METRIC LEARNING FOR LARGE IMBALANCED FACE DATASETS

MAJOR PROJECT-II REPORT

SUBMITTED IN FULFILMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE OF

MASTER OF TECHNOLOGY
IN
INFORMATION SYSTEMS

Submitted by:

ASHU KAUSHIK
2K19/ISY/03

Under the supervision of
DR. SEBA SUSAN
PROFESSOR

DEPARTMENT OF INFORMATION TECHNOLOGY
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi college of Engineering)
Bawana Road, Delhi-110042

JULY, 2021

DEPARTMENT OF INFORMATION TECHNOLOGY
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi college of Engineering)
Bawana Road, Delhi-110042

CANDIDATE'S DECLARATION

I, Ashu Kaushik, Roll No. 2K19/ISY/03 student of M.Tech, Information Systems, hereby declare that the Major Project-II report titled "METRIC LEARNING FOR LARGE IMBALANCED FACE DATASETS" which is submitted by me to the Department of Information Technology, Delhi Technological University, Delhi in fulfilment of the requirement for the award of the degree of Master of Technology, is original and not copied from any source without proper citation. This work has not previously formed the basis for the award of any Degree, Diploma Associateship, Fellowship or other similar title or recognition.

Place: Delhi **ASHU KAUSHIK**

Date: Jul 23, 2021

DEPARTMENT OF INFORMATION TECHNOLOGY
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi college of Engineering)
Bawana Road, Delhi-110042

CERTIFICATE

I hereby certify that the Major Project-II report titled "METRIC LEARNING FOR LARGE IMBALANCED FACE DATASETS" which is submitted by Ashu Kaushik, Roll No. 2K19/ISY/03 Information Technology, Delhi Technological University, Delhi in fulfilment of the requirement for the award of the degree of Master of Technology, is a record of the project work carried out by the student under my supervision. To the best of my knowledge this work has not been submitted in part or full for any Degree or Diploma to this University or elsewhere.

Place: Delhi **PROF. SEBA SUSAN**

Date: July 23, 2021 **SUPERVISOR**

ABSTRACT

Computer vision is a very trending field nowadays. The amount of digital data i.e. image, audio, video etc. is increasing day-by-day at a faster rate. So, various algorithms are being developed around such digital payloads to extract the maximum potential of computer systems. In this project I'll be working on image analysis, how these digital images can be played with on computer systems, how to remodel them based upon certain distinct characteristic features and its classification using various classifiers and similarity metrics like SVM, cosine similarity. I'll also be using Metric learning after running inbuilt functionalities, transforming the image parameters and again doing performance analysis of classification which leads out to be better than the initial results. I'll also be using Deep neural networks for feature extraction only and then apply my procedure for classification. I will be trying to devise a new algorithm to help improve the performance metrics for large imbalance datasets.

ACKNOWLEDGEMENT

I am very thankful to **Dr. Seba Susan** (Professor, Department of Information Technology) and all the faculty members of the Department of Information Technology at DTU. They all provided me with immense support and guidance for the project.

I would also like to express my gratitude to the University for providing us with the laboratories, infrastructure, testing facilities and environment which allowed us to work without any obstructions.

I would also like to appreciate the support provided to us by our lab assistants, seniors and our peer group who aided us with all the knowledge they had regarding various topics.

ASHU KAUSHIK

2K19/ISY/03

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS, ABBREVIATION, NOMENCLATURE

RGB…………………………………………………………………..Red Green Blue

LFW……………………………………………………….Labelled Faces in the Wild

LMNN………………………………………………..Large margin nearest neighbour

HOG…………………………………………………Histogram of Oriented Gradients

G_x……………………………………………………………Gradient in x-direction

G_y……………………………………………………………Gradient in y-direction

SVM………………………………………………………….Support Vector Machine

RBF…………………………………………………………..Radial Basis Function

V……………………………………………………………………………Validation

CV…………………………………………………………………Cross Validation

ML…………………………………………………………………Metric Learning

CNN…………………………………………………Convolutional Neural Networks

NCA…………………………………………Neighbourhood Component Analysis

MLKR…………………………………………..Metric Learning for Kernel Regression

CHAPTER-1 INTRODUCTION

In computer science, Digital Image processing is referred to as the use of computers to process digital images through various old and novel algorithms. It has many advantages as compared to analog image processing. This allows us to have a much wider range of algorithms that can be applied to the input and can avoid problems of noise and distortions.

Digital Image- It is basically a two or three dimensional signal/ array in terms of computers. It is defined by mathematical function $f(x, y)$ where x and y are coordinates of x and y direction respectively.

It can be of 3 types-

1. **RGB**- 3 dimensional; each for R,G & B and each pixel value ranging from [0, 255].
2. **Greyscale** - 2 dimensional and each pixel value ranging from [0, 255].
3. **Binary**- 2 dimensional and each pixel value either 0 or 255 based upon a threshold.

Basically I am doing image classification analysis because manual checking and classifying an image is a very cumbersome process. So, we need to automate the process so that the model will automatically decide the image label based upon its learning from the distinct feature descriptors.

To play around with the image we cannot directly feed it into a classifier rather we need to extract certain distinct descriptive features using some preprocessing techniques and create a csv of these features so that it can be easily stored and converted into arrays, so that it can be easily fed directly into a classifier.

I will be using image processing techniques to extract the best features which can describe the image and then fed into a classifier to predict the test samples. One of the main problem is to solve the class imbalance between the majority and minority classes [28]. Metric learning will also be used to transform the characteristic features of the image and again fed into the classifier to get better results.

CHAPTER-2 LITERATURE REVIEW

Face recognition or object recognition is the latest trend on which the researchers are working. Everything which we are using nowadays somewhere or the other requires your facial information like your mobiles, laptops can be unlocked by looking at your face.

Developing fast, reliable algorithms is the need of the hour. Several brands have discontinued the use of finger biometrics for security/ verification purposes rather they have shifted to the facial biometrics, because our face have some age invariant features which does not change over the period of time rather remains constant. And a human face has various features like colour, texture, tone etc. which make it difficult to replicate.

Image classification also has a major role in object detection, helpful for the armed forces so that they can have automated weaponry in infeasible regions.

The author in [8] proposed a model Naive Deep Convolutional Neural Network for multi class classification of images of MFC dataset and then comparing it with the benchmarks of LFW dataset. It's a standard neural network consisting of 10 hidden layers out of which the last layer is softmax layer. The output of the 8^{th} hidden layer just before the softmax layer is taken as image features. PCA reduction is done and L2 distance norm is used to find the similarity between the two images.

Various feature descriptors are used to extract the unique identification information from the image. In [9], the author compares various descriptors and outputs that the HOG descriptors gives best results when coupled with SVM and RBF kernel.

In [11] Deep convolutional neural network is used for image classification but with a trick during the training with deep architecture. A CNN based model based on GoogleNet style [10] model called Inception v1. It consists of multiple convolutions, multiple filters, pooling layers in parallel with the inception layer. It is also followed by Relu, softmax layers called activation function layers which are non-linear in nature.

An approach based on class weights [25] has been proposed to resolve the problem of class imbalance among various classes. Features being used were extracted from Convolutional Auto Encoder.

Face recognition is a challenging task in various conditions where there is a large variation of poses, contrast, illumination, expressions etc. The problem becomes more difficult when there is a single data source. The author in [12] proposes an approach based on Discriminative Gaussian Process Latent Variable model named Gaussian Face to improve the variety of training data . It extracts data from multiple-source domains to improve the quality of training dataset, by getting more details related to illumination, expressions etc.

CHAPTER-3 PREPARING THE DATASET

3.1 DATASET USED

Labeled Face in the Wild (LFW) [4], a popular dataset in the field of computer vision, designed to study the problem of unconstrained face recognition. It has more than 13k images of different celebrities, out of which around 1680 have more than two images.

Size of the dataset = > 13k images = 13233
total number of celebs = 5749
Celebs with > 2 Images = 1680

There're a few errors in the dataset but have been left as it is so as to avoid confusion and since previous published knowledge shows there is no significant advantage after rectifying it as well.

I have used **LFWcrop** face dataset which is exactly the same as the LFW but only have the **cropped images** of the celebs without any other objects in the background so as to reduce the complexity. LFWcrop was created over a concern on the existing LFW dataset i.e. the face matching accuracy can be boosted by using the background part which is a forgery.

3.2 SPLITTING THE DATASET

It's a very important part for any dataset that shows how it's being split into testing and training samples.

Initially the dataset looks very complex and is cumbersome to work upon as it has various nested folders and files. So, I transformed the overall directory structure for my project purpose.

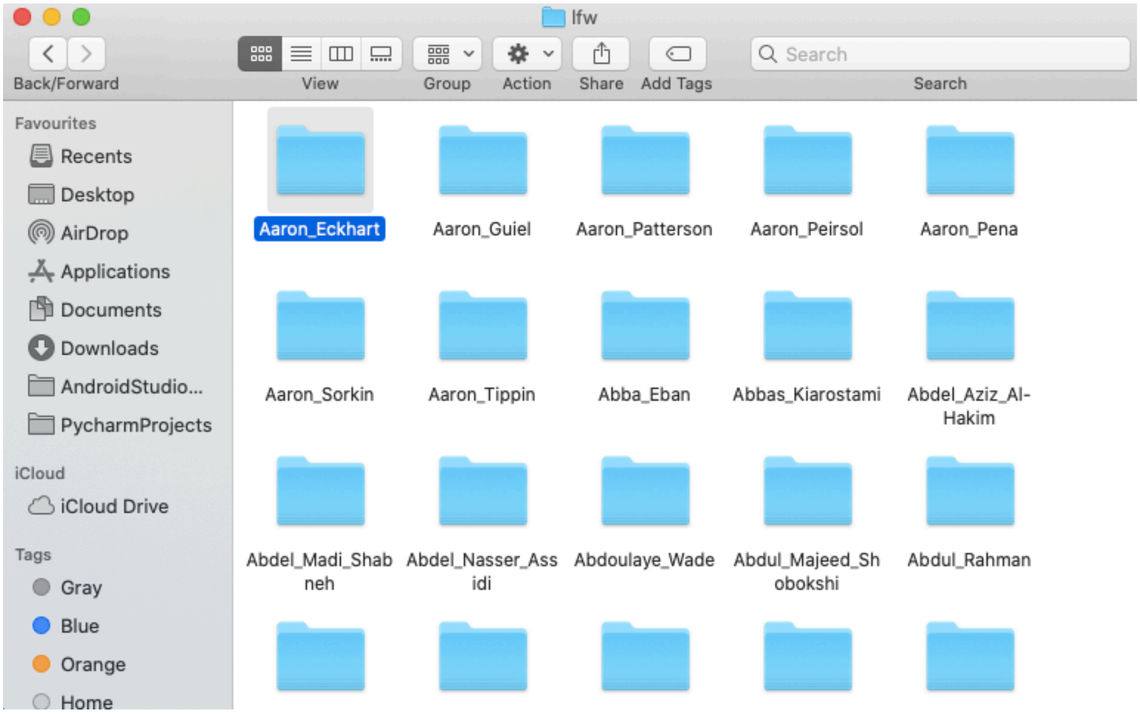

Fig. 3.1 Initial directory structure

The initial look of the directory was somewhat like the Fig. 3.1. I converted the directory hierarchy into a single folder for all celebs, which made it easier to work upon.

