Project Report

SEMESTER – 4

SUBMITTED BY:

SASIKANTH KOTTI, NIKHILA DHULIPALLA AND ADHUN THALEKKARA

Have we met before: ID'ing face

Introduction:

Many techniques in the fields of machine object recognition and pattern recognition rely heavily on face detection. Face domain can be categorized broadly into two – Identification and Verification. Face identification is the process of determining a person's identity based on a facial picture. Whereas facial verification is the process of confirming a purported identification based on a face photograph and either approving or rejecting the claim (one-to-one matching). One of the most groundbreaking achievements in the previous few years has been automated face recognition. Particularly with programmes like India's Aadhar and Apple's Face ID, which have opened up a slew of new research possibilities. Many obstacles arose as a result of the developments, particularly in areas such as plastic surgery and disguise.

Convolutional Neural Networks (CNN) as well as other classification models use a deep metric learning-based loss function to learn discriminative embeddings. The loss function seeks to close the gap between the embeddings of the same classes in the output manifold. The dissimilarity score between the two images is computed using a simple distance calculation in this embedding space. There are many applications makings use of deep metric learning algorithms - person reidentification, 3D object retrieval, biometric recognition, robot perception, patch matching and object recognition.

Tasks carried out as part of our project:

Task1: Implemented and reproduced existing results of the paper "On Matching Faces with Alterations due to Plastic Surgery and Disguise" including the LFW dataset.

Task2: Reproduced the results of the paper "Escaping the Big Data Paradigm with Compact Transformers".

Task3: Implemented and reproduced existing results of the paper "On Learning Density Aware Embeddings".

Task4: Proposed the novel architecture – LightAttentionCNN29.

Task5: We have carried out various experiments using different loss functions for different models. We have either used pre-trained models or fine-tuned them for the experiments.

Dataset details:

As part of the project, we have used three different datasets:

1. DFW 2018 dataset

2. LFW dataset
3. Plastic Surgery Face Database

Protocols defined for each dataset are as below:

1. DWF 2018 dataset
    - Protocol 1 (Impersonation): Genuine set contains pairs having '1', and imposter set contains pairs with '3'.
    - Number of samples in the train: 4997
    - Number of samples in the test: 25046
    - The mask matrices contain values belonging to {0,1,2,3,4}. Here,
        - 0 - No use
        - 1 - Genuine Validation
        - 2 - Genuine Disguise
        - 3 - Imposter Impersonator
        - 4 - Cross-subject Imposter
    - That is, the value of the element (i,j) specifies whether the pair created by the ith and the jth image is of no use, a genuine validation pair, a genuine disguise pair, an imposter impersonator pair, or an impostor cross-subject pair.
    - The ordering of images is the same as provided with the dataset in the "Training_data_face_name.txt" and "Testing_data_face_name.txt" text files. training_data_mask_matrix.txt contains a 3386x3386 matrix and testing_data_mask_matrix.txt contains a 7771x7771 matrix.
    - Note that both these matrices are symmetric matrices since pairs (i,j) and (j,i) refer to the same pair.
    - The training and testing mask matrices can be used for extracting the relevant pairs/scores by using Protocol 1.

2. LFW dataset
    - LFW dataset has 13233 images, 5749 people and 1680 identities with two or more images.
    - Dataset split into Train and Test. The train has 1180 identities and the test has 500 identities.
    - Some identities have just a single face image.
    - Test contains Probe & Gallery with non-overlapping face images.
    - Train dataset, again split (stratify split) into Train and Validation in the ratio of 70-30 during fine-tuning.

3. Plastic Surgery Face dataset:
    - Number of samples in train: 648
    - Number of samples in validation: 72
    - Number of samples in test: 180

Experiments carried out:

1. On LFW Dataset:
    - We used this dataset for face identification tasks.
    - Models used:
        - LightCNN29 pre-trained init (pre-trained and fine-tuned with different loss functions combinations)
        - Compact Convolution Transformer (pre-trained and fine-tuned with different loss functions combinations)
    - Loss functions used:
        - ArcFace
        - Contrastive + ArcFace
        - Contrastive + Variance
        - SemiHard Triplet
        - SupCon with SNR Distance

2. On Plastic Surgery Face Dataset:
    - We used this dataset for face verification tasks.
    - Models used:
        - LightCNN29 (fine-tuned)
        - Loss functions used:
        - Contrastive

