i

***Dedicated to***

*Work is dedicated to my Head of the Department, Project*

*Co-ordinator as well as Project Faculty, for their encouragement and support of project*

**Rajiv Gandhi University of Knowledge and Technologies**

**Basar,Nirmal,Telangana,INDIA.**

**DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING**

**CERTIFICATE**

This is to certify that the **Major Project report** entitled “***Current-based Data-retention-Characterization of GainCell embedded DRAMs across Design and Variations Space”*** being submitted to the Rajiv Gandhi University of Knowledge Technologies, Basar by **Mr. SAMA JAYAPRAKASH (ID: B151236** is work done by him and submitted during 2020 –2021 academic year, in partial fulfillment of the requirements for the award of the degree of BACHELOR OF TECHNOLOGY in ELECTRONICS AND COMMUNICATION ENGINEERING.

Mr. Chintam Shravan, Mr. V S S K R Naganjaneyulu Gudapati,

Major Project Coordinator, Head of the Department,

Asst. Professor in ECE, Asst. Professor in ECE,

Rajiv Gandhi University Rajiv Gandhi University

of Knowledge Technologies, of Knowledge Technologies,

Basar. Basar.

**CERTIFICATE OF EXAMINATION**

This is to certify that I have examined the thesis entitled ***” Current-based Data-retention-Characterization of GainCell embedded DRAMs across Design and Variations Space***” submitted by **Mr. SAMA JAYAPRAKASH (ID: B151236)** and hereby accord my approval of it as a study carried out and presented in a manner required for its acceptance in partial fulfillment for the award of the graduate degree for which it has been submitted. This approval does not necessarily endorse or accept every statement made, opinion expressed or conclusion drawn as recorded in the thesis. It only signifies the acceptance of the thesis for the purpose for which it has been submitted.

External Examiner

Date: 05th April, 2021

Place: Basar

**DECLARATION**

I hereby declare that the work embodied in this thesis has been carried out by me under the supervision of Mr. Chintam Shravan in the department of Electronics and Communication Engineering, Rajiv Gandhi University of Knowledge Technologies, Basar and has not been submitted to any other University. Information derived from the published and unpublished work of others has been acknowledged and a list of references is given.

Place : Basar

Sama Jayaprakash - B151236

**ABSTRACT**

Due to the technological advancements and research in memory cells, scientists are looking for low power, on chip memories with higher storage capacity and smaller size that can be fabricated on a single chip. The demand for this memory cells have been increasing day by day due to the advent of huge data generated by the digital world today. Everything can be seemed as a data. This data has to be stored, processed for getting relative information for the specific applications. Big data applications use memory a lot for storing data in the data centres where a huge hardware is required for storing them and their maintenace is another big challenge for tech companies. In this paper, GainCell embedded DRAM’s data retention time is calculated. The present topic of memory cell is GCeDRAM because it provides high density, low leakage currents which is the requirement of today’s world. But there is a limitation to this cell that it needs to be refreshed every time period called refresh period otherwise it may lost the data. The time upto which the DRAM cell can hold the data after a write operation is done until it is read is called as the Data retention time. The calculation of DRT is important because it can provide the details of when to refresh the cell inorder to avoid the data loss. The existing technologies are not efficient for calculating it accurately within the feasible times. Without using highly costliest software licenses, this current based data retention characterization methodology provides way for calculating DRT with 4% error rate within feasible run time.

**ACKNOWLEDGMENTS**

First we Would like to thank management of RGUKT-Basar for giving me

the opportunity to do Major Project.

We also would like to thank all the people who worked along with us

RGUKT-Basar, with their patience and openness they created an enjoyable

working environment.

It is indeed with a great sense of pleasure and immense of gratitude that we

acknowledge the help of these individuals.

We are highly indebted to the Vice-Chancellor and Administrative Officer, for the facilities provided to accomplish this internship.

We would like to thank my Head of the Department , ECE for his

constructive criticism throughout my internship.

We would like to thank our department Internship Coordinator, Mr. Chintam Shravan, Assistant Professor, ECE for their support and advice to get and complete internship in the above said organization.

We are extremely grateful to my department staff members, family members and friends who helped me in successful completion of this internship.

**CONTENTS**

Dedication i

Certificate ii

Certificate of Examination iii

Declaration iv

Abstract v

Acknowledgements vi

Contents vii

List of Figures ix

Abbreviations x

1. Chapter I - Introduction 11

1.1 Brief Information 11

1.2 Related work 12

1.3 Motivation and objectives 13

1.4 Overview 13

1.5 Mathematical notations 14

1.6 Nomenclature 14

2. Chapter II - Face Recognition in general 15

2.1 History of face recognition 15

2.2 Face recognition tasks 15

2.3 The process of face recognition 15

2.3.1 Face detection 16

2.3.2 Pre processing 16

2.3.3 Feature Extraction 16

2.3.4 Feature Matching 16

3. Chapter III - Assessment 17

3.1 General aspects of face detections algorithms 17

4. Chapter IV - Analysis 19

4.1 Model structure 19

4.2 Triplet loss and Selection 20

4.3 Deep learning basics 22

4.3.1 Stochastic Gradient Descent 23

4.3.2 AdaGrad 24

4.3.3 Relu 25

4.4 CNN architecture 26

4.4.1 Zeiler & Fergus architecture 26

4.4.2 Inception model 26

4.5 Datasets and Evaluation 27

4.5.1 Embedding Dimensionality 27

5. Chapter VI - System Implementation 31

5.1 Front End Implementation 31

5.2 Python Introduction 32

5.3 Anaconda 35

5.3.1 Anaconda Installation 36

5.4 Convolutional Neural Networks 42

5.4.1 Tensorflow 44

5.4.2 Keras 45

5.5 Coding of FACENET model using Keras 46

5.6 Application of Facial Recognition 52

5.7 Scope of Facial Recognition technology in India 53

6. Future 54

7. Conclusion 55

8. References 56

**List of figures**

1 The Process of Face Recognition…....………………………………..16

2 Pixel image…………………………………………………………....17

3 Pixel image hidden layers…………………………………………….18

4 Model structure……………………………………………………….19

5 Embeddings of image………………………………………………...20

6 Triplet image………………………………………………………….21

7 Gradient descent image……………………………………………….23

8 Relu graph…………………………………………………………….25

9 Zeiler & Fergus architecture………………………………………….26

10 Inception model image……………………………………………….27

11 Inception model architecture…………………………………………28

12 Youtube Face database sample……………………………………….30

13-23 Python and Anaconda Installation………………………………..34-42

24 Mapping of neuron layers……………………………………………43

**Abbreviations**

**Abbreviation Description**

(eDRAM) Embedded dynamic random access memory

GC’s Gain Cells

LPV Low Power Variations

GPV Global Power Variations

MC Monte-Carlo

**Chapter - I**

**Current based Data Retention Characterization methodology**

1. **Introduction:** The use of data-intensive applications which used more power and silicon area are now dominated by embedded memories. The first memory cell we discuss here is SRAM.

**SRAM:** SRAM is usually builtin CMOS Technology with six transistors. Two cross-coupled inverters are used to store the information like in a flip-flop. For the access control two further transistors are needed. If the Write Line is enabled then data can be read and set with the Bit-Lines.

**States of an SRAM-Cell:**

**•Standby:** Write Line is disabled, no reading and writing is possible

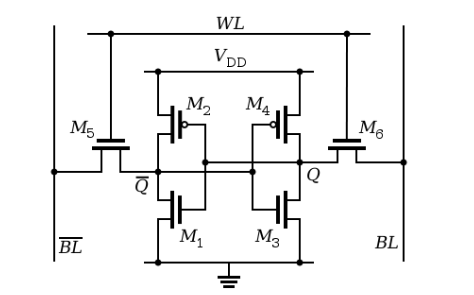
•**Reading:** Reading starts with preloading the Bit lines to 1. Now the Write line gets activated. If Q is 1 then BL gets pulled to one and 𝐵𝐿bar towards zero. A sense amplifier senses which line has the higher voltage. If capacitor c2 is discharging, bitbar line will have a decreasing voltage and if the bit and bit bar lines voltages have given to the sense amplifier, it outputs 1 and vice versa.

•**Writing:** (Consider Q=0 and Qbar=1). Setting the Bit line to 1 (BL = 1) needs to pull down the bit bar line to ground. As it is pulled to ground, the transistor M1 will be off if the voltage of bit line bar becomes less than threshold voltage and M2 will be on which makes it to have 1 and vice versa.

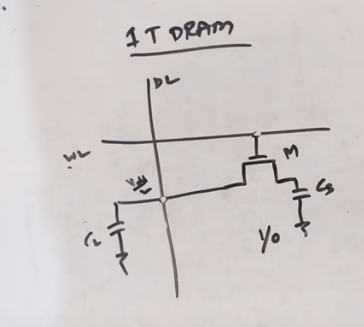
**Advantages:** Quick, easy to control, integrated in the chip -> fast because no bus is needed like in DRAM

**Disadvantages:** Many transistors are needed -> expensive, higher power consumption than DRAM.

In read operation, if the stored value is 0, the c1 capacitor discharges and if voltage is greater than threshold voltage of M3 then M3 is ON outputting the Qbar=0 and it is contradicting state because Q=0 already.



