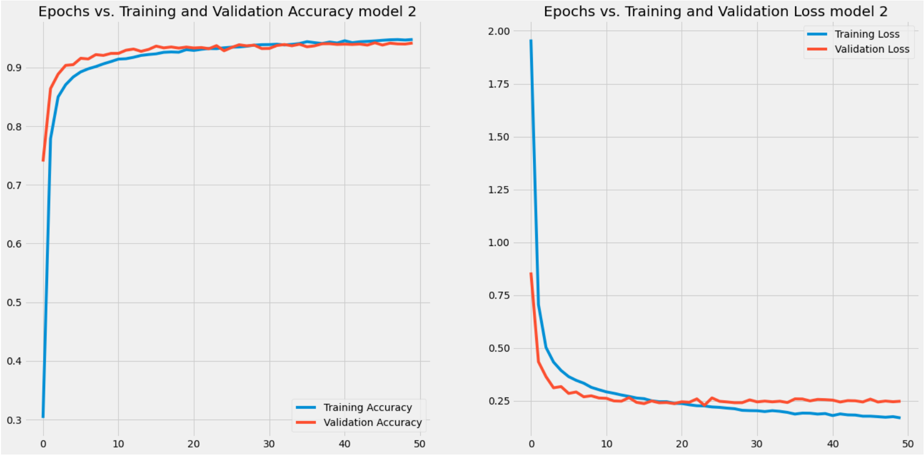


Figure IV.19: Accuracy and loss plots model 2

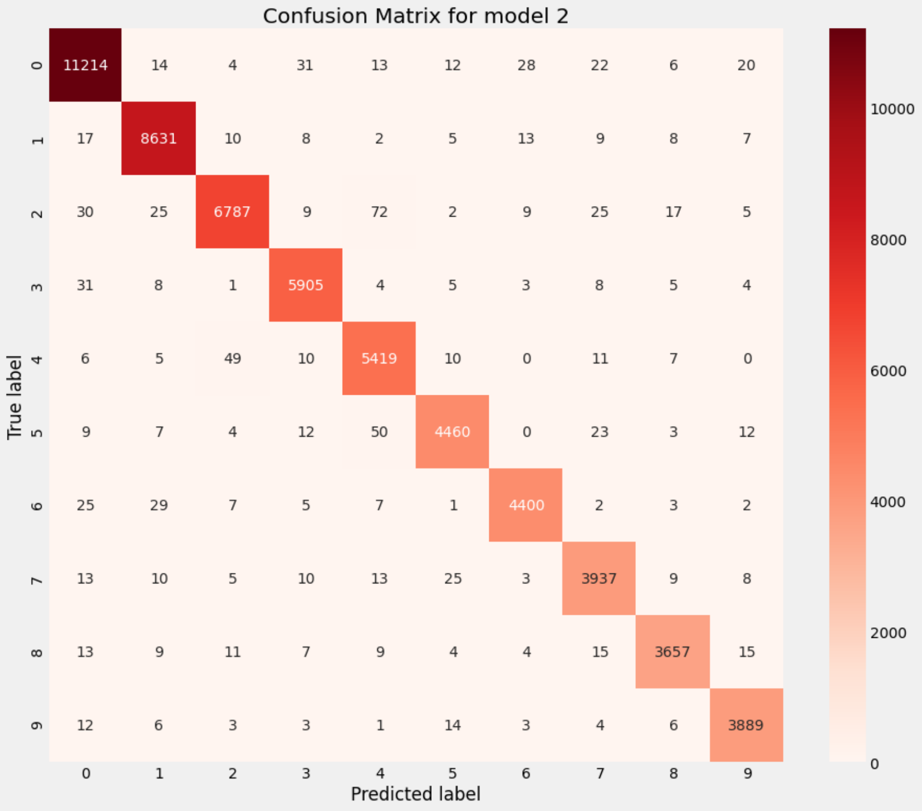
From Figure IV.20 The accuracy of learning and validation increases with the number period, this reflects that in every age the model learns more information. If the precision isdecreased then we will need more information to teach our model and therefore we must increase the number of epochs . Likewise, the learning and validation error decreases with the number of epochs.

We got an error rate of 25.33% in train and 24.84% in validation ,and an accuracy rate of 94.33% in train and 94.17% in validation.

Compared to the first model ,the accuracy rate in train is less than it, and the error rate in train is bigger than it ,But in validation , the accuracy is bigger and the error is less than it.

In confusion matrix IV.20, we must add 1 to labels to represents the classes of datasets for example , 0 is the 1 and 1 is the 2 and 9 is 10 and so on.