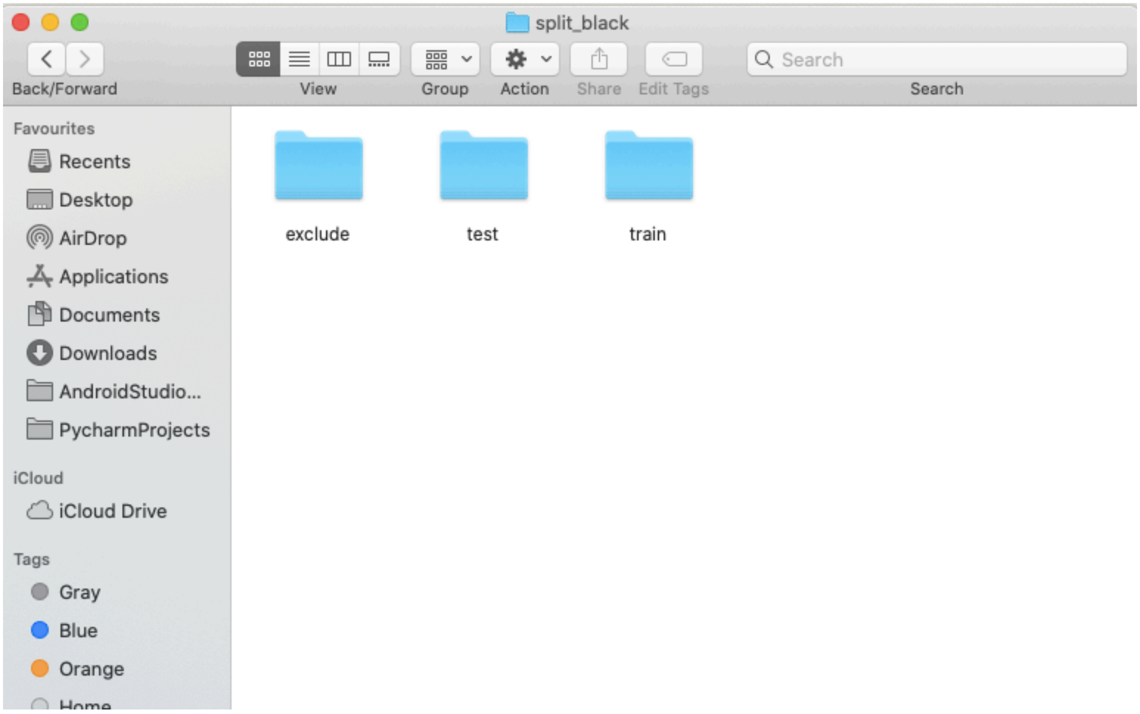

Fig. 3.2 New directory structure

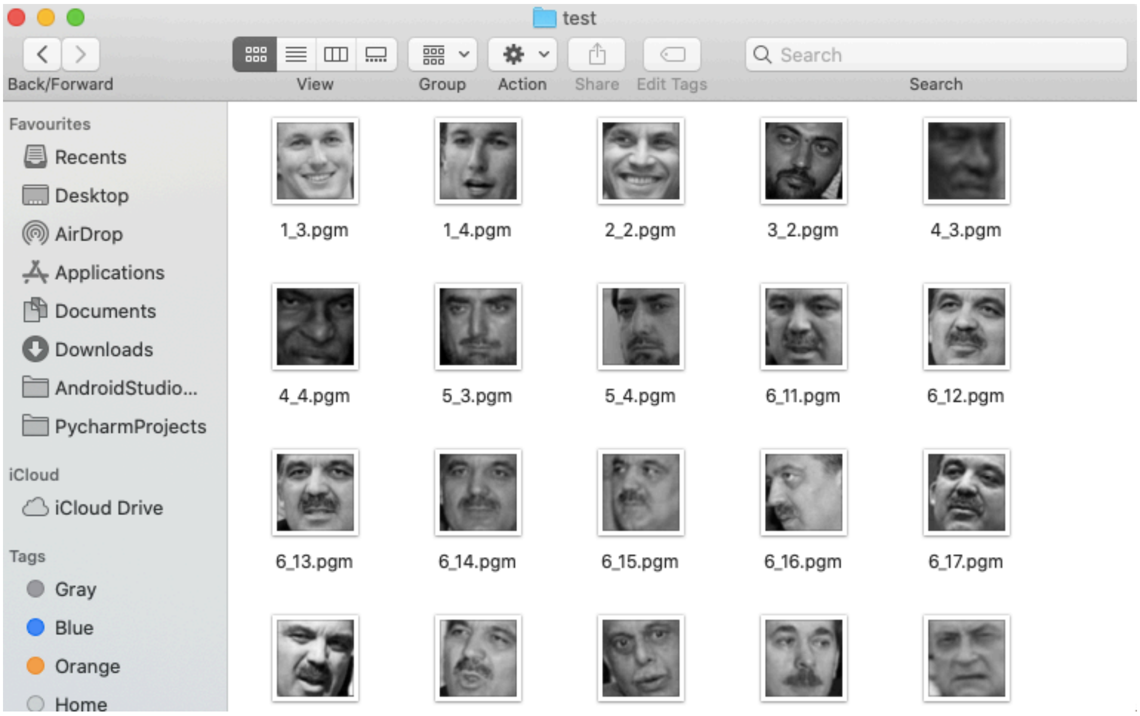

Fig. 3.3 New naming nomenclature

All files were moved to a folder called split_black which further contains the test, train, and exclude folder.

Exclude folder contains the celebs which contains < 2 image samples.

In test and train folders the samples (with ≥ 2 images/celeb) are divided exactly into two halves.

Also the celebs were renamed to numerical values so that it can be easily fed into a classifier and are named from 1 to n.

Naming nomenclature used for each sample- label_no. of image; eg. "2_3.pgm " i.e. celeb 2's 3rd sample.

CHAPTER-4 FEATURE DESCRIPTORS AND CLASSIFICATION

4.1 WHAT IS A FEATURE DESCRIPTOR ?

It is an algorithm which takes an image as input and outputs its unique feature vectors. It encodes distinctive information into a series of numbers called vectors and acts as a numerical fingerprint of the image that can be used to differentiate it from other samples. It is a simple representation of the image consisting of the most important information only.

Some most popular Feature descriptors are:-

1. HOG- Histogram of oriented gradients
2. SIFT- scale invariant feature transform
3. SURF- speeded-up robust feature

4.2 HISTOGRAM OF ORIENTED GRADIENTS

It is a feature descriptor method mostly used to extract feature vectors from the image. It is mainly used in the field of computer vision more appropriately for object detection.

How is it different from other descriptors?

This descriptor mainly focuses on structure and form factor of the image. It is different as it also takes the direction sense into consideration while detecting the edges where there is a prominent change in the neighbouring pixel value [1]. It takes both magnitude (gradient) and direction to extract features.

These params i.e. gradient and direction are calculated on ‘localised' portions of the image. One single image is thus broken down into smaller image cells and then both the values are calculated for each cell/region.

For each smaller region a separate Histogram is made using gradient and orientation of pixel values.

Algorithm [7]-

1. Preprocess the image - It is a very crucial step of any machine learning algorithm. We need to scale the image to a width to height ratio of 1:2. The image size should be preferably 64x128 because this makes calculations a bit easier and is also the exact value used in the original paper. [1]

2. Calculating gradients both in x and y directions - gradient is defined as the minute changes in x and y directions eg.

$$\begin{bmatrix} 152 & 68 & 125 \\ 78 & 86 & 87 \\ 200 & 56 & 210 \end{bmatrix}$$

Let us consider the pixel value as 86. To get the gradient in x-direction, subtract the value of the left from the right pixel and similarly for the y-direction, subtract the below pixel from the above one.

i.e. change in X direction, G_x = 87-78 = 9
& in Y-direction, G_y = 68-56 = 12

We'll get two matrices, one for G_x and one for G_y. It is similar to the Sobel kernel. The gradient would be higher around the edges as there would be sharp change in the intensity.

3. Calculating the magnitude & orientation- magnitude and orientation of each pixel value is calculated using Pythagoras theorem.

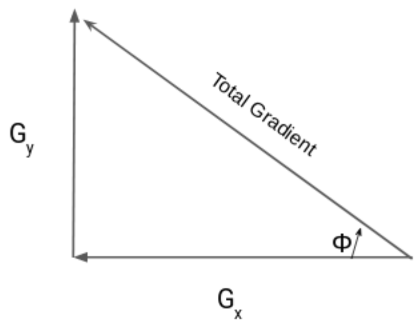

Fig. 4.1 Gradient and orientation

$$\text{Total magnitude} = \sqrt{(G_x)^2 + (G_y)^2} \tag{4.1}$$

$$\text{Total gradient magnitude} = \sqrt{(9)^2 + (12)^2} = 15 \tag{4.2}$$

Orientation can be calculated as $\theta = atan\dfrac{G_y}{G_x} = 53.13°$ (4.3)

So, now we have both the magnitudes and orientation values for each pixel. Now we need to generate a histogram using this.

4. Calculating histogram of gradients- we make bins of 20° each therefore we have 9 bins in 180° and we fill the gradient value in proportions to bins according to the orientation value. The bigger angle closer to the angle value get a bigger proportion.

5. Initially the hog is calculated for 8x8 cells; we get a 9x1 feature vector.

6. Normalise the gradients in 16x16 cells(36x1 feature vector) & get the features of the complete image.

4.3 CONVOLUTIONAL NEURAL NETWORKS (CNN)

It is a Deep Learning algorithm in which input given is an image, and various objects in the images are assigned weights and biases to become differentiable from others. It requires very less preprocessing when compared to other classification techniques. While in primitive methods filters are hand-engineered, with enough training, ConvNets have the ability to learn these filters/characteristics.

The ConvNet architecture is somewhat similar to the connectivity pattern of neurons in human brain and was inspired by the organization of the Visual Cortex. Individual neurons respond to stimuli only in a restricted region of the visual field known as the Receptive Field. A collection of such fields overlap to cover the entire visual area.

CNN's are basically an ANN and the prominent application is in the Image Analysis field. It detects data patterns in the images. It consists of various hidden layers called as convolutional layers(can be one or more than one). It has some non-convolutional layers too, but the basis are convolution layers which perform the main operations.

Convolution is a mathematical operation being performed on two functions (lets say f and g) that produces a third function which expresses how the shape of one is modified by the other. It uses various filters just to do convolution at different layer of the network just to detect the prominent edges i.e. edge detection, smoothening and sharpening.

Pooling is also done to reduce the size/ dimensions of the feature vectors. Various types of pooling are there like Max > Sum > Average based upon the preference given here.

Some of the activation functions used here are like ReLu, Sigmoid etc.