**DRAM :**

****

1. The level of charge on the memory cell capacitor determines whether that particular bit is a logical "1" or "0" - the presence of charge in the capacitor indicates a logic "1" and the absence of charge indicates a logical "0".
2. The basic dynamic RAM memory cell has the format that is shown below. It is very simple and as a result it can be densely packed on a silicon chip and this makes it very cheap.
3. Two lines are connected to each dynamic RAM cell - the Word Line (W/L) and the Bit Line (B/L) connect as shown so that the required cell within a matrix can have data read or written to it.

**READ operation:** The word line is asserted and it becomes the transistor to become ON and whatever the data present in the capacitor gets shared with the bit line. Lets say if capacitor is stored with logic 1, then the effective voltage becomes greater than than vdd and the sense amplifier outputs as 1. On the other hand, if the capacitor stores a logic 0 value, then capacitor voltage looses and if it sensed by sense amplifier , it outputs as a 0.

**Write Operation :** In write operation, the data/bit lines acts as inputs. The word line is asserted and transistor becomes ON and whatever data is present on the data line it is shared to capacitor.

This DRAM is not preferable because of its destructive read operation. It means for a read operation, the stored value gets lost after reading.

We cannot fabricate capacitor on chip so it is not preferred in IC designs.

1 T-1C eDRAM requires dedicated process steps to fabricate the cell capacitor resulting in additional complexity and cost, and it suffers from a destructive read operation, which increases power consumption.

GC-eDRAM offers higher density due to its lower transistor count, low-leakage, and inherent two-ported operation. Compared to 1 T-1C eDRAM, it is fully logic-compatible, thus it does not require additional process steps, and its read operation is nondestructive.

**1.1 Brief Information:**

Facial recognition is one of the most exciting applications of deep learning. The rise in the adoption of facial recognition systems has been phenomenal in recent years. The threats like data leakage, privacy violation etc. , originating from careless use of facial recognition systems are pretty real, and hence proper measures should be taken to avoid them. FaceNet provides a unique architecture for performing tasks like face recognition, verification and clustering. It uses deep convolutional networks along with triplet loss to achieve state of the art accuracy.

FaceNet provides a unified embedding for face recognition, verification and clustering tasks. It maps each face image into a euclidean space such that the distances in that space correspond to face similarity, i.e. an image of person A will be placed closer to all the other images of person A as compared to images of any other person present in the dataset.

The main difference between FaceNet and other techniques is that it learns the mapping from the images and creates embeddings rather than using any bottleneck layer for recognition or verification tasks. Once the embeddings are created all the other tasks like verification, recognition etc. can be performed using standard techniques of that particular domain, using these newly generated embeddings as the feature vector. For example we can use k-NN for face recognition by using embeddings as the feature vector and similarly we can use any clustering technique for clustering the faces together and for verification we just need to define a threshold value.

**1.2 Related work:**

Similar to other recent works which employ deep net-works, this approach is a purely data driven method which learns its representation directly from the pixels of the face. Rather than using engineered features, we use a large dataset of labelled faces to attain the appropriate in-variances to pose, illumination, and other variational conditions.

We explore two different deep network architectures that have been recently used to great success in the computer vision community. Both are deep convolutional networks . The first architecture is based on the Zeiler & Fergus model which consists of multiple interleaved layers of convolutions, non-linear activations, local response normalizations, and max pooling layers. The second architecture is based on the Inception model of Szegedyet al. which was recently used as the winning approach for ImageNet 2014. These networks use mixed layers that run several different convolutional and pooling layers in parallel and concatenate their responses. It is found that these models can reduce the number of parameters by up to 20 times and have the potential to reduce the number of FLOPS required for comparable performance.

**1.3 Motivation and Objectives:**

Face recognition has recently received a blooming attention and interest from the scientific community as well as from the general public. The interest from the general public is mostly due to the recent events of terror around the world, which has increased the demand for useful security systems. Facial recognition applications are far from limited to security systems. To construct these different applications, precise and robust automated facial recognition methods and techniques are needed. However, these techniques and methods are currently not available or only available in highly complex, expensive setups. The objectives of this report will be:

* To discuss and summarize the process of facial recognition.
* To look at currently available facial recognition techniques.
* To analyze and apply the FaceNet model in our project.

**1.4 Overview:**

In the fulfilment with the objectives this report is divided into five parts, where each part requires knowledge from the preceding parts.

**Part I** : **Face Recognition in General.** Presents a summary of the history of face recognition. Discusses the different commercial face recognition systems, the general face recognition process and the different considerations regarding facial recognition.

**Part II : Assessment**. Presents an assessment of the central tasks of face recognition identified in Part I, which include face detection, preprocessing of facial images and feature extracting.

**Part III : Analysis.** Analysing the design and development of facenet.

**Part IV : Coding of FaceNet**. Presents a discussion of possible ideas to future work and concludes on the work done in this report.

**1.5 Mathematical Notation:**

Throughout this report the following mathematical notations are used:

Scalar values are denoted with lower-case italic Latin or Greek letters: x

Vectors are denoted with lower-case, non-italic bold Latin or Greek letters.

Matrices are denoted with capital, non-italic bold Latin or Greek letters:

X=[ a b

c d ]

Sets of objects such as scalars, vectors, images etc. are shown in vectors with curly braces:

{a, b, c, d}

Indexing into a matrix is displayed, as row-column subscript of either scalars or vectors:

Mxy=Mx,x= [x, y]

The mean vector of a specific data set, is denoted with lower-case, non-italic bold Latin or Greek letters with a bar: ̄x

**1.6 Nomenclature:**

Xi A sample vector in the input space.

Yi A sample vector in the output space.

Φ An eigenvector matrix.

Φi The ith eigenvector.

Λ A diagonal matrix of eigenvalues.

Λi The eigenvalue corresponding to the ith eigenvector.

**Chapter II**

**Face Recognition in General**

Part - I

**2.1 History of face recognition:**

Most of the developed techniques during the first stages of facial recognition focused on the automatic detection of individual facial features. The greatest advantages of these geometrical feature-based methods are the insensitivity to illumination and the intuitive understanding of the extracted features. However, even today facial feature detection and measurement techniques are not reliable enough for the geometric feature-based recognition of a face and geometric properties alone are inadequate for face recognition.

Due to this drawback of geometric feature-based recognition, the technique has gradually been abandoned and an effort has been made in researching holistic color-based techniques, which has provided better results. Holistic color-based techniques align a set of different faces to obtain a correspondence between pixel intensities, a nearest neighbor classifier can be used to classify new faces when the new image is first aligned to the set of already aligned images.

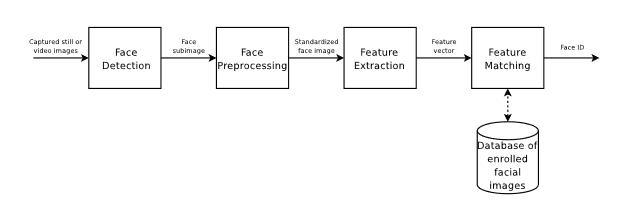
**2.2 Face Recognition Tasks:**

The three primary face recognition tasks are:

* Verification (authentication) - Am I who I say I am? (one to one search)
* Identification (recognition) - Who am I? (one to many search)
* Watch list - Are you looking for me? (one to few search)

Different schemes are to be applied to test the three tasks described above. Which scheme to use depends on the nature of the application.

**2.3 The Process of Face Recognition:**



**Figure -1**

**2.3.1 Face detection:**

The aim of face detection is localization of the face in an image. In the case of video input.

**2.3.2 Preprocessing:**

The aim of the face preprocessing step is to normalize the coarse face detection, so that a robust feature extraction can be achieved. Depending on the application, face preprocessing includes: Alignment (translation, rotation, scaling) and light normalization/correlation.

**2.3.3 Feature Extraction:**

The aim of feature extraction is to extract a compact set of interpersonal discriminating geometrical or/and photometrical features of the face. Methods for feature extraction include: PCA, FLDA and Locality Preserving Projections(LPP).

**2.3.4 Feature Matching:**

Feature matching is the actual recognition process. The feature vector obtained from the feature extraction is matched to classes (persons) of facial images already enrolled in a database. The matching algorithms vary from the fairly obvious Nearest Neighbor to advanced schemes like Neural Networks.

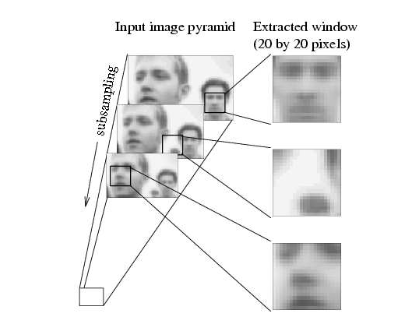
**Chapter - III**

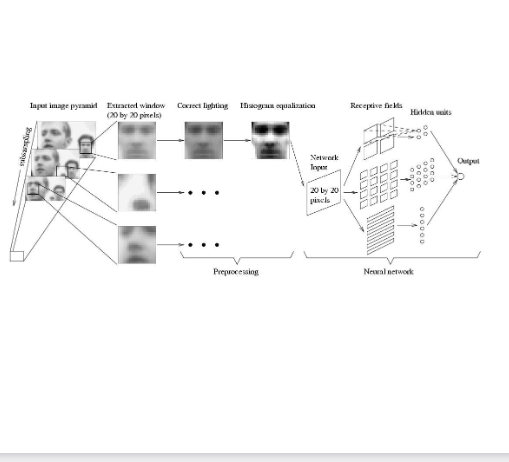
**Assessment**

Part -II

**3.1 General Aspects of Face Detections Algorithms:**

Most face detection algorithms work by systematically analyzing subregions of an image. An example of how to extract these subregions could be, to capture a sub image of 20×20 pixels in the top left corner of the original image and continuing to capture subimages in a predefined grid.