The confusion matrix allows us to assess the performance of our model, since it reflects the metrics of True Positive, True Negative, False Positive and False Negative. , as shown in Figure IV.20 closely illustrates the position of these metrics for each class.And compared to the model 1 , There are more corrects in classes than model 1 ,For example,the model has correctly classified the images of number 1 with 11214 images, and compared than model 1 , there are less errors than it , same things with accuracy , the model 2 predicted better than model 1.



Confusion Matrix for model 2

True label \ Predicted label	0	1	2	3	4	5	6	7	8	9
0	11214	14	4	31	13	12	28	22	6	20
1	17	8631	10	8	2	5	13	9	8	7
2	30	25	6787	9	72	2	9	25	17	5
3	31	8	1	5905	4	5	3	8	5	4
4	6	5	49	10	5419	10	0	11	7	0
5	9	7	4	12	50	4460	0	23	3	12
6	25	29	7	5	7	1	4400	2	3	2
7	13	10	5	10	13	25	3	3937	9	8
8	13	9	11	7	9	4	4	15	3657	15
9	12	6	3	3	1	14	3	4	6	3889

Figure IV.20: Confusion matrix for model 2

4.3 Results obtained for model 3

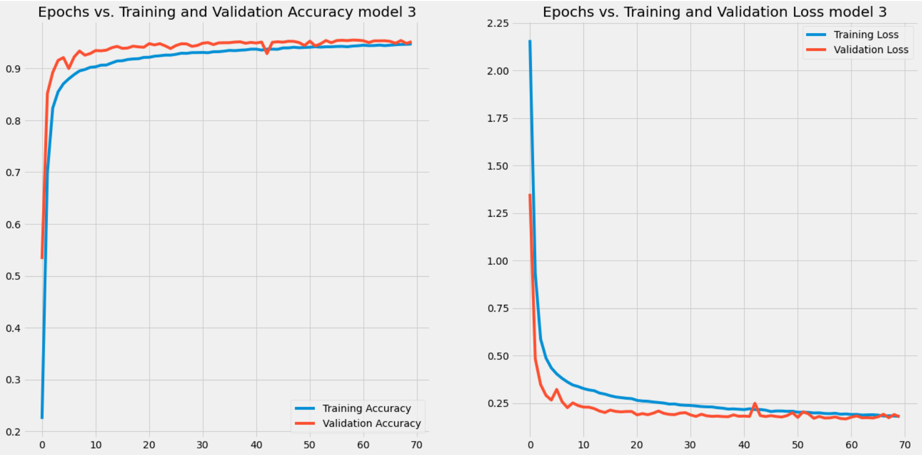


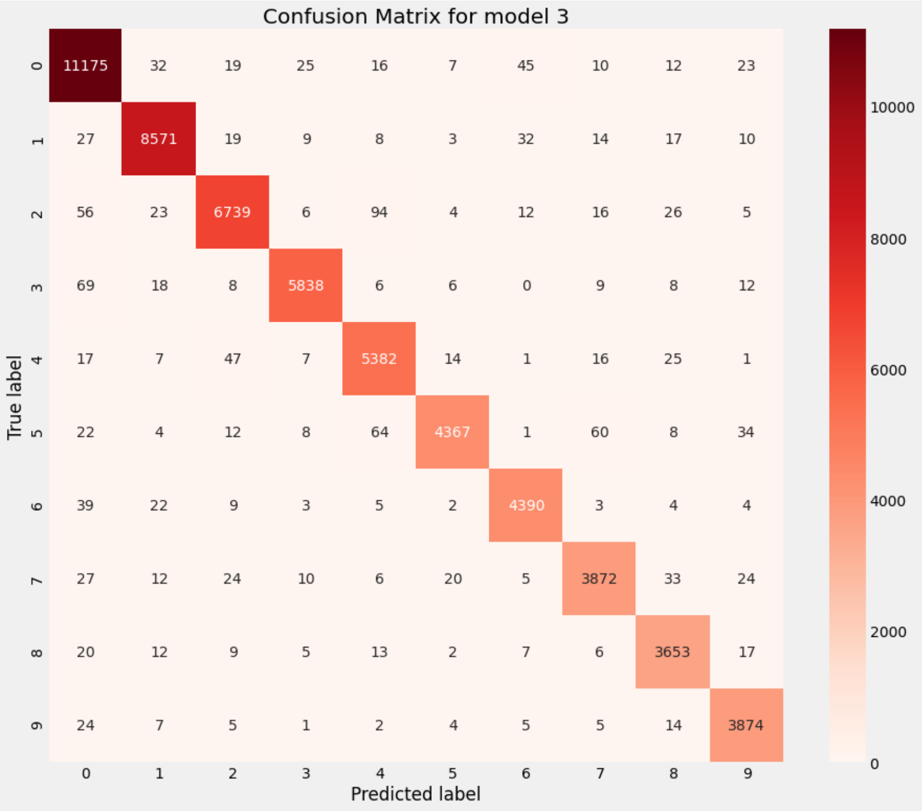
Figure IV.21: Accuracy and loss plots model 3

After analyzing the results obtained, the following remarks are noted: From Figure IV.21 The accuracy of learning and validation increases with the number iteration, this reflects that in every iteration the model learns more information. If the precision is decreased then we will need more information to teach our model and therefore we must increase the number of epochs .Likewise, the learning and validation error decreases with the number of epochs.