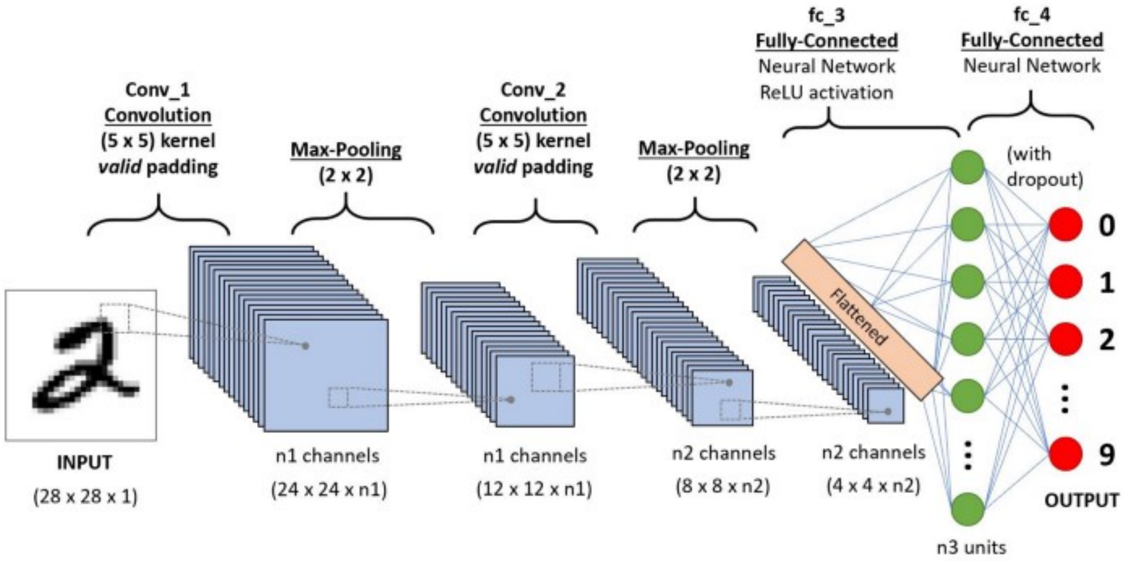

Fig. 4.2 Basic CNN

4.4 VGGFACE (DEEP CNN MODEL)

It is basically a Convolutional Neural network when we look it from a broader perspective. It is very deep, in the sense because it is constituted of a long sequence of convolutional layers.

layer	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
type	input	conv	relu	conv	relu	mpool	conv	relu	conv	relu	mpool	conv	relu	conv	relu	conv	relu	mpool	conv
name	–	conv1_1	relu1_1	conv1_2	relu1_2	pool1	conv2_1	relu2_1	conv2_2	relu2_2	pool2	conv3_1	relu3_1	conv3_2	relu3_2	conv3_3	relu3_3	pool3	conv4_1
support	–	3	1	3	1	2	3	1	3	1	2	3	1	3	1	3	1	2	3
filt dim	–	3	–	64	–	–	64	–	128	–	–	128	–	256	–	256	–	–	256
num filts	–	64	–	64	–	–	128	–	128	–	–	256	–	256	–	256	–	–	512
stride	–	1	1	1	1	2	1	1	1	1	2	1	1	1	1	1	1	2	1
pad	–	1	0	1	0	0	1	0	1	0	0	1	0	1	0	1	0	0	1

layer	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37
type	relu	conv	relu	conv	relu	mpool	conv	relu	conv	relu	conv	relu	mpool	conv	relu	conv	relu	conv	softmx
name	relu4_1	conv4_2	relu4_2	conv4_3	relu4_3	pool4	conv5_1	relu5_1	conv5_2	relu5_2	conv5_3	relu5_3	pool5	fc6	relu6	fc7	relu7	fc8	prob
support	1	3	1	3	1	2	3	1	3	1	3	1	2	7	1	1	1	1	1
filt dim	–	512	–	512	–	–	512	–	512	–	512	–	–	512	–	4096	–	4096	–
num filts	–	512	–	512	–	–	512	–	512	–	512	–	–	4096	–	4096	–	2622	–
stride	1	1	1	1	1	2	1	1	1	1	1	1	2	1	1	1	1	1	1
pad	0	1	0	1	0	0	1	0	1	0	1	0	0	0	0	0	0	0	0

Fig. 4.3 VGGFace layers from original paper

ARCHITECTURE

1. Size of the input image required is 224x224. Then there is a different set of convolutional layers in VGG-16 architecture.

2. First we have two convolutional layers (layer 1 & 2) of 64 channel of 3x3 kernel with stride 1 and padding 1.

3. After that we have two convolutional layers (layer 3 & 4) of 128 channel of 3x3 kernel.

4. Then we have three convolutional layers (layer 5, 6 & 7) of 256 channel of 3x3 kernel.

5. After that we have three convolutional layers (layer 8, 9 & 10) of 512 channel of 3x3 kernel.

6. Then we have three convolutional layers (layer 11, 12 & 13) of 512 channel of 3x3 kernel.

7. After each set of layers we have Max pool layer with max stride 2.

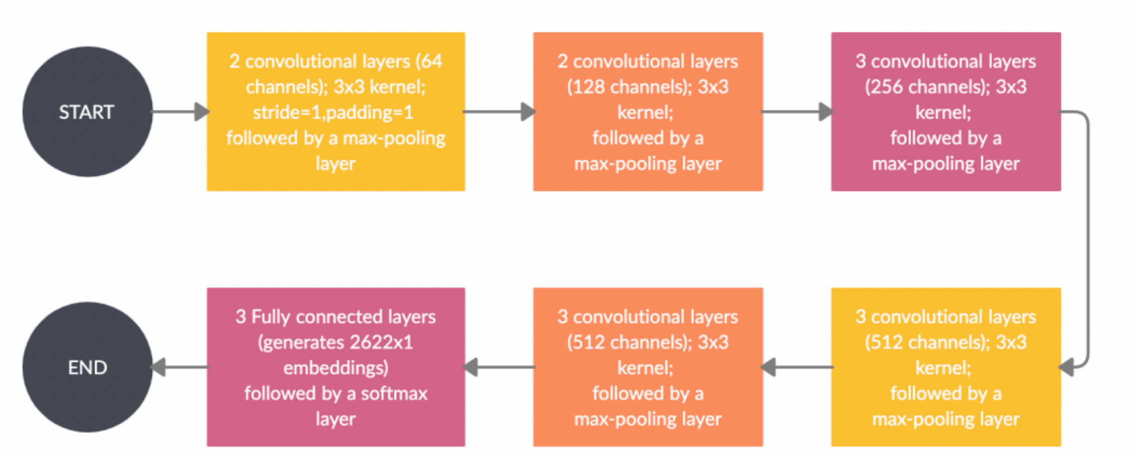

Fig. 4.4 VGGFace architecture

4.5 METRIC LEARNING

WHAT IS METRIC LEARNING ?

Various algorithms of machine learning [2] requires a distance measure between data points. Initially standard metrics were used like Euclidean, City block, cosine etc. with a prior knowledge of the domain. Sometimes it becomes difficult to design a metric suited for a particular dataset.
Metric learning automatically constructs task-specific distance metrics for weakly supervised data.

It is broadly classified into two types of problems:-

1. Supervised learning- It has access to all the features and the labels for each instance of a classification problem. The main task is to learn a distance metric which places same class labels close together and pushing them away from other labels.

2. Weakly supervised learning- It only has access to some supervised data points at tuple level only.

MAHALANOBIS DISTANCE

Given a real valued parameter matrix L of shape (num_dims, n_features), where n_features is the number of features in the dataset. Therefore the Mahalanobis distance associated with L is

$$D(x, x') = \sqrt{(Lx - Lx')^T (Lx - Lx')} \quad (4.4)$$

It is nothing but the Euclidean distance between two points after the feature space defined by L is linearly transformed. **If L is identity matrix then, Mahalanobis distance = Euclidean distance**

TYPES OF SUPERVISED METRIC LEARNING

1. LMNN (Large margin nearest neighbour)
2. NCA (Neighbourhood Component Analysis)
3. LFDA (Local Fischer Discriminant Analysis)
4. MLKR (Metric Learning for Kernel Regression)

4.4 PROPOSED ALGORITHM FOR FINDING MAJORITY AND MINORITY CLASSES [23]

- **Algorithm** Sum-based partitioning into (*Majority*, *Minority*) classes
- **Input**: List of *n* classes sorted in the decreasing order of number of samples
- **Parameter:** *temp*=0
- **Output**: Partition of the *n* classes into (*Majority*, *Minority*) groups

  1: Set i=1.

  2: Create two partitions of sorted list of classes 1:*i* and *i*+1:*n*.

  3: Find individual sums of samples in the two partitions

  4: Find the difference between the two sums.

  5: If difference is less than *temp*, set *temp* equal to difference and *ans* = *i*.

  6: *i*=*i*+1

  7: REPEAT steps 2 to 6 UNTIL *i*=n-1

  8: *Majority* class=1:*ans*; *Minority* class=*ans*+1: *n*

  9: **return** solution

Transformation used by me till date - **LMNN, NCA, MLKR**

The learned metric attempts to keep the K-nearest neighbours of the same class as close as possible, while the samples of different classes are separated by large margin.

This is learned by solving the optimisation problem ()-

$$\min_{\mathbf{L}} \sum_{i,j} \eta_{ij} \left\| \mathbf{L}\left(\mathbf{x_i} - \mathbf{x_j}\right) \right\|^2 + c \sum_{i,j,l} \eta_{ij}\left(1 - y_{ij}\right)\left[1 + \left\| \mathbf{L}\left(\mathbf{x_i} - \mathbf{x_j}\right) \right\|^2 - \left\| \mathbf{L}\left(\mathbf{x_i} - \mathbf{x_l}\right) \right\|^2\right]_+ \tag{4.7}$$

where x_i is a data point, x_j is one of the nearest neighbour having the same label, and x_l be the data points in the same region with different labels η_{ij}, y_{ij} belong to {0,1} are both indicators.

I have used k=3 [2, 5, 29], with 5 samples/celeb for the top 186 classes i.e. total samples used for transformation = 186*5 = 930 and all the results are shown in the next section.

4.5 COSINE SIMILARITY

It is a similarity metric used to see how similar two features/ documents are. Mathematically, it measures the cosine of the angle between the two feature vectors given by the equation below

$$\cos\theta = \frac{\vec{a} \cdot \vec{b}}{\|\vec{a}\|\|\vec{b}\|} = \frac{\sum_i^n a_i b_i}{\sqrt{\sum_i^n a_i^2}\sqrt{\sum_i^n b_i^2}} \tag{4.8}$$

4.6 METHODS USED FOR CLASSIFICATION

SVM, its variants, cosine similarity was used to classify the testing samples.