**Figure -2** 

****

**Figure -3**

All these subimages are then evaluated using a face detection algorithm. Subsampling of the image in a pyramid fashion enables capture of different sizes of face.

**Chapter -IV**

**ANALYSIS**

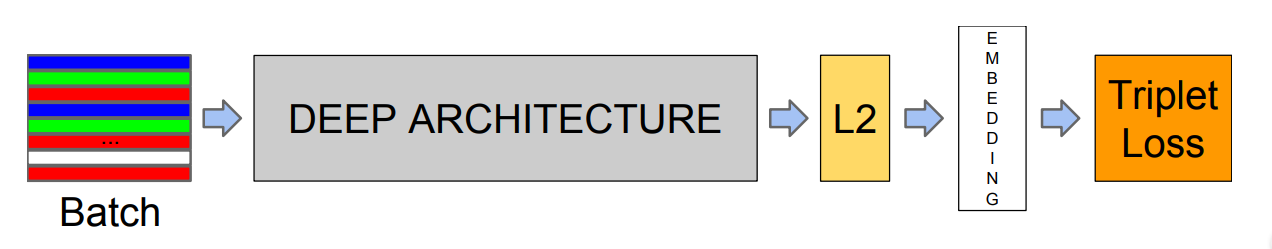
**PART - III**

Google has presented a paper, called FaceNet, that directly learns a mapping from face images to a compact Euclidean space where distances directly correspond to a measure of face similarity. Once this space has been produced, tasks such as face recognition, verification and clustering can be easily implemented using standard techniques with FaceNet embeddings as feature vectors. The method uses a deep convolutional network trained to directly optimize the embedding itself, rather than an intermediate bottleneck layer as in previous deep learning approaches. To train, we use triplets of roughly aligned matching / non-matching face patches generated using a novel online triplet mining method.

The benefit of this approach is much greater representational efficiency: we achieve state-of-the-art face recognition performance using only 128-bytes per face. On the widely used Labeled Faces in the Wild (LFW)dataset, this system achieves a new record accuracy of 99.63%. On YouTube Faces DB it achieves 95.12%. Our system cuts the error rate in comparison to the best published result by 30% on both datasets.

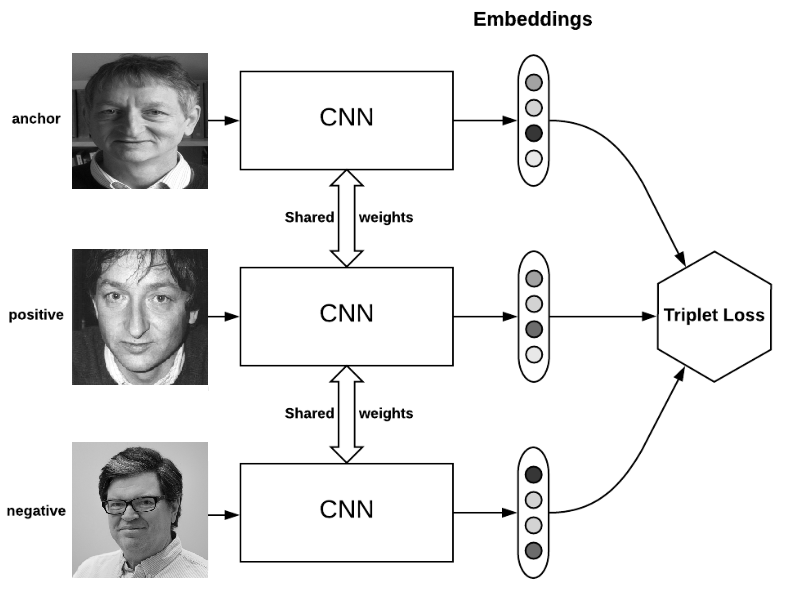
**4.1 Model structure:**

This network consists of a batch input layer and a deep CNN followed by L2 normalization, which results in the face embedding. This is followed by the triplet loss during training.



**Figure - 4**

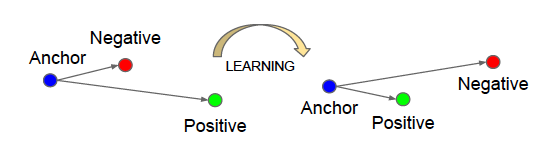
One of the important aspects of FaceNet is its loss function . It uses triplet loss function. In order to calculate the triplet loss, we need 3 images namely anchor, positive and negative. We will explore triplet loss in great detail in the next section.



**Figure - 5**

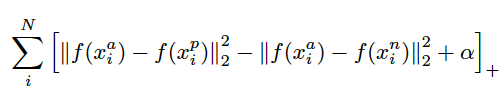
**4.2 Triplet Loss and Selection:**

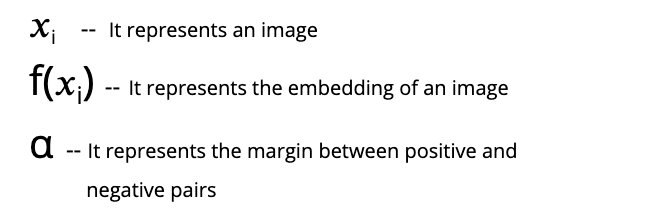
The intuition behind triplet loss function is that we want our anchor image (image of a specific person A) to be closer to positive images (all the images of person A) as compared to negative images (all the other images).



**Figure - 6**

In other words, we can say that we want the distances between the embedding of our anchor image and the embeddings of our positive images to be lesser as compared to the distances between embedding of our anchor image and embeddings of our negative images.

Triplet loss function can be formally defined as —



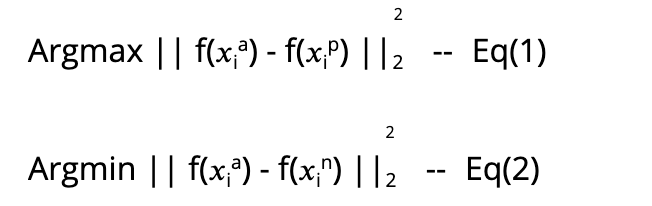
Here, the superscript a, p and n correspond to anchor, positive and negative images respectively.

Alpha is defined here as the margin between positive and negative pairs. It is essentially a threshold value which determines the difference between our image pairs. If let’s say alpha is set to 0.5, then we want the difference between our anchor-positive and anchor-negative image pairs to be at least 0.5.pairs that will satisfy this condition and hence our model won’t learn much from them and will also converge slowly because of that.

Choosing the correct image pairs is extremely important as there will be a lot of image

In order to ensure fast convergence, it is crucial to select triplets that violate the triplet constraint.

We essentially want to select the following —



Eq (1) means that given an anchor image of person A, we want to find a positive image of A such that the distance between those two images is largest.

Eq (2) means that given an anchor image of person A, we want to find a negative image such that the distance between those two images is smallest.

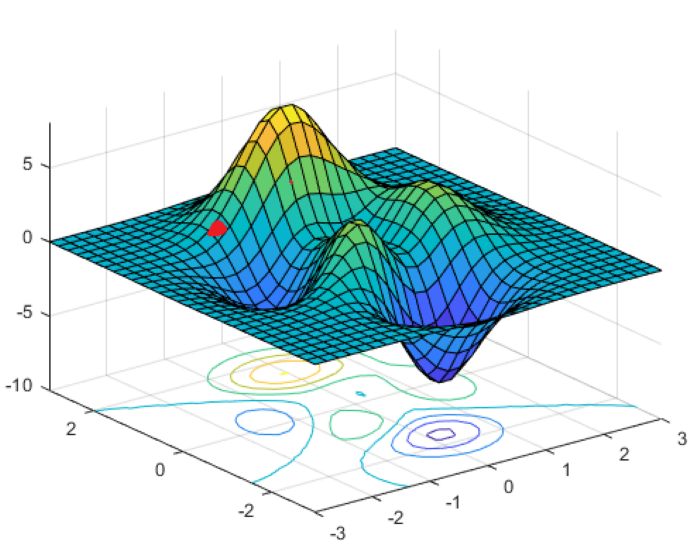
So, we are just selecting the **hard positives** and **hard negatives** here. This approach helps us in speeding convergence as our model learns useful representations. But there is a problem associated with this approach, it is computationally infeasible to compute hard positives and hard negatives over the entire dataset.

**4.3 Deep Learning Basics :**

FaceNet trains CNNs using **Stochastic Gradient Descent** (SGD) with standard backprop and **AdaGrad**. The initial learning rate is 0.05, alpha is set to 0.2 and **ReLU** ischosen as the activation function.