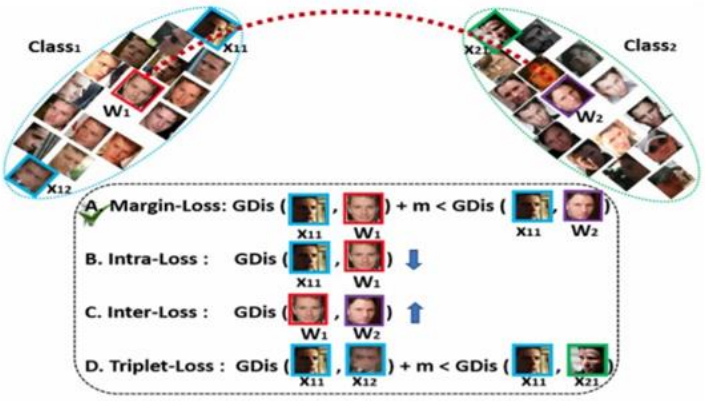
3. On DFW Dataset:
    - We used this dataset for face verification tasks.
    - Models used:
        - LightCNN29 (fine-tuned)
        - LAttCNN29 (Light Attention CNN 29)
    - Loss functions used:
        - Contrastive
        - Center + Angular
        - Cosine Augmented Center loss
        - ArcFace + Variance Loss

All about the Loss Functions implemented:

We employed a variety of loss functions and conducted extensive experiments to better understand how they behaved when applied to different types of models (pre-trained or fine-tuned). The loss functions we used are as follows:

1. Additive Angular Margin Loss (ArcFace): ArcFace has a clear geometric interpretation due to the exact correspondence to the geodesic distance on the hypersphere. In other words, the loss function will maximize the distance between the cluster centers as they are a part of the hypersphere.

$$L_3 = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{s(\cos(\theta_{y_i}+m))}}{e^{s(\cos(\theta_{y_i}+m))}+\sum_{j=1,j\neq y_i}^{n}e^{s\cos\theta_j}}$$



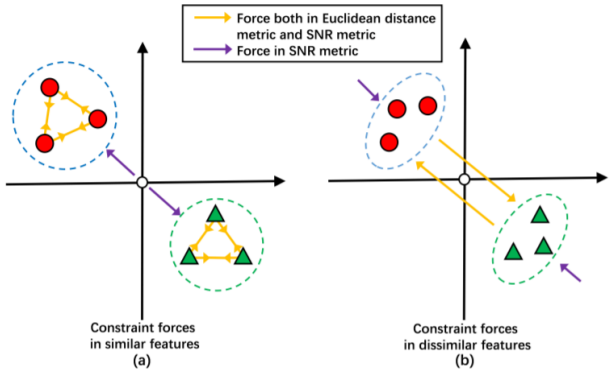
2. SupCon - Supervised Contrastive Loss : Extended self-supervised batch contrastive approach into a fully-supervised setting, which helps to leverage the label information effectively. Clusters of points belonging to the same class are pulled together in embedding space, while simultaneously pushing apart clusters of samples from different classes.

$$\mathcal{L}_{out}^{sup} = \sum_{i\in I}\mathcal{L}_{out,i}^{sup} = \sum_{i\in I}\frac{-1}{|P(i)|}\sum_{p\in P(i)}\log\frac{\exp(\boldsymbol{z}_i\cdot\boldsymbol{z}_p/\tau)}{\sum_{a\in A(i)}\exp(\boldsymbol{z}_i\cdot\boldsymbol{z}_a/\tau)}$$

3. Contrastive Loss: Takes L2 distance between pairs then it maximizes the distance between negative pairs and minimizes the distance between positive pairs.

4. SNRDistance: Based on Signal-to-Noise Ratio(SNR) for measuring the similarity of image pairs for deep metric learning

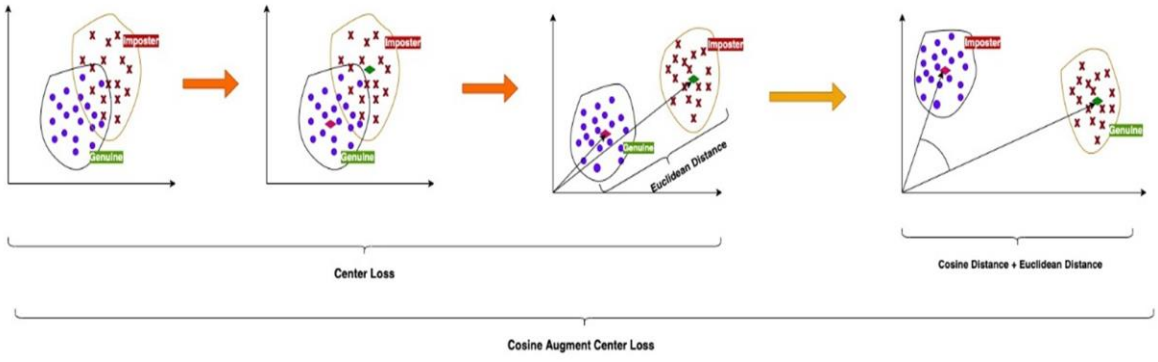
$$d_S(\boldsymbol{h}_i,\boldsymbol{h}_j) = \frac{1}{\mathrm{SNR}_{ij}} = \frac{var(\boldsymbol{n}_{ij})}{var(\boldsymbol{h}_i)}.$$



We used these loss functions individually and as combinations along with LightCNN29 and LAttCNN29(Light Attention CNN29 - proposed architecture). Also, we proposed one loss function named Cosine Augmented Center Loss which we will discuss in the next section.