1. Inbuilt SVM
2. Inbuilt SVM + Transformation
3. Cosine similarity
4. Cosine similarity + Transformation (LMNN, NCA, MLKR) {subset of majority classes used for learning the transformation metric} [31]
5. Cosine Similarity + 2-way metric learning {subset of majority and minority classes used to learn two different transformation metrics} [30]

CHAPTER-5 EXPERIMENTAL RESULTS

5.1 SEM-2 RESULTS (HOG FEATURES)

Table 5.1.1 Class wise accuracy LFW dataset, SVM

Name	Label	NOS	V	CV	V_ML	CV_ML
George_W_Bush_0001.pgm	533	530	0.932	0.917	0.951	0.928
Colin_Powell_0001.pgm	310	236	0.915	0.966	0.958	0.958
Tony_Blair_0001.pgm	1589	144	0.722	0.736	0.764	0.736
Donald_Rumsfeld_0001.pgm	393	121	0.783	0.902	0.767	0.918
Gerhard_Schroeder_0001.pgm	538	109	0.778	0.818	0.815	0.891
Ariel_Sharon_0001.pgm	112	77	0.842	0.846	0.816	0.821
Hugo_Chavez_0001.pgm	631	71	0.6	0.667	0.571	0.722
Junichiro_Koizumi_0001.pgm	857	60	0.767	0.733	0.733	0.8
Jean_Chretien_0001.pgm	711	55	0.852	0.5	0.852	0.643
John_Ashcroft_0001.pgm	770	53	0.538	0.704	0.692	0.704
Jacques_Chirac_0001.pgm	657	52	0.577	0.5	0.615	0.692
Serena_Williams_0001.pgm	1469	52	0.692	0.885	0.846	0.885
Vladimir_Putin_0001.pgm	1629	49	0.625	0.76	0.625	0.8
Luiz_Inacio_Lula_da_Silva_0001.pgm	995	48	0.458	0.542	0.625	0.708
Gloria_Macapagal_Arroyo_0001.pgm	550	44	0.727	0.818	0.818	0.955
Arnold_Schwarzenegger_0001.pgm	117	42	0.286	0.19	0.429	0.238
Jennifer_Capriati_0001.pgm	722	42	0.619	0.524	0.714	0.571
Lleyton_Hewitt_0001.pgm	981	41	0.45	0.571	0.6	0.619
Laura_Bush_0001.pgm	933	41	0.5	0.524	0.7	0.81
Hans_Blix_0001.pgm	589	39	0.474	0.7	0.526	0.85
Alejandro_Toledo_0001.pgm	43	39	0.632	0.45	0.579	0.4
Nestor_Kirchner_0001.pgm	1190	37	0.333	0.579	0.333	0.684
Andre_Agassi_0001.pgm	79	36	0.722	0.611	0.5	0.611
Alvaro_Uribe_0001.pgm	65	35	0.529	0.667	0.588	0.611
Silvio_Berlusconi_0001.pgm	1492	33	0.125	0.235	0.188	0.294

Table 5.1.2 Class wise accuracy LFW dataset, cosine similarity

Name	Label	NOS	V	CV	V_ML	CV_ML
George_W_Bush_0001.pgm	533	530	0.668	0.679	0.762	0.755
Colin_Powell_0001.pgm	310	236	0.729	0.78	0.831	0.814
Tony_Blair_0001.pgm	1589	144	0.514	0.389	0.569	0.403
Donald_Rumsfeld_0001.pgm	393	121	0.6	0.607	0.767	0.77
Gerhard_Schroeder_0001.pgm	538	109	0.463	0.364	0.648	0.691
Ariel_Sharon_0001.pgm	112	77	0.579	0.487	0.658	0.564
Hugo_Chavez_0001.pgm	631	71	0.143	0.194	0.229	0.278
Junichiro_Koizumi_0001.pgm	857	60	0.533	0.567	0.733	0.767
Jean_Chretien_0001.pgm	711	55	0.556	0.286	0.556	0.393
John_Ashcroft_0001.pgm	770	53	0.192	0.333	0.423	0.481
Jacques_Chirac_0001.pgm	657	52	0.462	0.346	0.654	0.385
Serena_Williams_0001.pgm	1469	52	0.577	0.423	0.615	0.538
Vladimir_Putin_0001.pgm	1629	49	0.417	0.2	0.375	0.52
Luiz_Inacio_Lula_da_Silva_0001.pgm	995	48	0.375	0.417	0.583	0.708
Gloria_Macapagal_Arroyo_0001.pgm	550	44	0.318	0.227	0.273	0.318
Jennifer_Capriati_0001.pgm	722	42	0.524	0.429	0.571	0.476
Arnold_Schwarzenegger_0001.pgm	117	42	0.048	0.0	0.143	0.143
Laura_Bush_0001.pgm	933	41	0.15	0.286	0.15	0.286
Lleyton_Hewitt_0001.pgm	981	41	0.3	0.19	0.3	0.381
Alejandro_Toledo_0001.pgm	43	39	0.158	0.1	0.211	0.35
Hans_Blix_0001.pgm	589	39	0.474	0.45	0.632	0.45
Nestor_Kirchner_0001.pgm	1190	37	0.222	0.158	0.333	0.211
Andre_Agassi_0001.pgm	79	36	0.278	0.389	0.333	0.556
Alvaro_Uribe_0001.pgm	65	35	0.235	0.278	0.353	0.444
Megawati_Sukarnoputri_0001.pgm	1083	33	0.438	0.412	0.562	0.412

Inbuilt SVM- accuracy was 27.4% and 24.8% for V and CV respectively.

Cosine similarity- accuracy was 21.5% and 19.1% for V and CV respectively.

Table 5.1.3 Overall Performance Analysis SVM + transformation

Performance Metric	V	CV
accuracy score	0.287	0.274
f1- macro	0.0590	0.053
AUC score	0.528	0.525

Table 5.1.4 Overall Performance Analysis Cosine Similarity + transformation

Performance Metric	V	CV
accuracy score	0.2684	0.246
f1- macro	0.1006	0.0967
AUC score	0.556	0.554

5.2 SEM-3 RESULTS (VGGFACE FEATURES)

Table 5.2.1 Class wise accuracy LFW dataset, SVM

Name	Label	NOS	V	CV	V_ML	CV_ML
George_W_Bush_0001.pgm	533	530	0.932	0.917	0.951	0.928
Colin_Powell_0001.pgm	310	236	0.915	0.966	0.958	0.958
Tony_Blair_0001.pgm	1589	144	0.722	0.736	0.764	0.736
Donald_Rumsfeld_0001.pgm	393	121	0.783	0.902	0.767	0.918
Gerhard_Schroeder_0001.pgm	538	109	0.778	0.818	0.815	0.891
Ariel_Sharon_0001.pgm	112	77	0.842	0.846	0.816	0.821
Hugo_Chavez_0001.pgm	631	71	0.6	0.667	0.571	0.722
Junichiro_Koizumi_0001.pgm	857	60	0.767	0.733	0.733	0.8
Jean_Chretien_0001.pgm	711	55	0.852	0.5	0.852	0.643
John_Ashcroft_0001.pgm	770	53	0.538	0.704	0.692	0.704
Jacques_Chirac_0001.pgm	657	52	0.577	0.5	0.615	0.692
Serena_Williams_0001.pgm	1469	52	0.692	0.885	0.846	0.885
Vladimir_Putin_0001.pgm	1629	49	0.625	0.76	0.625	0.8
Luiz_Inacio_Lula_da_Silva_0001.pgm	995	48	0.458	0.542	0.625	0.708
Gloria_Macapagal_Arroyo_0001.pgm	550	44	0.727	0.818	0.818	0.955
Arnold_Schwarzenegger_0001.pgm	117	42	0.286	0.19	0.429	0.238
Jennifer_Capriati_0001.pgm	722	42	0.619	0.524	0.714	0.571
Lleyton_Hewitt_0001.pgm	981	41	0.45	0.571	0.6	0.619
Laura_Bush_0001.pgm	933	41	0.5	0.524	0.7	0.81
Hans_Blix_0001.pgm	589	39	0.474	0.7	0.526	0.85
Alejandro_Toledo_0001.pgm	43	39	0.632	0.45	0.579	0.4
Nestor_Kirchner_0001.pgm	1190	37	0.333	0.579	0.333	0.684
Andre_Agassi_0001.pgm	79	36	0.722	0.611	0.5	0.611
Alvaro_Uribe_0001.pgm	65	35	0.529	0.667	0.588	0.611
Silvio_Berlusconi_0001.pgm	1492	33	0.125	0.235	0.188	0.294

Table 5.2.2 Class wise accuracy LFW dataset, cosine similarity

Name	Label	NOS	V	CV	V_ML	CV_ML
George_W_Bush_0001.pgm	533	530	0.668	0.679	0.762	0.755
Colin_Powell_0001.pgm	310	236	0.729	0.78	0.831	0.814
Tony_Blair_0001.pgm	1589	144	0.514	0.389	0.569	0.403
Donald_Rumsfeld_0001.pgm	393	121	0.6	0.607	0.767	0.77
Gerhard_Schroeder_0001.pgm	538	109	0.463	0.364	0.648	0.691
Ariel_Sharon_0001.pgm	112	77	0.579	0.487	0.658	0.564
Hugo_Chavez_0001.pgm	631	71	0.143	0.194	0.229	0.278
Junichiro_Koizumi_0001.pgm	857	60	0.533	0.567	0.733	0.767
Jean_Chretien_0001.pgm	711	55	0.556	0.286	0.556	0.393
John_Ashcroft_0001.pgm	770	53	0.192	0.333	0.423	0.481
Jacques_Chirac_0001.pgm	657	52	0.462	0.346	0.654	0.385
Serena_Williams_0001.pgm	1469	52	0.577	0.423	0.615	0.538
Vladimir_Putin_0001.pgm	1629	49	0.417	0.2	0.375	0.52
Luiz_Inacio_Lula_da_Silva_0001.pgm	995	48	0.375	0.417	0.583	0.708
Gloria_Macapagal_Arroyo_0001.pgm	550	44	0.318	0.227	0.273	0.318
Jennifer_Capriati_0001.pgm	722	42	0.524	0.429	0.571	0.476
Arnold_Schwarzenegger_0001.pgm	117	42	0.048	0.0	0.143	0.143
Laura_Bush_0001.pgm	933	41	0.15	0.286	0.15	0.286
Lleyton_Hewitt_0001.pgm	981	41	0.3	0.19	0.3	0.381
Alejandro_Toledo_0001.pgm	43	39	0.158	0.1	0.211	0.35
Hans_Blix_0001.pgm	589	39	0.474	0.45	0.632	0.45
Nestor_Kirchner_0001.pgm	1190	37	0.222	0.158	0.333	0.211
Andre_Agassi_0001.pgm	79	36	0.278	0.389	0.333	0.556
Alvaro_Uribe_0001.pgm	65	35	0.235	0.278	0.353	0.444
Megawati_Sukarnoputri_0001.pgm	1083	33	0.438	0.412	0.562	0.412

Table 5.2.3 Comparison of various Metric Learning Techniques like LMNN, NCA, MLKR

Name	Label	NOS	LMNN_V	LMNN_CV	NCA_V	NCA_CV	MLKR_V	MLKR_CV
George_W_Bush_0001.pgm	533	530	0.94	0.913	0.898	0.932	0.849	0.815
Colin_Powell_0001.pgm	310	236	0.932	0.915	0.89	0.814	0.856	0.847
Tony_Blair_0001.pgm	1589	144	0.75	0.653	0.611	0.542	0.611	0.417
Donald_Rumsfeld_0001.pgm	393	121	0.867	0.82	0.767	0.623	0.717	0.689
Gerhard_Schroeder_0001.pgm	538	109	0.815	0.836	0.537	0.691	0.537	0.636
Ariel_Sharon_0001.pgm	112	77	0.842	0.872	0.684	0.872	0.658	0.795
Hugo_Chavez_0001.pgm	631	71	0.943	0.917	0.829	0.917	0.771	0.889
Junichiro_Koizumi_0001.pgm	857	60	0.933	0.9	0.9	0.867	0.767	0.633
Jean_Chretien_0001.pgm	711	55	0.704	0.679	0.593	0.393	0.519	0.393
John_Ashcroft_0001.pgm	770	53	0.846	0.815	0.692	0.704	0.385	0.407
Jacques_Chirac_0001.pgm	657	52	0.769	0.692	0.731	0.692	0.577	0.5
Serena_Williams_0001.pgm	1469	52	0.731	0.923	0.692	0.692	0.731	0.885
Vladimir_Putin_0001.pgm	1629	49	0.792	0.88	0.667	0.72	0.583	0.68
Luiz_Inacio_Lula_da_Silva_0001.pgm	995	48	0.667	0.875	0.625	0.833	0.583	0.542
Gloria_Macapagal_Arroyo_0001.pgm	550	44	0.773	0.955	0.636	0.727	0.591	0.636
Arnold_Schwarzenegger_0001.pgm	117	42	0.714	0.524	0.476	0.524	0.524	0.571
Jennifer_Capriati_0001.pgm	722	42	0.714	0.714	0.524	0.619	0.619	0.476
Laura_Bush_0001.pgm	933	41	0.8	0.762	0.75	0.667	0.5	0.762
Lleyton_Hewitt_0001.pgm	981	41	0.85	0.762	0.75	0.619	0.6	0.571
Alejandro_Toledo_0001.pgm	43	39	0.684	0.7	0.474	0.6	0.526	0.4
Hans_Blix_0001.pgm	589	39	1.0	0.9	0.789	0.85	0.474	0.6
Nestor_Kirchner_0001.pgm	1190	37	0.833	0.895	0.667	0.684	0.5	0.579
Andre_Agassi_0001.pgm	79	36	0.778	0.722	0.556	0.722	0.333	0.444
Alvaro_Uribe_0001.pgm	65	35	0.647	0.778	0.412	0.611	0.471	0.667
Megawati_Sukarnoputri_0001.pgm	1083	33	0.875	0.882	0.625	0.706	0.562	0.588
Silvio_Berlusconi_0001.pgm	1492	33	0.812	0.647	0.625	0.588	0.375	0.176