**4.3.1 Stochastic Gradient Descent:**

It is an optimisation technique that is used to optimise our loss function.



**Figure -7**

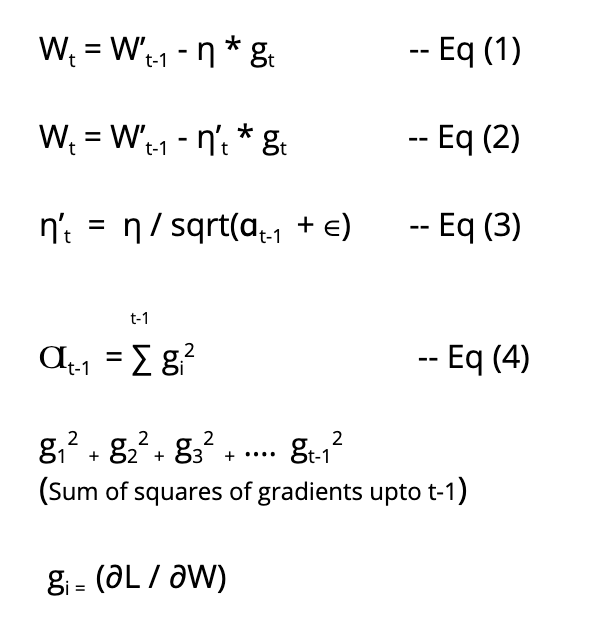
The two axes (x and y) represent weights and the third axis (z) represents the loss with respect to those two weights.

Let’s call this red point as point A. We will start our journey from this point A. The intuition behind SGD is that we want to traverse this hill-like structure in such a way that we reach the global minima (lowest point of this hill). Now you might have understood the **Descent** part of SGD. So now let’s focus on the **Gradient** part.

Gradient just gives us the direction of the **steepest ascent** in a n- dimensional plane (similar to a derivative, which determines the slope of a line ).

**4.3.2 AdaGrad:**

AdaGrad is used to generate variable learning rates. Fixed learning rates do not work well in deep learning. In case of CNNs where each layer is used to detect a different feature (edges, patterns etc.), a fixed learning will just not work, as different layers in our network require different learning rates to work optimally. To better understand AdaGrad, let’s look at a few equations.



Eq (1) — It is just the regular weight update equation of SGD. Here we are using a fixed learning rate (η).

Eq (2) — It is the weight update equation of AdaGrad. In this case we are using a variable learning rate (η’t).

Eq (3) — It determines the formula for calculating the variable learning rate.

Eq (4) — It determines the formula for calculating Ɑt-1.

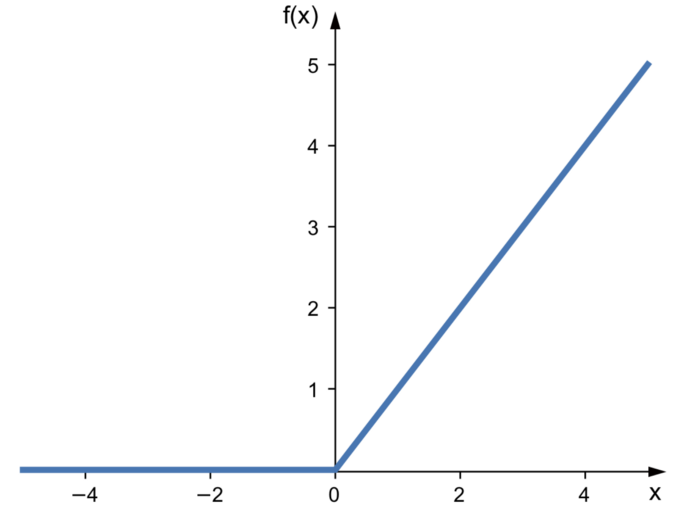
Ɑt-1 is just the sum of squares of gradients upto t-1. ‘t’ is the iteration number. So we just calculate the gradient at each step and add their squares together to generate Ɑt-1 and since Ɑt-1 will change with every iteration, our learning rate will also change.

**4.3.2 ReLU:**

ReLU is the non-linear activation function that we are using. Before diving into details about ReLU, let’s understand why we need non-linear activation functions? We need them as if we use only a linear activation function, then essentially our output will just be a linear combination of our input regardless of the number of layers in our network.

Another point to consider is that, without non-linear activation functions, we can’t create neural networks which can solve intricate problems i.e. our decision boundary will always be linear if we use linear activation functions.

ReLU is the successor of sigmoid and tanh activation functions. I won’t go into much details here about these functions, but I will briefly explain the issues in sigmoid and tanh that led to the discovery of ReLU. So the big problem with both sigmoid and tanh is that of **vanishing gradients**, i.e. they both output a value between 0 and 1 and while calculating gradients using back propagation (refer to Eq (1) ) we have to multiply various values which lie between 0 and 1. After a few iterations, the value may become so small and insignificant that our weights will stop updating. Another issue with both of them is that they both are expensive to compute i.e. we have to compute functions like exponent and tan, which are computationally expensive.



**Figure - 8**

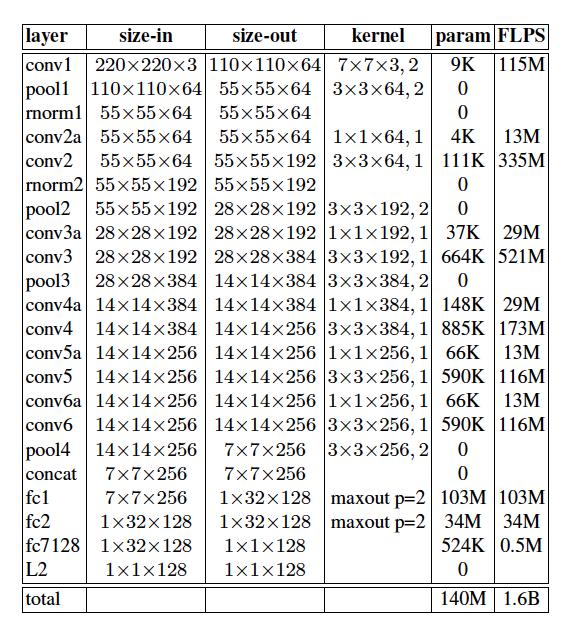
**4.4 CNN Architectures:**

FaceNet uses 2 types of CNNs, namely Zeiler & Fergus architecture and GoogLeNet style Inception model

**4.4.1 Zeiler & Fergus architecture:**

Zeiler & Fergus architecture is used for visualising the training process of a CNN. We try to understand the internal workings of a CNN with the help of this architecture. This architecture introduced a novel visualisation technique that gives insight into the function of intermediate layers and the operation of the classifiers.

The Zeiler & Fergus architecture used in the FaceNet research paper is shown below.

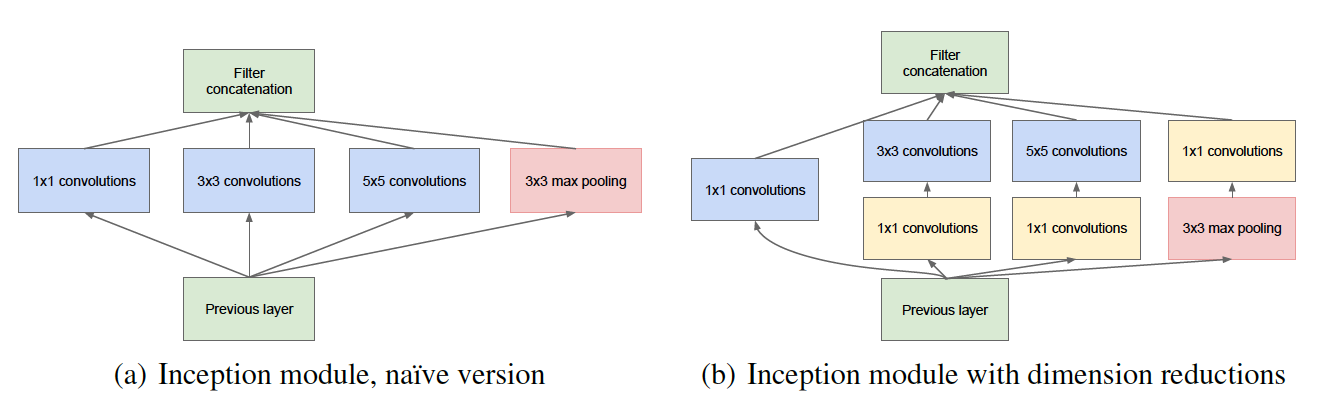


**Figure - 9**

This model has 140 million parameters and 1.6 billion FLOPS (Floating point operations per second) per image

**4.4.2 Inception Model:**

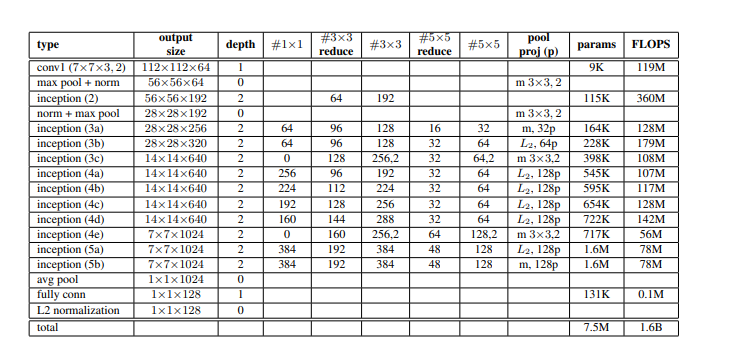
The main idea behind Inception network architecture is that of using multiple filters of different sizes simultaneously. In any other traditional network architecture we usually choose a filter of let’s say size 3\*3 , 5\*5 etc, but in Inception architecture we use multiple filters simultaneously and concatenate their results.