Likewise, the learning and validation error decreases with the number of epochs.

We got an error rate of 14.76% in train and 17.86% in validation ,and an accuracy rate of 96.12% in train and 95.15% in validation.

Compared to the model 1 and 2,Both accuracy and validation increased ,as well the error rate decreases ,this is due to add layers to the model, and the decrease in the number of batch size of images entering the neurons (32 images instrad of 164 images in first model and 64 in the second model) and increasing the large of data with data augmantation from keras. The confusion matrix allows us to assess the performance of our model, since it reflects the metrics of True Positive, True Negative, False Positive and False Negative. Figure IV.22 closely illustrates the position of these metrics for each class.



	0	1	2	3	4	5	6	7	8	9
0	11175	32	19	25	16	7	45	10	12	23
1	27	8571	19	9	8	3	32	14	17	10
2	56	23	6739	6	94	4	12	16	26	5
3	69	18	8	5838	6	6	0	9	8	12
4	17	7	47	7	5382	14	1	16	25	1
5	22	4	12	8	64	4367	1	60	8	34
6	39	22	9	3	5	2	4390	3	4	4
7	27	12	24	10	6	20	5	3872	33	24
8	20	12	9	5	13	2	7	6	3653	17
9	24	7	5	1	2	4	5	5	14	3874

Figure IV.22: Confusion matrix for model 3

Results comparison table

Models	accuracy	Error	Validation accuracy	Validation error	epoch
Model 1	94.51%	22.63%	93.66%	22.30%	50
Model 2	94.63%	17.03%	94.17%	24.84%	50
Model 3	96.12%	14.76%	95.15%	17.86%	70

Table IV.2: Comparaison between our models

The table IV.2 shows the architecture used in each model as well as the number of epochs. The results obtained are expressed in terms of learning accuracy and error ,Model 3 showed the best results found.The number of epochs and convolution layers and batch size reflect these good results. In general, a large and deep convolutional neural network gives good results and our network performance degrades if a any layer is removed.So, depth is essential to achieve good results. The results obtained improved as we deepened our network and increased the number of epochs with increses batch size entering into each neuron as shown in result of table IV.2 in model 3 . The learning base is also a determining element in the networks of convolutional neurons, it is necessary to have a large learning base to achieve best results.

Conclusion

In this chapter, we have presented a classification approach based on convolutional neural networks, for this we have used three models with different architectures and we have shown the different results obtained in terms of accuracy and error. The comparison of the results found has showed that the number of epochs, the size of the base and the depth of networks are factors important for best results.

General conclusion

In this thesis we talked through the concepts of Artificial Intelligence with Machine Learning in general and , Deep Learning with Artificial neural networks and image processing in particular. We also have introduced convolutional neural networks one of most popular type of ANNs by presenting the different types of layers used in the classification: the convolutional layer, the pooling and the fully connected layer , . and took a some pre-traind CNN . We also talked about the methods of regularization (dropout and data augmentation) used to avoid the problem of over learning.

The network parameters are difficult to define only after many, many experiments to gain enough experience for it . This is why we have defined different models with different architectures in order to obtain the best results in terms of accuracy and error.

Increasing epoch , batch size of image , deeper CNN and Huge information , all of these are a reason for best model and accuracy , this is leads us to came up against several hurdles in the implementation phase ,The time of execution is too expensive. due to the large dimension of the base which requires the use of a GPU instead of a CPU.

We used the GPU of Google colab , Unfortunately, We weren't so lucky, after all Google Colab gives us a limited GPU (12GB NVIDIA Tesla K80 GPU that can be used up to 12 hours) , with Huge amount of data , sessions begins crashed, then restarts and then crashes again , to solve that we need to increase the limitation of GPU, , for that we need to pay for increasing.

Instead of ,we have solved with reducing data , but the essential element of deep learning is Huge amount of data to gain a optimal accuracy. In addition to this, Internet streaming is extremely slow and choppy during work .

In the future, I hope that the student or researcher will be enhanced with better techniques , tools and super machines to reach the best results and motivate them to do the best

Bibliography

[1] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011.

[2] Ian Goodfellow, David Warde-Farley, Mehdi Mirza, Aaron Courville, and Yoshua Bengio. Maxout networks. In *International conference on machine learning*, pages 1319–1327. PMLR, 2013.