Table 5.2.4 Class wise accuracy LFW dataset with VGGFACE features

Name	Label	NOS	SVM_V	SVM_CV	COS_V	COS_CV
George_W_Bush_0001.pgm	533	530	0.981	0.985	0.928	0.921
Colin_Powell_0001.pgm	310	236	0.975	0.983	0.941	0.958
Tony_Blair_0001.pgm	1589	144	0.944	0.875	0.778	0.653
Donald_Rumsfeld_0001.pgm	393	121	0.933	0.902	0.8	0.77
Gerhard_Schroeder_0001.pgm	538	109	0.852	0.927	0.741	0.855
Ariel_Sharon_0001.pgm	112	77	0.947	0.949	0.842	0.897
Hugo_Chavez_0001.pgm	631	71	0.914	0.889	0.971	0.917
Junichiro_Koizumi_0001.pgm	857	60	0.967	0.867	0.967	0.9
Jean_Chretien_0001.pgm	711	55	0.963	0.643	0.815	0.679
John_Ashcroft_0001.pgm	770	53	0.769	0.815	0.808	0.815
Jacques_Chirac_0001.pgm	657	52	0.846	0.731	0.808	0.577
Serena_Williams_0001.pgm	1469	52	0.808	0.962	0.731	0.923
Vladimir_Putin_0001.pgm	1629	49	0.833	0.8	0.875	0.84
Luiz_Inacio_Lula_da_Silva_0001.pgm	995	48	0.833	0.958	0.792	0.875
Gloria_Macapagal_Arroyo_0001.pgm	550	44	0.909	0.909	0.909	0.909
Arnold_Schwarzenegger_0001.pgm	117	42	0.762	0.714	0.714	0.524
Jennifer_Capriati_0001.pgm	722	42	0.667	0.81	0.714	0.619
Laura_Bush_0001.pgm	933	41	0.9	0.905	0.6	0.762
Lleyton_Hewitt_0001.pgm	981	41	0.95	0.762	0.85	0.81
Alejandro_Toledo_0001.pgm	43	39	0.842	0.9	0.684	0.65
Hans_Blix_0001.pgm	589	39	0.895	1.0	0.895	0.85
Nestor_Kirchner_0001.pgm	1190	37	0.944	0.842	0.889	0.789
Andre_Agassi_0001.pgm	79	36	0.944	0.778	0.722	0.778
Alvaro_Uribe_0001.pgm	65	35	0.882	0.889	0.588	0.722
Megawati_Sukarnoputri_0001.pgm	1083	33	0.875	0.882	0.875	0.824
Silvio_Berlusconi_0001.pgm	1492	33	0.75	0.824	0.688	0.529