**Figure - 10**

In Fig (a), we are using multiple filters of size 1\*1, 3\*3 and 5\*5 along with a max pooling layer, and then we have concatenated the results. This is the main intuition behind Inception network architecture. The problem with this approach is that it is computationally very expensive. So, in order to avoid this problem we use 1\*1 convolutions for dimensionality reduction.

In Fig (b), we will use a 1\*1 filter with every other convolution in order to reduce dimensionality and make this architecture computationally feasible.



**Figure - 11**

This Inception model architecture used in the FaceNet research paper has 6.6M — 7.5M parameters and around 500M — 1.6 B FLOPS. Various variations of the Inception model are used in FaceNet, some of them are optimised to run on mobile phones and hence have comparatively less parameters and filters.

**4.5 Datasets and Evaluation:**

We calculate the true accepts (TA) as follows —



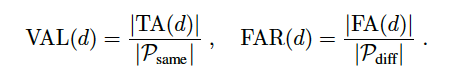
True accepts are the face pairs that were **correctly** classified as the same at threshold ‘d’.

We define the false accepts (FA) as follows —



False accepts are the face pairs that were **incorrectly** classified as the same*.*

The validation rate (VAL) and false accept rate (FAR) for a given face distance ‘d’ is defined as

**

**4.5.1 Embedding Dimensionality:**

We explored various embedding dimensionalities and selected 128 for all experiments.. One would expect the larger embeddings to perform at least as good as the smaller ones, however, it is possible that they require more training to achieve the same accuracy. That said, the differences in the performance are statistically insignificant. It should be noted that during training a 128 dimensional float vector is used, but it can be quantized to 128-bytes without loss of accuracy. Thus each face is compactly represented by a 128 dimensional byte vector, which is ideal for large scale clustering and recognition. Smaller embeddings are possible at a minor loss of accuracy and could be employed on mobile devices.

**Results:**

This model uses 4 different types of architecture on Labeled Faces in the wild and Youtube Face DB dataset. These

* **Labeled Faces in the Wild Dataset:**This architecture uses standard, non-restricted protocol on LFW dataset. First, this model uses 9 training splits to set *L2* distance threshold value and then on the tenth split, it classifies the two images as the same or different*.*  
  There are two methods of preprocessing of the images of dataset on which accuracy is reported:
  + Fixed center crop of the image provided in LFW
  + A face detector is used on LFW images if that fails then LFW face alignment is used

This model achieves a classification accuracy of *98.87%* accuracy with 0.15% standard error and in the second case *99.63%* accuracy with *0.09%* standard error. This reduces the error rate reported by *DeepFace* by a factor of more than *7* and other state-of-the-art DeepId by *30%*.

* **Youtube Face database:**On Youtube Face Dataset it reported an accuracy of *95.12%* with standard error *0.39* using the first 100 frames. It is better than *91.4%* accuracy proposed by DeepFace and *93.5%* reported by DeepId on 100 frames

**Figure - 12**

A result of **Face clustering** (clustering of images of the same person) from FaceNet paper shows that the model is invariant to occlusion, pose, lighting and even age, etc.

**Chapter- V**

**SYSTEM IMPLEMENTATION**

**Part - V**

**5.1 FRONT END IMPLEMENTATION**

**5.1.1 Python Introduction and Installation :**

Python is a general purpose, dynamic, high level, and interpreted programming language. It supports an Object Oriented programming approach to develop applications. It is simple and easy to learn and provides lots of high-level data structures.

Python is easy to learn yet powerful and versatile scripting language, which makes it attractive for Application Development. Python's syntax and dynamic typing with its interpreted nature make it an ideal language for scripting and rapid application development. It supports multiple programming patterns, including object-oriented, imperative, and functional or procedural programming styles. Python is not intended to work in a particular area, such as web programming. That is why it is known as a multipurpose programming language because it can be used with web, enterprise, 3D CAD, etc. We don't need to use data types to declare variables because it is dynamically typed so we can write a=10 to assign an integer value in an integer variable. It makes development and debugging fast because there is no compilation step included in Python development, and the edit-test-debug cycle is very fast.

**Python Applications**

Python is known for its general purpose nature that makes it applicable in almost each domain of software development. Python as a whole can be used in any sphere of development. Here, we are specifying application areas where python can be applied.

**Web Applications:** We can use Python to develop web applications. It provides libraries to handle internet protocols such as HTML and XML, JSON, Email processing, request, beautifulSoup, Feedparser etc. It also provides Frameworks such as Django, Pyramid, Flask etc to design and develop web based applications. Some important developments are: PythonWikiEngines, Pocoo, PythonBlogSoftware etc. The useful library and package are SciPy, Pandas, IPython etc. SciPy is a group of packages of engineering.

**Desktop GUI Applications:** Python provides Tk GUI library to develop user interface in python based applications. Some other useful toolkits wxWidgets, Kivy, pyqt that are usable on several platforms. The Kivy is popular for writing multitouch applications.

**Software Development:** Python is helpful for the software development process. It works as a support language and can be used for build control and management, testing etc.

**Scientific and Numeric:** Python is popular and widely used in scientific and numeric computing. Some useful libraries and packages are SciPy, Pandas, IPython etc. SciPy is a group of packages of engineering, science and mathematics.

**Business Application:** Python is used to build Business applications like ERP and e-commerce systems. Tryton is a high level application platform.

**Console Based Application:** We can use Python to develop console based applications. For example: IPython.

**Audio or Video based Applications:** Python is awesome to perform multiple tasks and can be used to develop multimedia applications. Some of the real applications are: TimPlayer, cplay etc.

**3D CAD Applications:** To create CAD applications Fandango is a real application which provides full features of CAD.

**Enterprise Applications:** Python can be used to create applications which can be used within an Enterprise or an Organization. Some real time applications are: OpenErp, Tryton, Picalo etc.

**Applications for Images:** Using Python several applications can be developed for image. Applications developed are: VPython, Gogh, imgSeek etc. Python's syntax and dynamic typing with its interpreted nature make it an ideal language for scripting and rapid application development. It supports multiple programming patterns, including object-oriented, imperative, and functional or procedural programming styles.

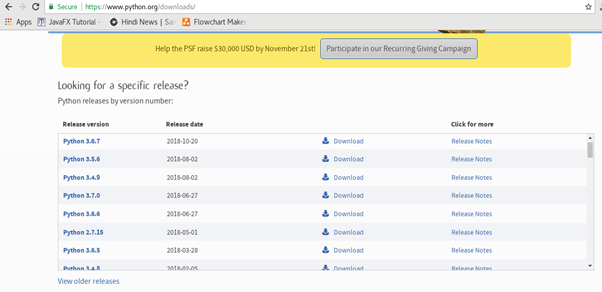
There are several such applications which can be developed using Python

.

**5.2 Installation on Windows:**

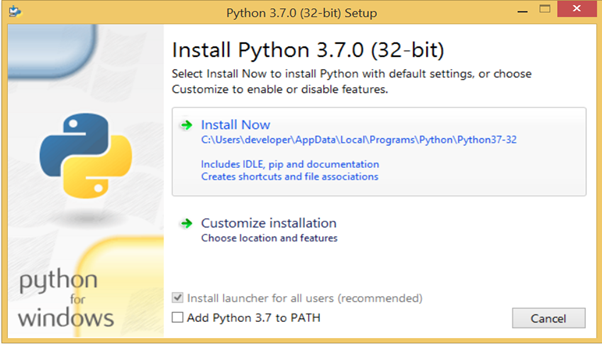
Visit the link [*https://www.python.org/downloads/*](https://www.python.org/downloads/) to download the latest release of Python. In this process, we will install Python 3.6.7 on our Windows operating system.

Double-click the executable file which is downloaded; the following window will open. Select Customize installation and proceed.

****

**Figure -13**

Double-click the executable file which is downloaded; the following window will open. Select Customize installation and proceed.

****

**Figure -14**

The following window shows all the optional features. All the features need to be installed and are checked by default; we need to click next to continue.

**5.3 Anaconda:**

Some tutorials install packages with conda instead of pip. So what is conda?

Anaconda is a distribution of Python (and R). It is free and open-source and makes package management and deployment simpler. Keep reading to see how. It is the standard platform for python data Science and open-source machine learning. Anaconda is used by data scientists, IT professionals and business leaders.

**Benefits of Using Python Anaconda**

Why should you use Anaconda for your project? Well, it does have the following benefits:

1. It is free and open-source.

2) It has more than 1500 Python/R data science packages.