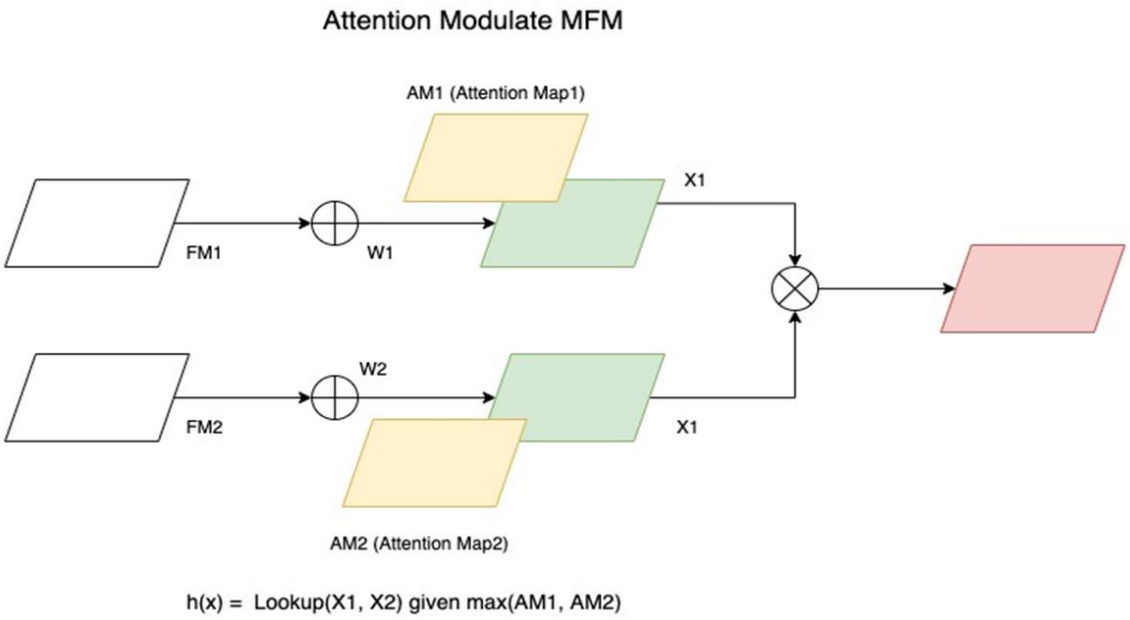
Cosine Augmented Center Loss:

The proposed Cosine Augmented Center Los is a combination of center loss along with Center Distance + Euclidean Distance loss. In Center Loss it will increase the intra-class distance and decrease the inter class distance. Cosine Distance will tend to increase the angles between the clusters and euclidean distance helps to increase the distance between the cluster centers. Combining these two the clusters should increase the distance between the clusters more.



Attention Modulated MFM:

We also tried to improve MFM (Max Feature Map) by adding an attention layer to take the information rich features from them and combine it. The below figure shows the setup of Attention Modulated MFM.



FM1 and FM2 are the layers and w1 and w2 are the weights which get multiplied and result in X1 and X2. We generate the attention layers of the same AM1(Attention Map) and AM2 using SimAM which is a simple parameter attention module which can create attention layers without training.

$$h(x) = Lookup(x1,x2) \text{ given } max(AM1, AM2)$$

The above equation is used to find the final layer. For each 1x1 cell it will find out for which attention map it has the maximum value. Then the X value corresponding to that AM and indices will be the result in the h(x) for those indices.
By this way we can filter out the max important features and not just the max pool which can produce a new information rich layer.