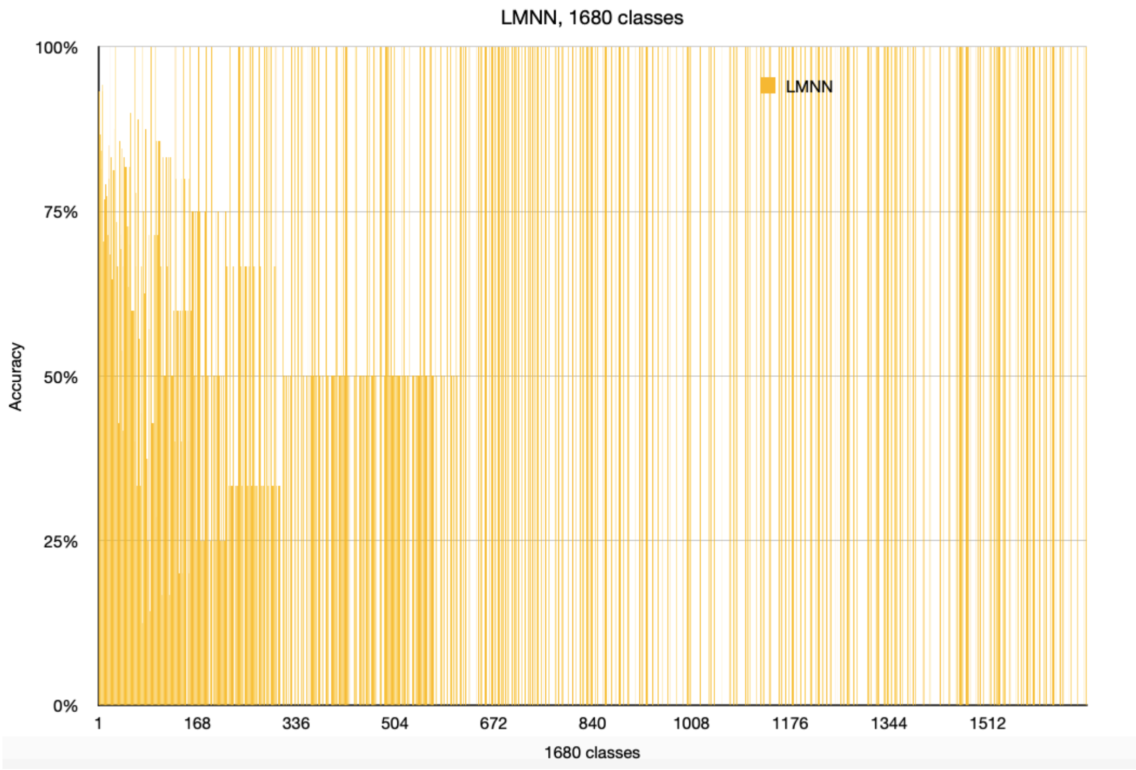

Fig. 5.2.1 Class-wise accuracy of 1680 LFW classes using LMNN

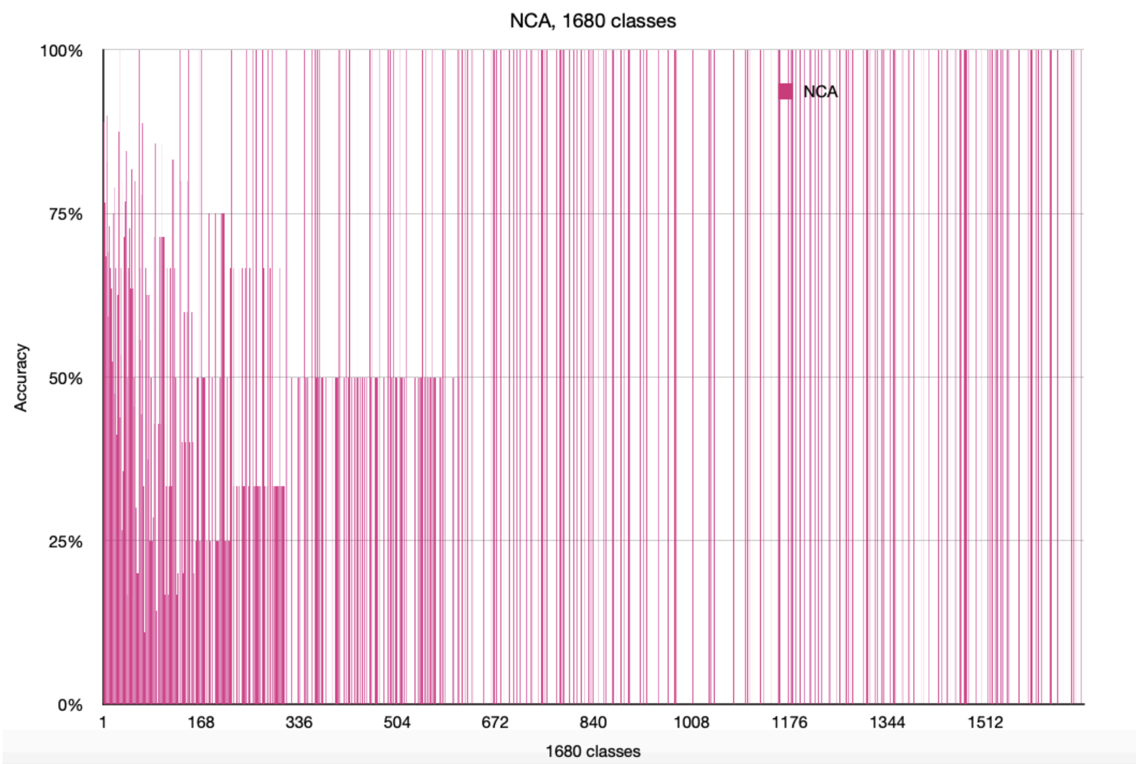

Fig. 5.2.2 Class-wise accuracy of 1680 LFW classes using NCA

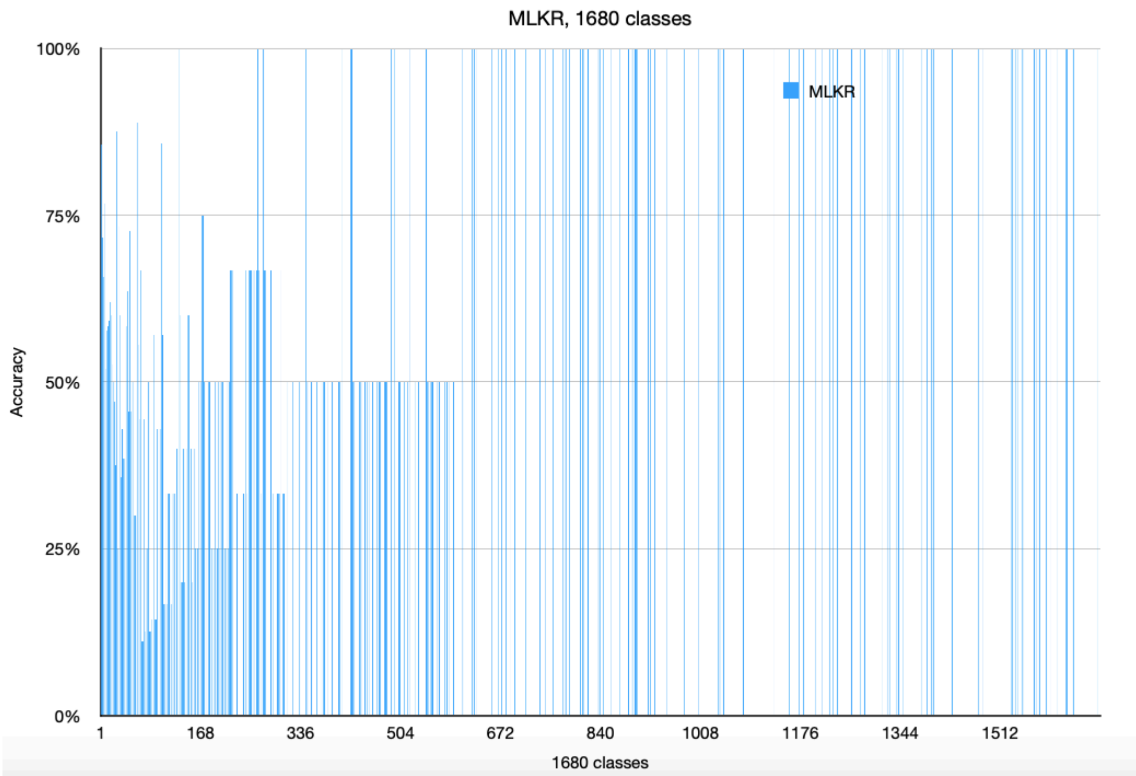

Fig. 5.2.3 Class-wise accuracy of 1680 LFW classes using MLKR

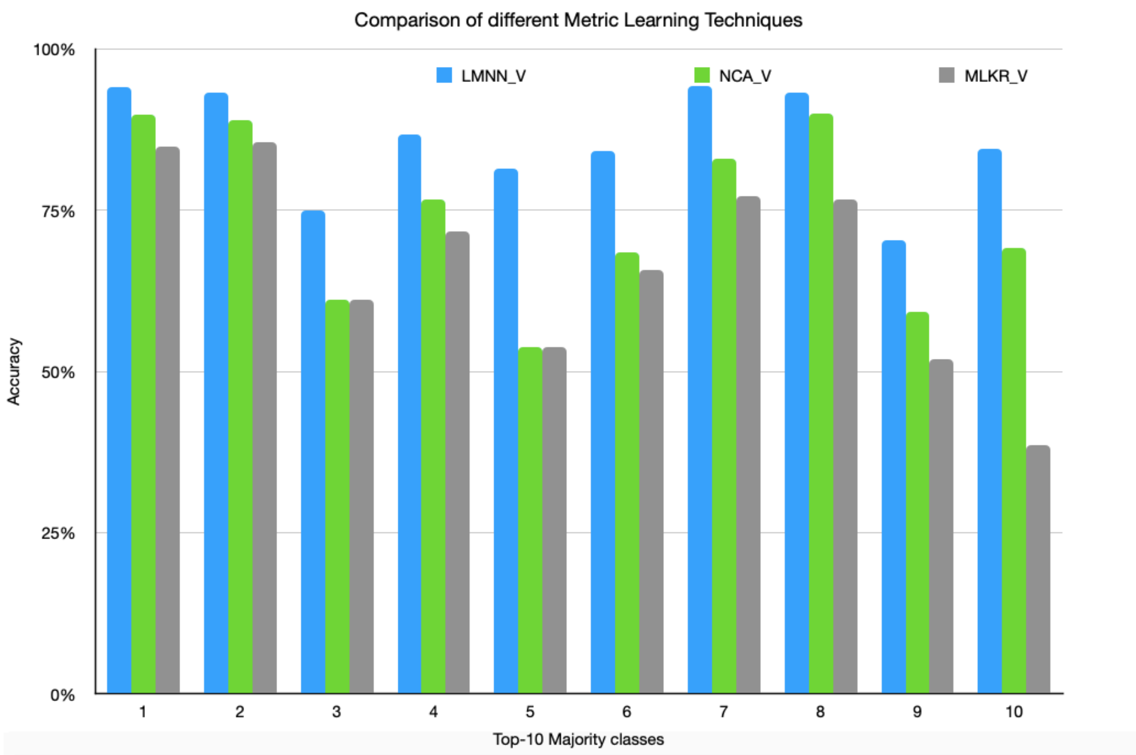

Fig. 5.2.4 Comparison of various metric learning techniques

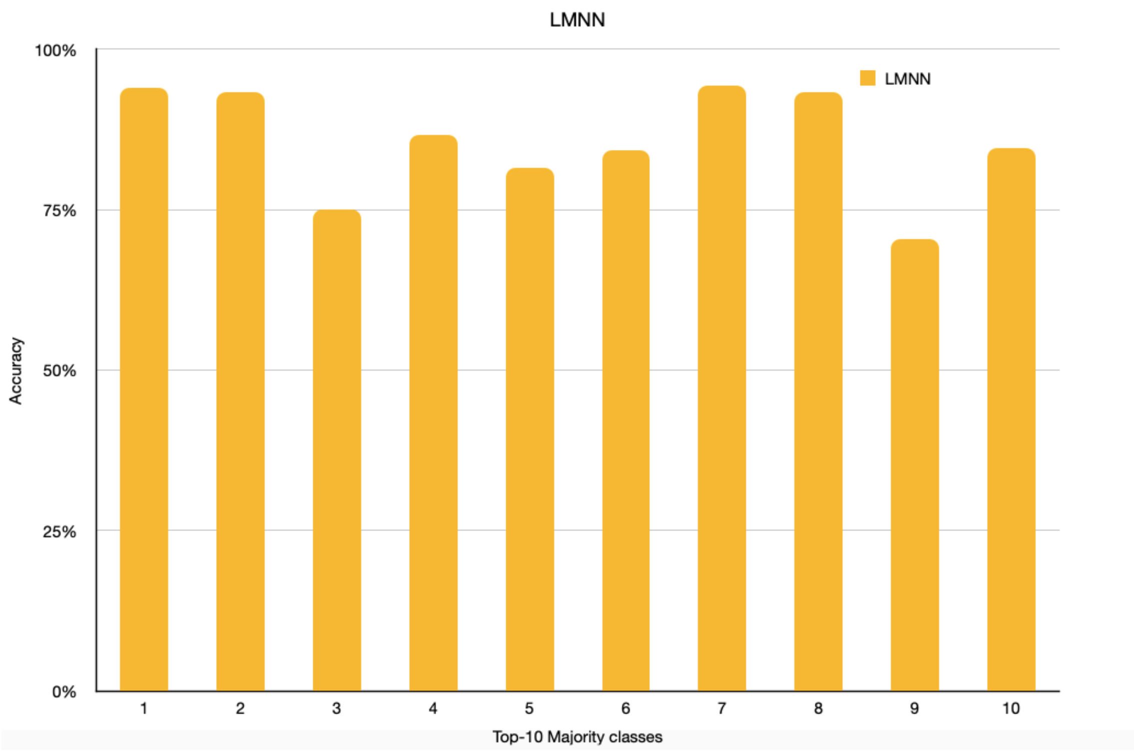

Fig. 5.2.5 Top-10 classes accuracy, LMNN

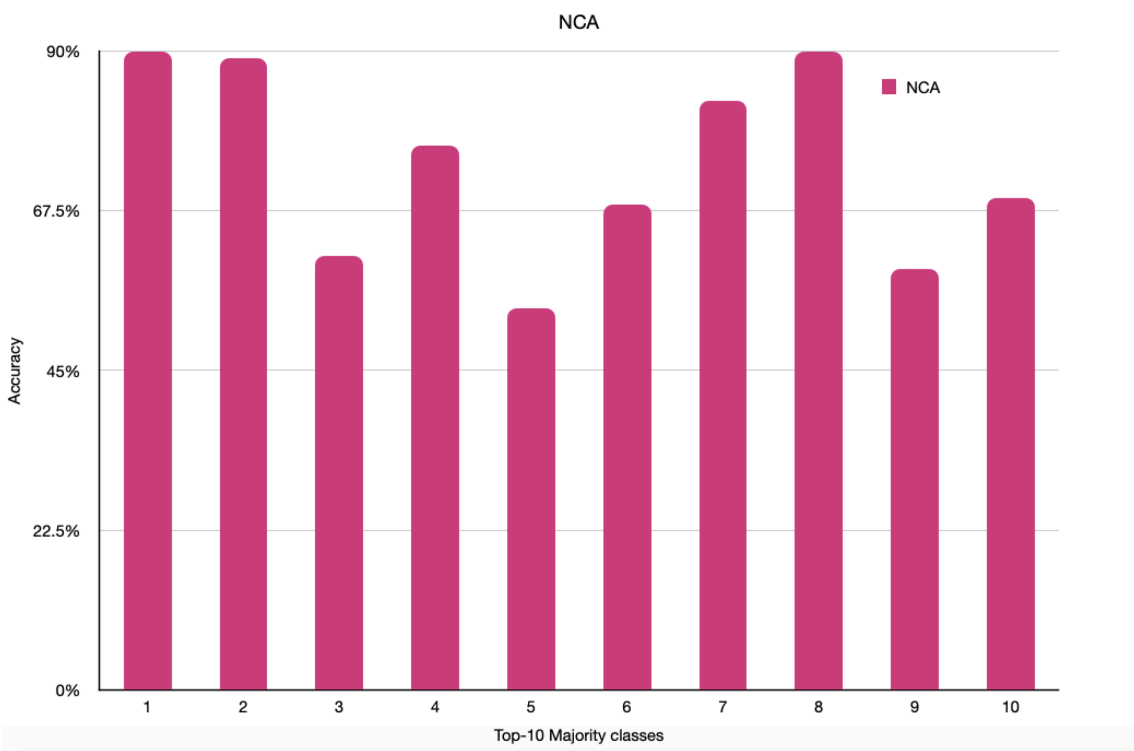

Fig. 5.2.6 Top-10 classes accuracy, NCA

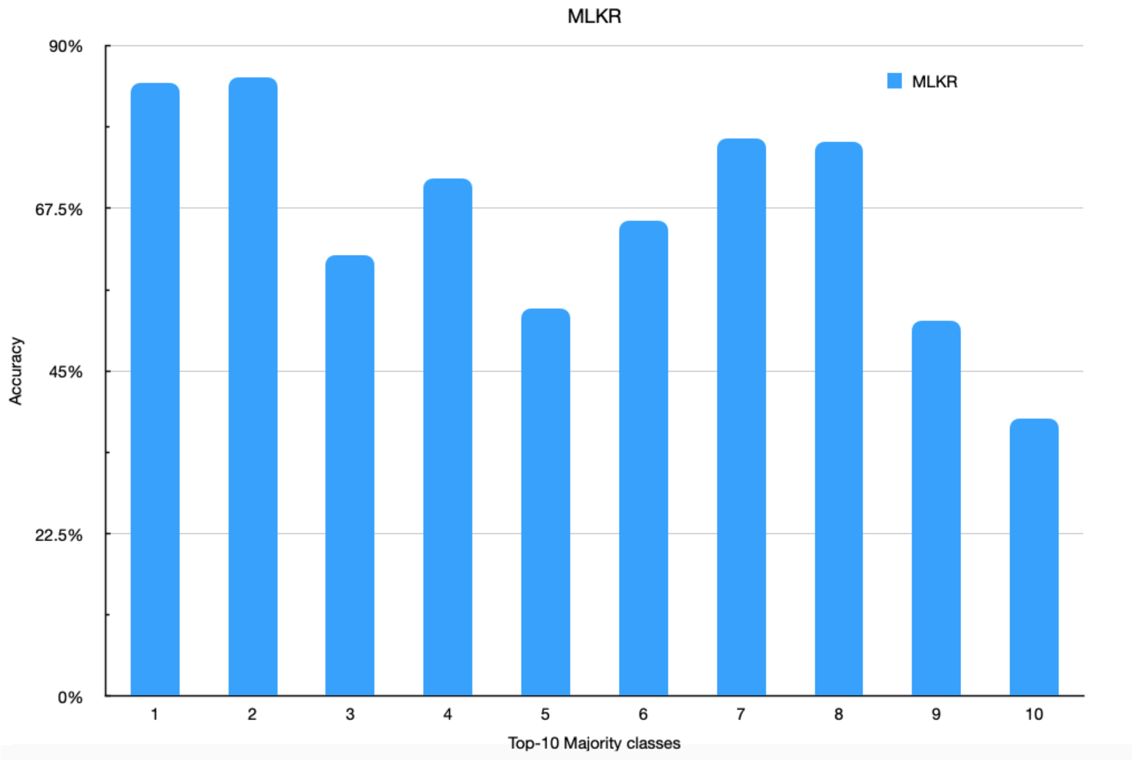

Fig. 5.2.7 Top-10 classes accuracy, MLKR

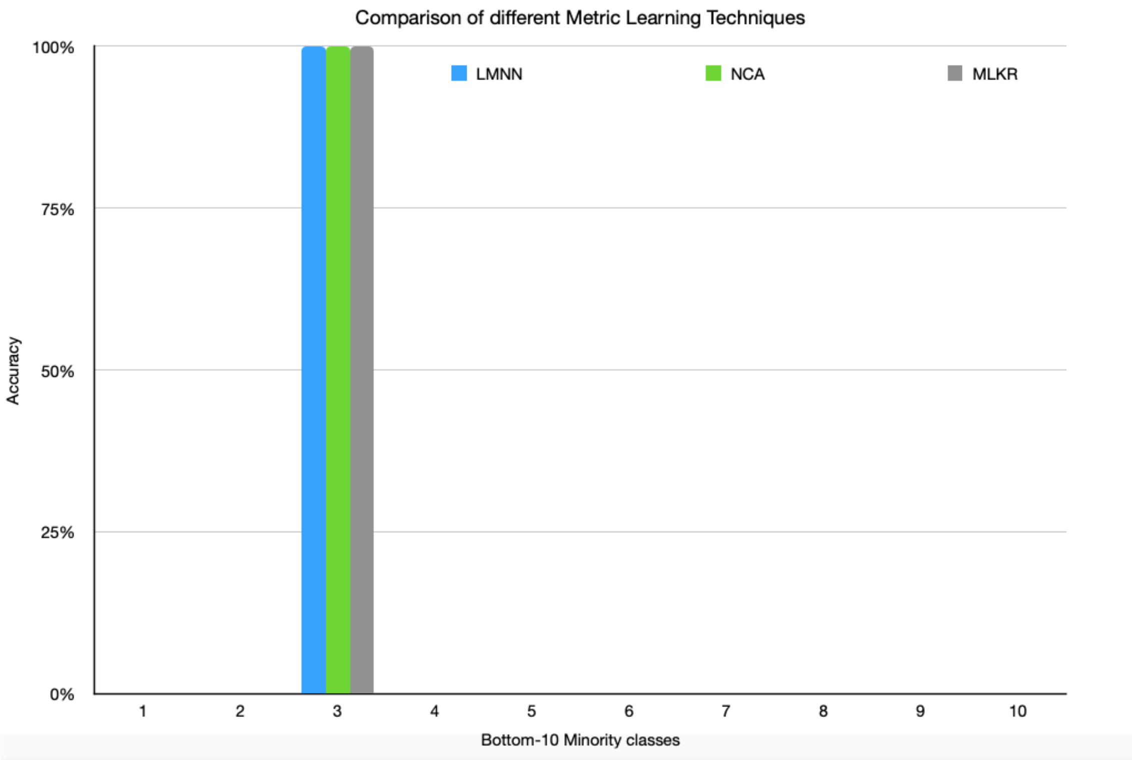

Fig. 5.2.8 Bottom-10 classes accuracy comparison, LMNN, NCA & MLKR

Table 5.2.5 Overall Performance Analysis with VGGFACE features

Method	AUC		F1-score		Accuracy	
	V	CV	V	CV	V	CV
VGG-Face + SVM[15]	0.654	0.635	0.287	0.263	0.554%	0.511%
VGG-Face + Cosine Similarity [16]	0.689	0.675	0.342	0.329	0.554%	0.519%
VGG-Face + LMNN	0.697	0.681	0.355	0.339	0.570%	0.536%
VGG-Face + NCA	0.634	0.625	0.233	0.225	0.444%	0.424%
VGG-Face + MLKR	0.600	0.591	0.174	0.167	0.365%	0.339%

5.3 SEM-4 RESULTS (VGGFACE FEATURES)

Table 5.3.1 Overall Performance Analysis with VGGFACE features with and without metric learning(2-way); Subset of majority classes

Name	Label	NOS	ML_V	ML_CV	COS_V	COS_CV
George_W_Bush_0001.pgm	533	530	0.955	0.928	0.928	0.921
Colin_Powell_0001.pgm	310	236	0.949	0.949	0.941	0.958
Tony_Blair_0001.pgm	1589	144	0.736	0.667	0.778	0.653
Donald_Rumsfeld_0001.pgm	393	121	0.9	0.852	0.8	0.77
Gerhard_Schroeder_0001.pgm	538	109	0.87	0.855	0.741	0.855
Ariel_Sharon_0001.pgm	112	77	0.895	0.872	0.842	0.897
Hugo_Chavez_0001.pgm	631	71	0.943	0.944	0.971	0.917
Junichiro_Koizumi_0001.pgm	857	60	0.967	0.9	0.967	0.9
Jean_Chretien_0001.pgm	711	55	0.852	0.786	0.815	0.679
John_Ashcroft_0001.pgm	770	53	0.769	0.852	0.808	0.815
Jacques_Chirac_0001.pgm	657	52	0.846	0.692	0.808	0.577
Serena_Williams_0001.pgm	1469	52	0.769	0.885	0.731	0.923
Vladimir_Putin_0001.pgm	1629	49	0.833	0.84	0.875	0.84