3) Anaconda simplifies package management and deployment.

4) It has tools to easily collect data from sources using machine learning and AI.

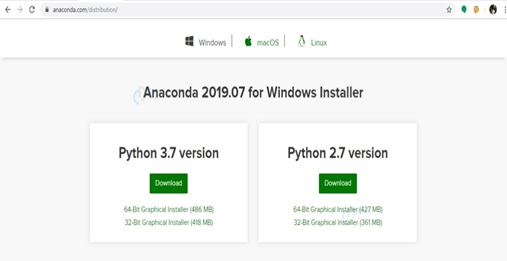
5) It creates an environment that is easily manageable for deploying any project

6) Anaconda is the industry standard for developing, testing and training on a single machineIt has good community support- you can ask your questions there.

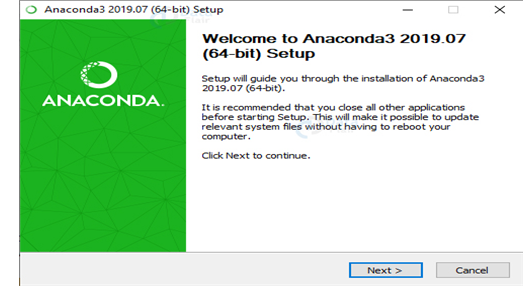
What you get

**5.3.1 Python Anaconda Installation:**

Next in the Python anaconda tutorial is its installation. The latest version of Anaconda at the time of writing is 2019.07. Follow these steps to download and install Anaconda on your machine:

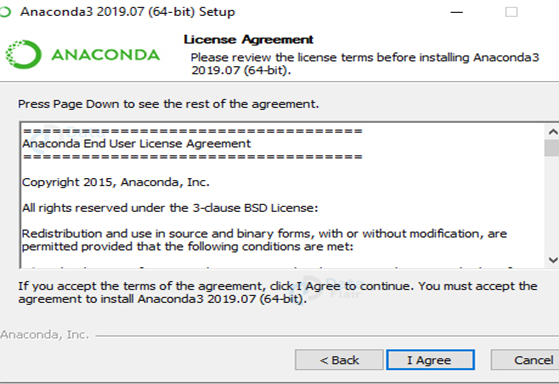
→ Go to this link and download Anaconda for Windows, Mac, or Linux: – [Download anaconda](https://www.anaconda.com/distribution/) **Figure -15** 

You can download the installer for Python 3.7 or for Python 2.7 (at the time of writing). And you can download it for a 32-bit or 64-bit machine.

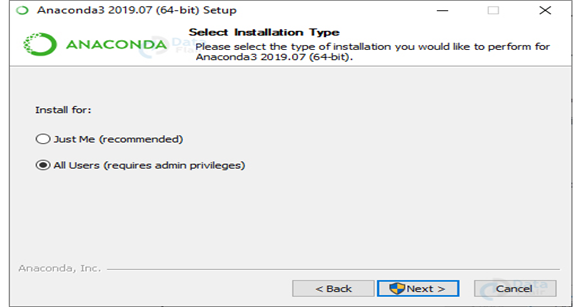
→ Click on the downloaded .exe to open it. This is the Anaconda setup. Click next.

**Figure -16**

Now, you’ll see the license agreement. Click on ‘I Agree’

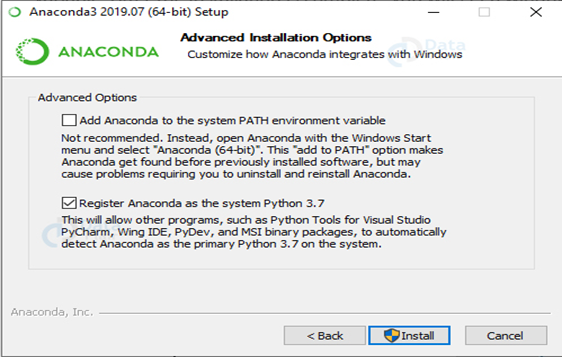


**Figure -17**

→ You can install it for all users or just for yourself. If you want to install it for all users, you need administrator privileges.

**Figure -18**

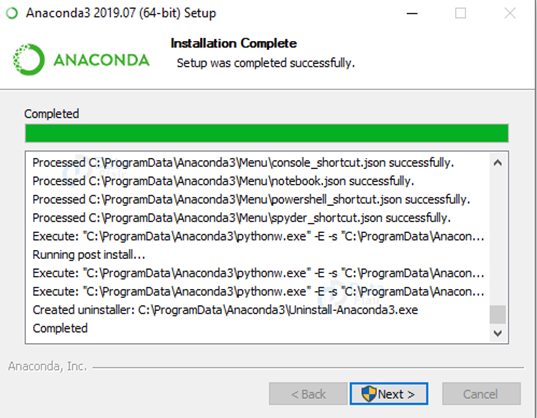
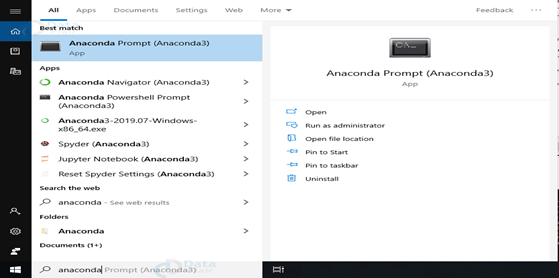
Now, you’ll get some advanced options. You can add Anaconda to your system’s PATH environment variable, and register it as the primary system Python 3.7. If you add it to PATH, it will be found before any other installation. Click on ‘Install’.



**Figure -19**

It will unpack some packages and extract some files on your machine. This will take a few minutes.

The installation is complete. Click Next.

**Figure -20** 

**Figure -21**

If you search for Anaconda now, you will see the following options:

· The Anaconda prompt

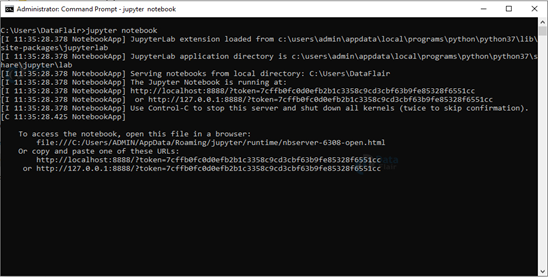
· Anaconda Navigator

· Anaconda Powershell prompt

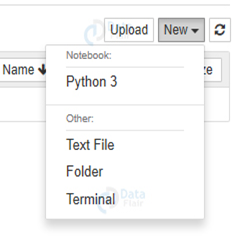
· Spyder IDE and

· Jupyter Notebook

If you go to your command prompt and type ‘jupyter notebook’, it will open the Jupyter dashboard for you.



**Figure -22**



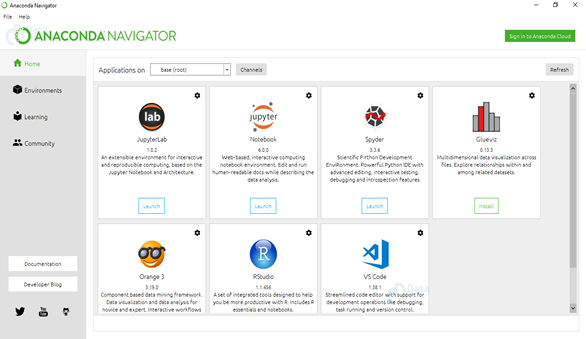
You can create a new notebook from the New menu at the top right.

This is your new notebook. You can type in this. To execute a statement or multiple statements at once, press Shift+Enter. Pressing only Enter will only take you to the next line.

This is your new notebook. You can type in this. To execute a statement or multiple statements at once, press Shift+Enter. Pressing only Enter will only take you to the next line.

**Anaconda Navigator**

Anaconda Navigator is a desktop GUI that ships with Anaconda and lets you launch applications and manage conda packages, environments, and channels without having to use a command-line interface. It can search for packages in a local Anaconda repository or on Anaconda Cloud. With Navigator, you don’t need to type commands in a terminal, it lets you work with packages and environments with just a click.



**Figure -23**

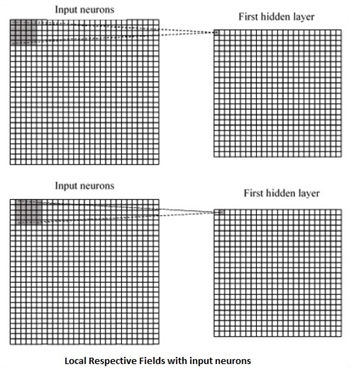
**5.4 Convolutional Neural Networks**

Convolutional Neural networks are designed to process data through multiple layers of arrays. This type of neural networks is used in applications like image Analysis or face recognition. The primary difference between CNN and any other ordinary neural network is that CNN takes input as a two-dimensional array and operates directly on the images rather than focusing on feature extraction which other neural networks focus on.

The dominant approach of CNN includes solutions for problems of recognition. Top companies like Google and Facebook have invested in research and development towards recognition projects to get activities done with greater speed. A convolutional neural network uses three basic ideas : Local respective fields, Convolution, Pooling.