[3] Alireza Makhzani and Brendan Frey. Winner-take-all autoencoders. *arXiv preprint arXiv:1409.2752*, 2014.

[4] Wolfgang Ertel. *Introduction to artificial intelligence*, pages 1–3. Springer, 2018.

[5] Andreas C Müller and Sarah Guido. *Introduction to machine learning with Python: a guide for data scientists*, page 25. " O'Reilly Media, Inc.", 2016.

[6] Rudolph Russell. *Machine Learning: Step-by-Step Guide To Implement Machine Learning Algorithms with Python*, page 46. 2018.

[7] Aurélien Géron. *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems.* O'Reilly Media, 2019.

[8] Francois Chollet et al. *Deep learning with Python*, volume 361, pages 39–54. Manning New York, 2018.

[9] Charu C Aggarwal et al. Neural networks and deep learning. *Springer*, 10:3–25, 2018.

[10] Michael Negnevitsky and Artificial Intelligence. A guide to intelligent systems. *Artificial Intelligence, 2nd edition, pearson Education*, pages 170–387, 2005.

[11] Kevin Gurney. *An introduction to neural networks.* CRC press, 1997.

[12] Larry Medsker and Lakhmi C Jain. *Applied Deep Learning*, pages 114–347. CRC press, 1999.

[13] Larry Medsker and Lakhmi C Jain. *Recurrent neural networks: design and applications*, pages 13–324. CRC press, 1999.

[14] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. 'MIT press Cambridge', 2016.

[15] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012.

[16] Mark A Kramer. Nonlinear principal component analysis using autoassociative neural networks. *AIChE journal*, 37(2):233–243, 1991.

[17] Pierre Sermanet, Soumith Chintala, and Yann LeCun. Convolutional neural networks applied to house numbers digit classification. In *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*, pages 3288–3291. IEEE, 2012.

[18] Matthew D Zeiler and Rob Fergus. Stochastic pooling for regularization of deep convolutional neural networks. *arXiv preprint arXiv:1301.3557*, 2013.

[19] Nitish Srivastava. Improving neural networks with dropout. *University of Toronto*, 182(566), 2013.

[20] Diederik P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. In *Advances in neural information processing systems*, pages 3581–3589, 2014.

[21] Ming Liang and Xiaolin Hu. Recurrent convolutional neural network for object recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3367–3375, 2015.

[22] Springenberg and Riedmiller. Improving deep neural networks with probabilistic maxout units. *In International Conference on Learning Representations (ICLR)*, 2014.

[23] Q. Chen M. Lin and S. Yan. Network in network. Springer, 2014.

[24] Honglak Lee, Roger Grosse, Rajesh Ranganath, and Andrew Y Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the 26th annual international conference on machine learning*, pages 609–616, 2009.

[25] J. Ibarz S. Arnoud I. J. Goodfellow, Y. Bulatov and V. Shet. Multi-digit number recognition from street view imagery using deep convolutional neural networks. *In International Conference on Learning Representations (ICLR)*, 2014.

[26] Li Wan, Matthew Zeiler, Sixin Zhang, Yann Le Cun, and Rob Fergus. Regularization of neural networks using dropconnect. In *International conference on machine learning*, pages 1058–1066. PMLR, 2013.

[27] Chen-Yu Lee, Patrick W Gallagher, and Zhuowen Tu. Generalizing pooling functions in convolutional neural networks: Mixed, gated, and tree. In *Artificial intelligence and statistics*, pages 464–472. PMLR, 2016.

[28] The python tutorial python 3.8.6rc1 documentation. `https://docs.python.org/3/tutorial/`. Accessed: 2 June 2021.

[29] Colaboratory google. `https://research.google.com/colaboratory/faq.html#resource-limits`. Accessed: 2 June 2021.

[30] Anaconda navigator anaconda documentation. `https://docs.anaconda.com/anaconda/navigator/`. Accessed: 2 June 2021.

[31] Project jupyter. `https://jupyter.org/`. Accessed: 2 June 2021.

[32] Matlab. `https://www.mathworks.com/products/matlab.html`. Accessed: 2 July 2021.

[33] Tensorflow. `https://www.tensorflow.org/`. Accessed: 2 June 2021.

[34] K. Team. Keras documentation: About keras. `https://keras.io/about/`. Accessed: 2 June 2021.

[35] Opencv. `https://opencv.org/about/`. Accessed: 2 June 2021.

[36] Meet django. `https://www.djangoproject.com/`. Accessed: 23 June 2021.

[37] The street view house numbers (svhn) dataset. `http://ufldl.stanford.edu/housenumbers/`. Accessed: 2 June 2021.

[38] Svhn classification with cnn. `https://www.kaggle.com/dimitriosroussis/svhn-classification-with-cnn-keras-96-acc`. Accessed: 3 March 2021.