Luiz_Inacio_Lula_da_Silva_0001.pgm	995	48	0.75	0.875	0.792	0.875
Gloria_Macapagal_Arroyo_0001.pgm	550	44	0.773	0.909	0.909	0.909
Jennifer_Capriati_0001.pgm	722	42	0.714	0.667	0.714	0.619
Arnold_Schwarzenegger_0001.pgm	117	42	0.714	0.619	0.714	0.524
Laura_Bush_0001.pgm	933	41	0.7	0.81	0.6	0.762
Lleyton_Hewitt_0001.pgm	981	41	0.95	0.81	0.85	0.81
Alejandro_Toledo_0001.pgm	43	39	0.579	0.75	0.684	0.65
Hans_Blix_0001.pgm	589	39	0.947	0.95	0.895	0.85
Nestor_Kirchner_0001.pgm	1190	37	1.0	1.0	0.889	0.789
Andre_Agassi_0001.pgm	79	36	0.778	0.778	0.722	0.778
Alvaro_Uribe_0001.pgm	65	35	0.647	0.667	0.588	0.722
Megawati_Sukarnoputri_0001.pgm	1083	33	0.938	0.941	0.875	0.824
Silvio_Berlusconi_0001.pgm	1492	33	0.875	0.588	0.688	0.529

Table 5.3.2 Overall Performance Analysis with VGGFACE features with and without metric learning(2-way); Subset of minority classes

Name	Label	NOS	ML_V	ML_CV	COS_V	COS_CV
LeBron_James_0001.pgm	942	5	1	0.667	0.5	0.667
Steffi_Graf_0001.pgm	1505	5	0.5	0	0	0
Arnoldo_Aleman_0001.pgm	118	5	0.5	0	0	0
Ai_Sugiyama_0001.pgm	22	5	0.5	0.667	0	0
Emanuel_Ginobili_0001.pgm	438	5	0.5	0.667	0	0
Grant_Hackett_0001.pgm	559	5	1	0.667	0.5	1
John_Stockton_0001.pgm	800	5	0.5	0	0	0
Allyson_Felix_0001.pgm	62	5	0.5	0.333	0	0
Gregg_Popovich_0001.pgm	565	5	0.5	0.667	0	0.333
George_Lopez_0001.pgm	524	5	0.5	0	0	0
Joseph_Biden_0001.pgm	836	5	1	0.333	0.5	0.333
Thabo_Mbeki_0001.pgm	1545	5	0.5	0	0	0
Clara_Harris_0001.pgm	301	5	1	0.667	0.5	1
Carrie-Anne_Moss_0001.pgm	242	5	1	0	0	0
Pamela_Anderson_0001.pgm	1225	5	0.5	0.667	0	0.333
Cruz_Bustamante_0001.pgm	321	5	0.5	0.333	0	0.333
Catherine_Deneuve_0001.pgm	246	5	1	1	0.5	0.667
Chanda_Rubin_0001.pgm	256	5	1	0.667	0	0
Nadia_Petrova_0001.pgm	1167	5	1	0	0.5	0
Madonna_0001.pgm	1003	5	1	0.333	0.5	0.333
Bertie_Ahern_0001.pgm	152	5	0.5	0	0	0
Martin_McGuinness_0001.pgm	1059	5	0.5	0	0	0
Harry_Schmidt_0001.pgm	595	4	1	0.5	0.5	0.5
Hipolito_Mejia_0001.pgm	619	4	1	1	0.5	1
Heather_Mills_0001.pgm	602	4	0.5	0.5	0	0.5
Nan_Wang_0001.pgm	1170	4	1	1	0.5	1

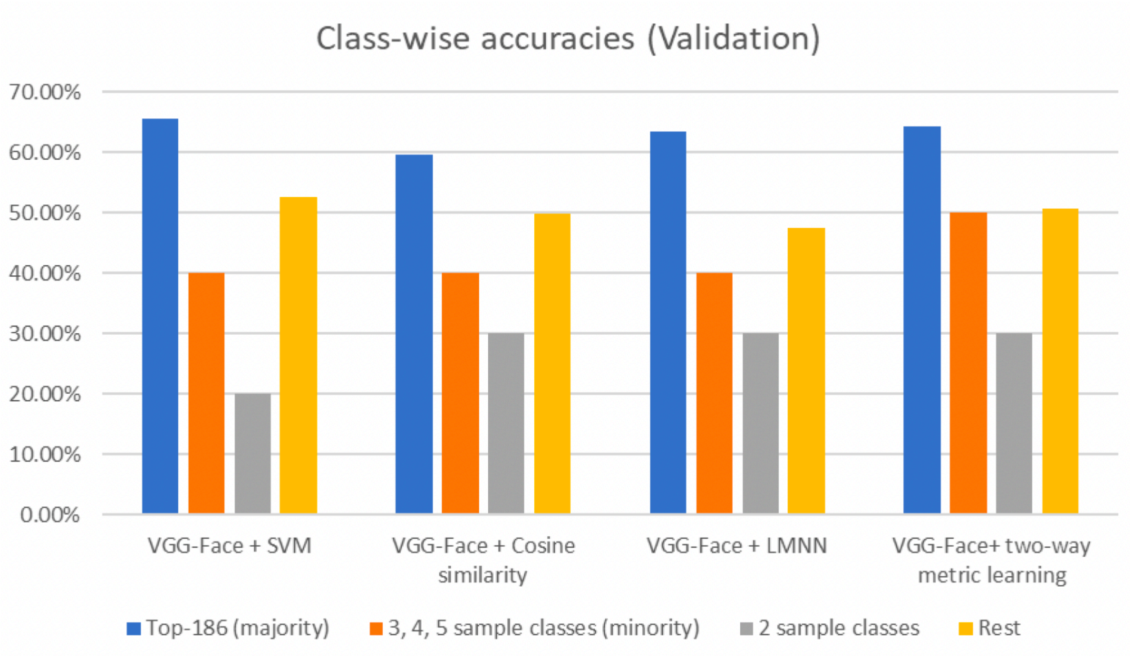

Fig. 5.3.1 Class-wise accuracies using 2-way metric learning [30]