CNN utilizes spatial correlations that exist within the input data. Each concurrent layer of a neural network connects some input neurons. This specific region is called the local receptive field. Local receptive field focuses on the hidden neurons. The hidden neurons process the input data inside the mentioned field not realizing the changes outside the specific boundary.

If we observe the below representation, each connection learns a weight of the hidden neuron with an associated connection with movement from one layer to another. Here, individual neurons perform a shift from time to time. This process is called “convolution”.

The mapping of connections from the input layer to the hidden feature map is defined as “shared weights” and bias included is called “shared bias”.

**Figure -24**

CNN or convolutional neural networks use pooling layers, which are the layers, positioned immediately after CNN declaration. It takes the input from the user as a feature map that comes out of convolutional networks and prepares a condensed feature map. Pooling layers helps in creating layers with neurons of previous layers.

**5.4.1 Tensor flow**

Tensor Flow is a software library or framework, designed by the Google team to implement machine learning and deep learning concepts in the easiest manner. It combines the computational algebra of optimization techniques for easy calculation of many mathematical expressions. Let us now consider the following important features of TensorFlow −

It includes a feature that defines, optimizes and calculates mathematical expressions easily with the help of multi-dimensional arrays called tensors.

It includes a programming support of deep neural networks and machine learning techniques.

It includes a high scalable feature of computation with various data sets.

TensorFlow uses GPU computing, automating management. It also includes a unique feature of optimization of the same memory and the data used.

Tensor Flow is well-documented and includes plenty of machine learning libraries. It offers a few important functionalities and methods for the same.

Tensor Flow is also called a “Google'' product. It includes a variety of machine learning and deep learning algorithms. TensorFlow can train and run deep neural networks for handwritten digit classification, image recognition, word embedding and creation of various sequence models.

**5.4.2 TensorFlow – Keras**

One of the most powerful and easy-to-use Python libraries for developing and evaluating deep learning models is Keras; It wraps the efficient numerical computation libraries Thaana and TensorFlow. The advantage of this is mainly that you can get started with neural networks in an easy and fun way.Keras is compact, easy to learn, high-level Python library run on top of TensorFlow framework. It is made with the focus of understanding deep learning techniques, such as creating layers for neural networks maintaining the concepts of shapes and mathematical details. The creation of framework can be of the following two types

Sequential API

Functional API

Consider the following eight steps to create deep learning model in Keras –

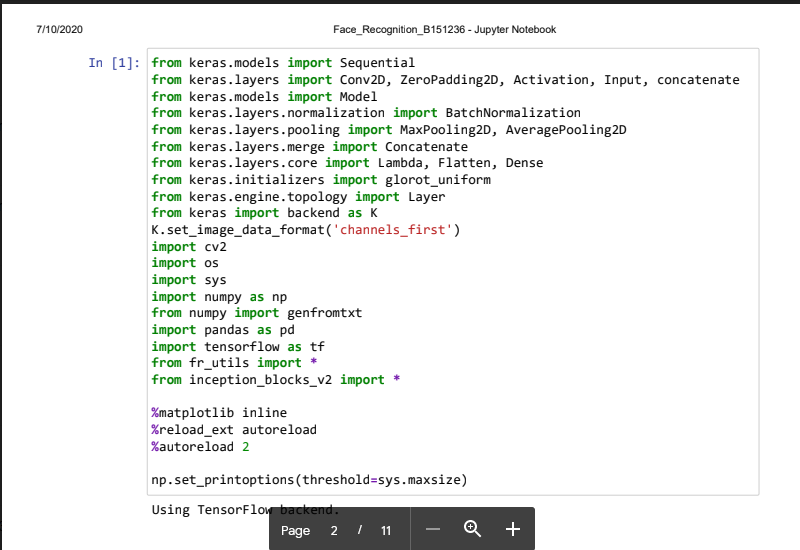
* Loading the data
* Preprocess the loaded data
* Definition of model
* Compiling the model
* Fit the specified model
* Evaluate it
* Make the required predictions
* Save the model

We will use the Jupyter Notebook for execution and display of output.

**5.5 CODING OF FACENET MODEL USING KERAS:**

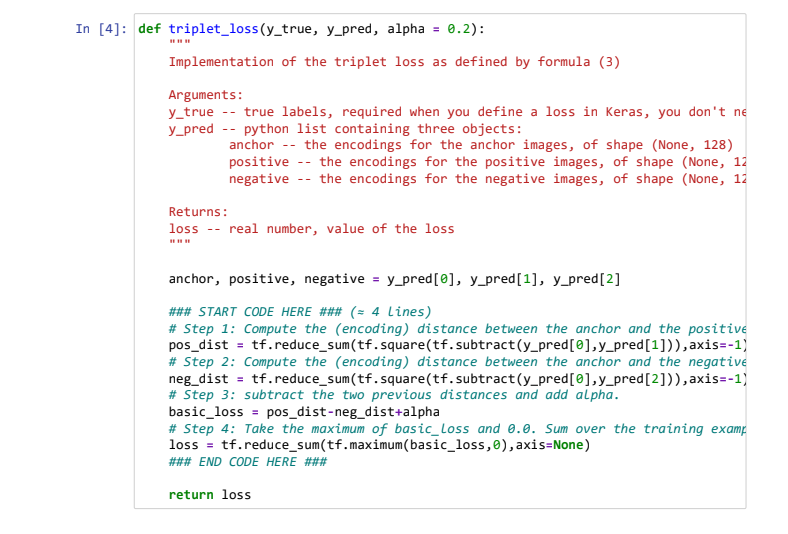
**CODE LINK:**<http://bit.ly/facerecognition_using_facenet>

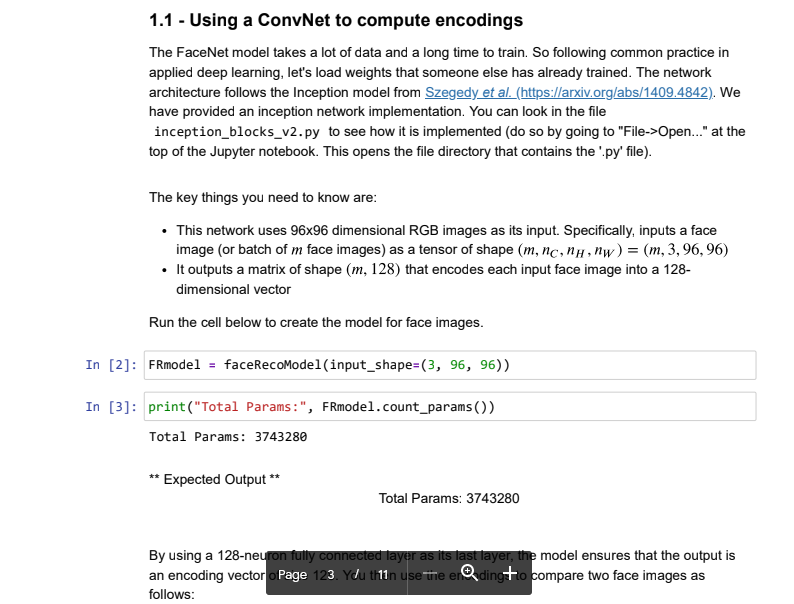
**Step1:** Open anaconda platform and create a new jupyter notebook (python) and import the required libraries as shown in the image below.



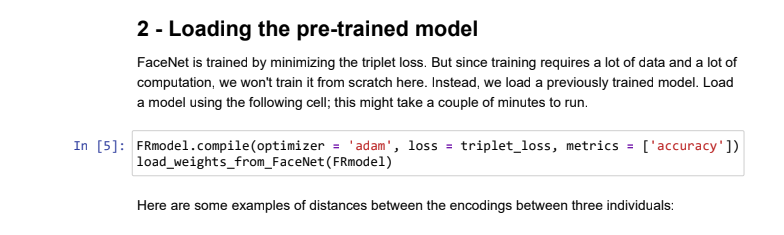
**Step2:**Loading the model and displaying the weight parameters.

**Step3:** Implementing the loss function (triplet loss)

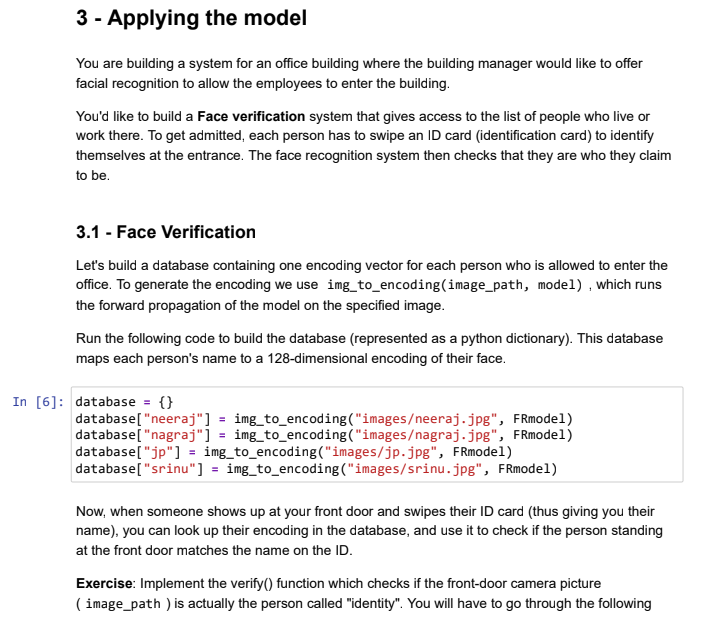


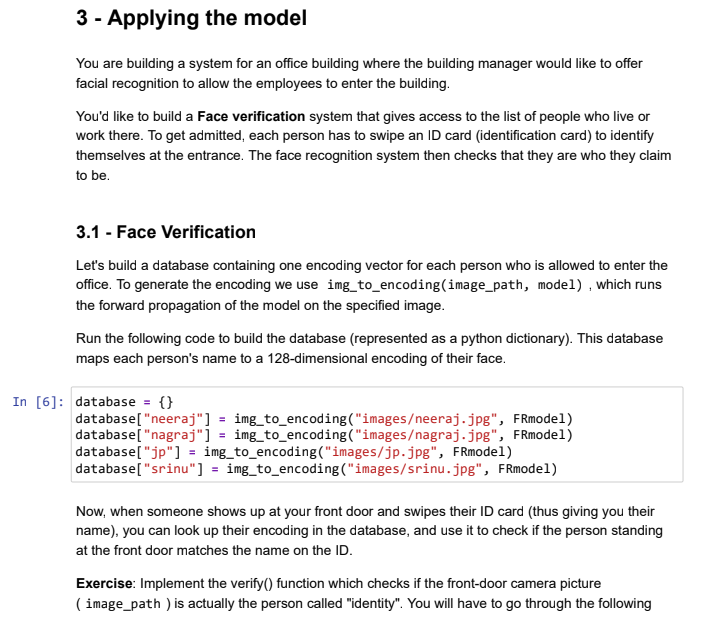


**Step4:** Loading the pretrained model

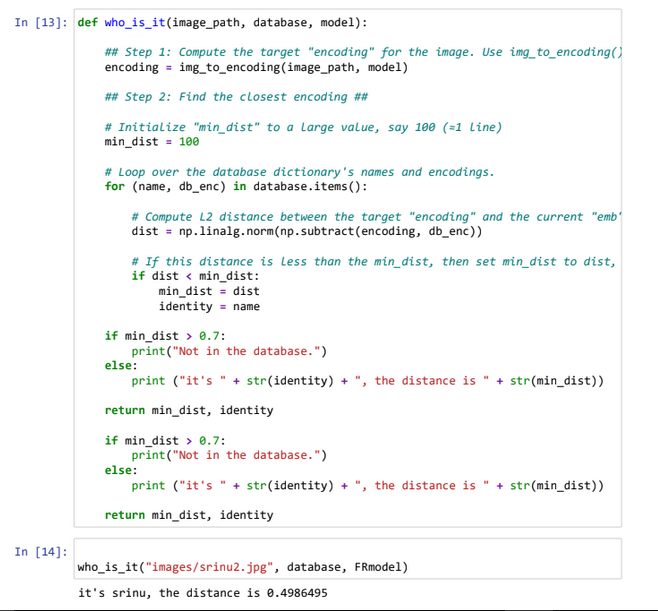


**Step5:** Applying the model for face Verification( Face verification means to check if the image is of the same person)

**Step6:** Verification with test set.



**Step7: Face recognition task** ( Here we will create a database of students and check whether the new image embedding vector distance is near to any of teh student embedding vectors in the database by means of threshold value.)





**External Links:**

**FaceNet Paper-** <https://arxiv.org/pdf/1503.03832.pdf>

**Drive link** - <http://bit.ly/facerecognition_using_facenet>

**5.6 Application of Facial Recognition:**

There are numerous applications of facial recognition. They can be segmented into blacklist and whitelist applications. Blacklist applications include the ones related to security & surveillance and identification of criminals. All other applications such as attendance tracking, access control and others fall under the category of whitelist applications.

|  |  |
| --- | --- |
| **End-Use** | **Top Applications** |
| **Offices** | Physical access to workspace facilities |
| **Government** | Helps to Identify missing children |
| **Banking and Telecom** | Help to know the current process to the customer, allow authentication of credit/debit cards |
| **Education** | Allow attendance tracking of the students and entry to labs |
| **Construction** | Control access to specific point at a site |
| **Real Estate Commercial** | Offers access to campus facilities like residence halls, common area, cafeteria, etc. |
| **Manufacturing** | Control and record access to specific locations for employees, visitors, vendors and maintenance staff |
| **Aviation** | Paperless travel at airports |
| **Warehouse** | Control process to provision entry and exit of vehicles |
| **Entertainment** | Access to multiplex cinema |

**5.7 Scope of Facial Recognition Technology in India :**

The world is using facial recognition technology and enjoying its benefits. Why should India be left out? There is a huge scope of this technology in India and it can help improve the country in various aspects. The technology and its applications can be applied across different segments in the country.

* Preventing the frauds at ATMs in India. A database of all customers with ATM cards in India can be created and facial recognition systems can be installed. So, whenever a user enters an ATM his photograph will be taken to permit access after it is matched with a stored photo from the database.
* Reporting duplicate voters in India.
* Passport and visa verification can also be done using this technology.
* Also, driving license verification can be done using the same approach.
* In the defence ministry, airports, and all other important places the technology can be used to ensure better surveillance and security.
* It can also be used during examinations such as Civil Services Exam, SSC, IIT, MBBS, and others to identify the candidates.
* This system can be deployed for verification and attendance tracking at various government offices and corporates.
* For access control verification and identification of authentic users it can also be installed in bank lockers and vaults.
* For identification of criminals the system can be used by police force also.

**6. Future**

The future of facial recognition technology is bright. Forecasters opine that this technology is expected to grow at a formidable rate and will generate huge revenues in the coming years. Security and surveillances are the major segments which will be deeply influenced. Other areas that are now welcoming it with open arms are private industries, public buildings, and schools. It is estimated that it will also be adopted by retailers and banking systems in coming years to keep fraud in debit/credit card purchases and payment especially the ones that are online. This technology would fill in the loopholes of a largely prevalent inadequate password system. In the long run, robots using facial recognition technology may also come to foray. They can be helpful in completing the tasks that are impractical or difficult for human beings to complete.

**7. Conclusion**

The increase in complexity in consumer goods such as automobiles and home appliances provides for a new market for technologies such as face recognition.Additionally, the recent events in the world has spurred a public demand for security and safety in public places; demands that could be partially met by deployment of these technologies.

**8. References:**

1. Y. Bengio, J. Louradour, R. Collobert, and J. Weston. Cur-riculum learning. InProc. of ICML, New York, NY, USA,2009. 2
2. D. Chen, X. Cao, L. Wang, F. Wen, and J. Sun. Bayesianface revisited: A joint formulation. InProc. ECCV, 2012. 2
3. D. Chen, S. Ren, Y. Wei, X. Cao, and J. Sun. Joint cascade face detection and alignment. InProc. ECCV, 2014. 7
4. J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, M. Mao,M. Ranzato, A. Senior, P. Tucker, K. Yang, Q. V. Le, andA. Y. Ng.Large scale distributed deep networks.InP. Bartlett, F. Pereira, C. Burges, L. Bottou, and K. Wein-berger, editors,NIPS, pages 1232–1240. 2012. 10
5. J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization.J.Mach. Learn. Res., 12:2121–2159, July 2011. 4
6. I. J. Goodfellow, D. Warde-farley, M. Mirza, A. Courville,and Y. Bengio. Maxout networks. InIn ICML, 2013. 4
7. G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller.Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts, Amherst, October2007. 5
8. Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E.Howard, W. Hubbard, and L. D. Jackel. Backpropagation Applied to handwritten zip code recognition.Neural Compu-tation, 1(4):541–551, Dec. 1989. 2, 4
9. M. Lin, Q. Chen, and S. Yan. Network in network.CoRR,abs/1312.4400, 2013. 2, 4, 6
10. C. Lu and X. Tang.Surpassing human-level face veri-fication performance on LFW with gaussianface.CoRR,abs/1404.3840, 2014. 1
11. D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning Representations by back-propagating errors.Nature, 1986. 2,4
12. M. Schultz and T. Joachims. Learning a distance metric from relative comparisons. In S. Thrun, L. Saul, and B. Schölkopf,editors,NIPS, pages 41–48. MIT Press, 2004. 2
13. T. Sim, S. Baker, and M. Bsat. The CMU pose, illumination,and expression (PIE) database. InIn Proc. FG, 2002. 2
14. Y. Sun, X. Wang, and X. Tang.Deep learning face representation by joint identification-verification.CoRR,abs/1406.4773, 2014. 1, 2, 3
15. Y. Sun, X. Wang, and X. Tang.Deeply learned face representations are sparse, selective, and robust.CoRR,abs/1412.1265, 2014. 1, 2, 5, 8
16. C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed,D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich.Going deeper with convolutions.CoRR, abs/1409.4842,2014. 2, 3, 4, 5, 6, 10
17. Y. Taigman, M. Yang, M. Ranzato, and L. Wolf. Deepface:Closing the gap to human-level performance in face verifica-tion. InIEEE Conf. on CVPR, 2014. 1, 2, 5, 7, 8, 9