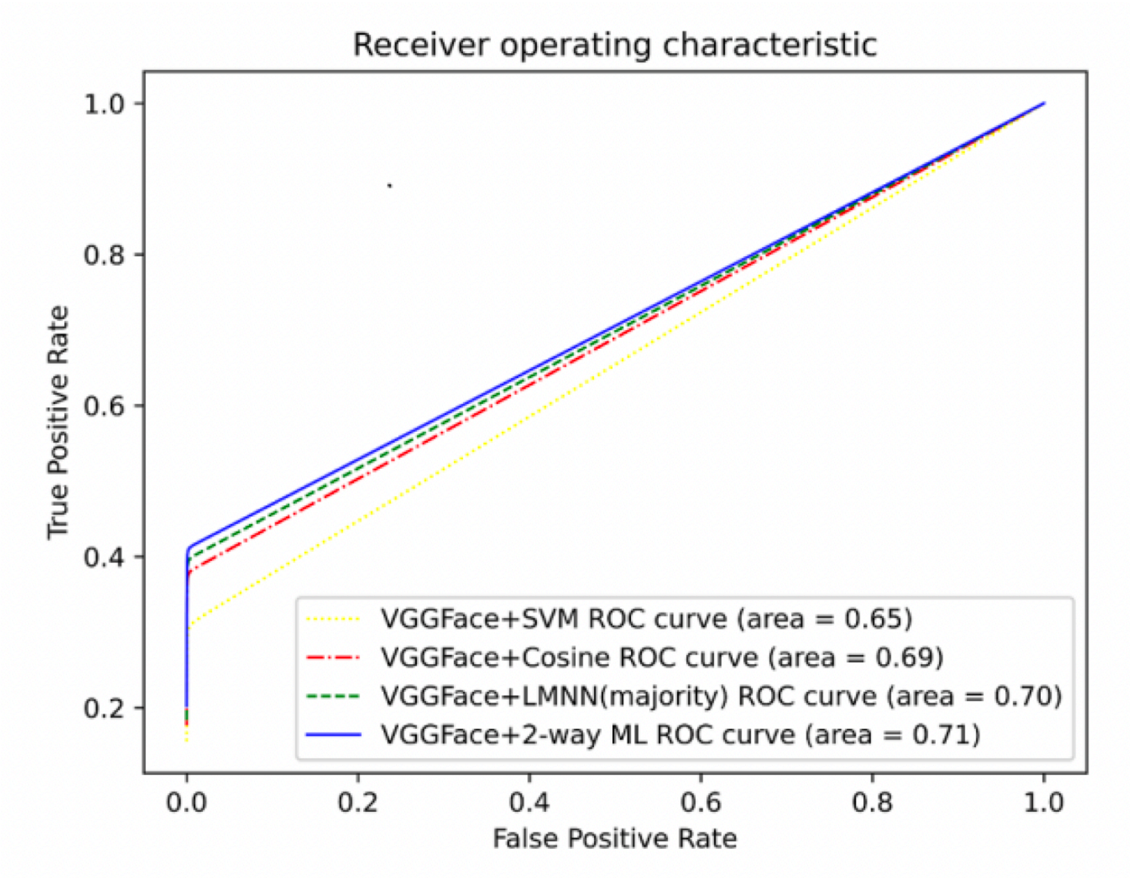

Fig. 5.3.2 ROC curve of proposed solutions [30]

CHAPTER-6 CONCLUSION

I ran SVM and cosine similarity modules over several datasets and one large dataset both before and after metric learning. After metric learning the class wise accuracy of various samples is increased by 10-20% and also reduced by around 5% for a few samples. But overall the accuracy and f1-scores are improved and thus it proved to be a better approach.

I successfully extracted the feature vectors from various Deep learning models and like VGGFace. Then while classifying them when I followed my proposed method on this feature set the accuracy of majority classes was risen significantly and minority classes marginally.

In the final semester we have successfully devised a two-way metric learning technique which helps us in not only improving the performance for majority classes but for minority classes as well. It is a novel method and can be applied before doing classification.

CHAPTER-7 REFERENCES

[1] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), San Diego, CA, USA, 2005, pp. 886-893 vol. 1, doi: 10.1109/CVPR.2005.177.

[2] Kilian Q. Weinberger and Lawrence K. Saul. 2009. Distance Metric Learning for Large Margin Nearest Neighbor Classification. J. Mach. Learn. Res. 10 (12/1/2009), 207–244.

[3] K. Q. Weinberger and L. K. Saul. Fast solvers and efficient implementations for distance metric learning. In Proceedings of the Twenty Fifth International Conference on Machine learning, pages 1160–1167, Helsinki, Finland, 2008.

[4] Gary B. Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller. Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments. University of Massachusetts, Amherst, Technical Report 07-49, October, 2007.

[5] https://en.wikipedia.org/wiki/Large_margin_nearest_neighbor

[6] C. Sanderson, B.C. Lovell. Multi-Region Probabilistic Histograms for Robust and Scalable Identity Inference. ICB 2009, LNCS 5558, pp. 199-208, 2009.

[7] https://www.analyticsvidhya.com/blog/2019/09/feature-engineering-images-introduction-hog-feature-descriptor/

[8] Zhou, Erjin & Cao, Zhimin & Yin, qi. (2015). Naive-Deep Face Recognition: Touching the Limit of LFW Benchmark or Not?

[9] P. Carcagni, M. Del Coco, P. Mazzeo, A. Testa, C. Distante, Features descriptors for demographic estimation: a comparative study, VAAM (2014)

[10] Szegedy, C., Liu, W., Jia, Y., et al.: Going deeper with convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–9 (2015)

[11] Ben Fredj, H., Bouguezzi, S. & Souani, C. Face recognition in unconstrained environment with CNN. Vis Comput (2020). https://doi.org/10.1007/s00371-020-01794-9

[12] Lu, Chaochao & Tang, Xiaoou. (2014). Surpassing Human-Level Face Verification Performance on LFW with GaussianFace.

[13] Cortes C. and Vapnik V., "Support vector networks", Machine Learning, 20:1--25, 1995.

[14] Taiwan colour & imaging technology (tcit). http://www.tcit-us.com/.

[15] Dadi, Harihara Santosh, and GK Mohan Pillutla. "Improved face recognition rate using HOG features and SVM classifier." *IOSR Journal of Electronics and Communication Engineering* 11, no. 04 (2016): 34-44.

[16] Chen, Dong, Xudong Cao, Fang Wen, and Jian Sun. "Blessing of dimensionality: High-dimensional feature and its efficient compression for face verification." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3025-3032. 2013.

[17] Erik Learned-Miller, Gary B. Huang, Aruni RoyChowdhury, Haoxiang Li, and Gang Hua. Labeled Faces in the Wild: A Survey. In Advances in Face Detection and Facial Image Analysis, edited by Michal Kawulok, M. Emre Celebi, and Bogdan Smolka, Springer, pages 189-248, 2016.

[18] Huang, Gary & Mattar, Marwan & Berg, Tamara & Learned-Miller, Eric. (2008). Labeled Faces in the Wild: A Database forStudying Face Recognition in Unconstrained Environments. Tech. rep..

[19] Y. Sun, L. Ding, X. Wang, and X. Tang. Deepid3: Face recognition with very deep neural networks. CoRR, abs/1502.00873, 2015.

[20] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf. Web-scale training for face identification. In Proc. CVPR, 2015.

[21] C. Lu and X. Tang. Surpassing human-level face verification performance on lfw with gaussian-face. AAAI, 2015.

[22] 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

[23] Susan, Seba, and Ashu Kaushik. "Weakly supervised metric learning with majority classes for large imbalanced image dataset." In *Proceedings of the 2020 the 4th International Conference on Big Data and Internet of Things*, pp. 16-19. 2020.

[24] F. Schroff, D. Kalenichenko and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, 2015, pp. 815-823, doi: 10.1109/CVPR.2015.7298682.

[25] Susan, Seba, Dhaarna Sethi, and Kriti Arora. "CW-CAE: pulmonary nodule detection from imbalanced dataset using class-weighted convolutional autoencoder." In *International Conference on innovative computing and communications*, pp. 825-833. Springer, Singapore, 2021.

[26] Saini, Manisha, and Seba Susan. "Bag-of-Visual-Words codebook generation using deep features for effective classification of imbalanced multi-class image datasets." *Multimedia Tools and Applications* 80, no. 14 (2021): 20821-20847.

[27] Saini, Manisha, and Seba Susan. "Deep transfer with minority data augmentation for imbalanced breast cancer dataset." *Applied Soft Computing* 97 (2020): 106759.

[28] Susan, Seba, and Amitesh Kumar. "The balancing trick: Optimized sampling of imbalanced datasets—A brief survey of the recent State of the Art." *Engineering Reports* 3, no. 4 (2021): e12298.

[29] Susan, Seba, and Amitesh Kumar. "DST-ML-EkNN: data space transformation with metric learning and elite k-nearest neighbor cluster formation for classification of imbalanced datasets." In *Advances in Artificial Intelligence and Data Engineering*, pp. 319-328. Springer, Singapore, 2021.

[30] Kaushik, Ashu, and Seba Susan. "TWO-WAY METRIC LEARNING WITH MAJORITY AND MINORITY SUBSETS FOR CLASSIFICATION OF LARGE EXTREMELY IMBALANCED FACE DATASET." *Jordanian Journal of Computers and Information Technology (JJCIT)* 7, no. 04 (2021).

[31] Kaushik, Ashu, and Seba Susan. "Metric Learning with Deep Features for Highly Imbalanced Face Dataset." In *International Conference on Innovative Computing and Communications*, pp. 639-646. Springer, Singapore, 2022.

CHAPTER-8 LIST OF PUBLICATIONS

1. Susan Seba, and Ashu Kaushik. "Weakly supervised metric learning with majority classes for large imbalanced image dataset." In *Proceedings of the 2020 the 4th International Conference on Big Data and Internet of Things*, pp. 16-19. 2020.

2. Kaushik, Ashu, and Seba Susan. "Metric Learning with Deep Features for Highly Imbalanced Face Dataset." In *International Conference on Innovative Computing and Communications*, pp. 639-646. Springer, Singapore, 2022.

3. Kaushik, Ashu, and Seba Susan. "TWO-WAY METRIC LEARNING WITH MAJORITY AND MINORITY SUBSETS FOR CLASSIFICATION OF LARGE EXTREMELY IMBALANCED FACE DATASET." *Jordanian Journal of Computers and Information Technology (JJCIT)* 7, no. 04 (2021).