Results:

On LFW Dataset:

LightCNN29 Pre-trained Init				
		Identification Accuracy		
Model	Loss Function	Rank1	Rank5	Rank10
LightCNN29	ArcFace	54.60%	68.30%	71.80%
	Contrastive + ArcFace	75.60%	89.80%	93.30%
	Contrastive + Variance	90.60%	93.80%	94.70%
	SemiHard Triplet	72.00%	84.00%	88.80%
	SupCon with SNR Dist	86.60%	91.30%	92.80%
Compact Convolutional Transformer(CCT)	Contrastive + Variance	100%	100%	100%

Fine-tuned				
		Identification Accuracy		
Model	Loss Function	Rank1	Rank5	Rank10
LightCNN29	Contrastive + Variance	96.40%	98.30%	98.50%
Compact Convolutional Transformer(CCT)	Contrastive + Variance	4%	13.40%	19.10%

On Plastic Surgery Face Database:

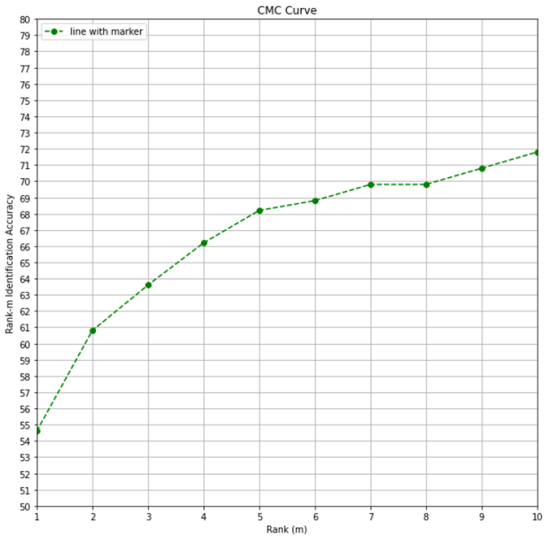
Fine-tuned					
		Identification Accuracy			Verification Accuracy
Model	Loss Function	Rank1	Rank5	Rank10	1% FAR
LightCNN29	Contrastive	95.55%	99.44%	99.44%	70%

On Plastic Surgery Face Database:

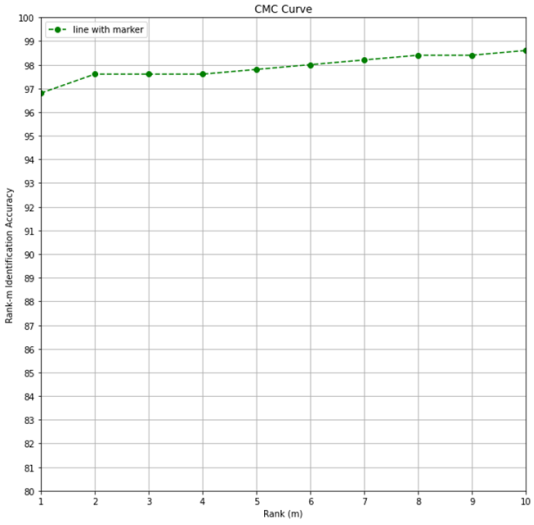
Fine-tuned		
		Verification Accuracy
Model	Loss Function	1% FAR
LightCNN29	Contrastive	49%
LAttCNN29 (Light Attention CNN 29)	Center + Angular	45%
LAttCNN29 (Light Attention CNN 29)	Cosine Augmented Center loss	23%
LAttCNN29 (Light Attention CNN 29)	ArcFace + Variance Loss	49%

CMC, ROC and FAR:

1. **On LFW dataset:**
    a. **Using Pre-trained models with different loss combinations:**
        - ArcFace (LightCNN29)

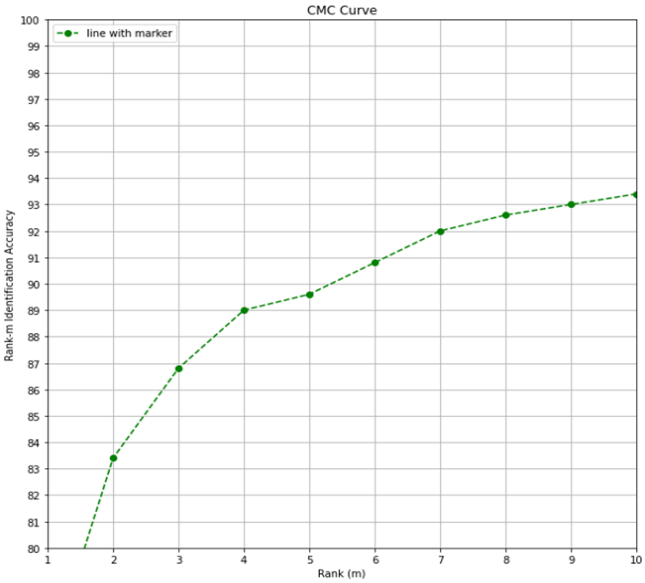


Rank1 Identification accuracy for LightCNN29 using a pre-trained model with only Arc Face Loss(%) 54.6.

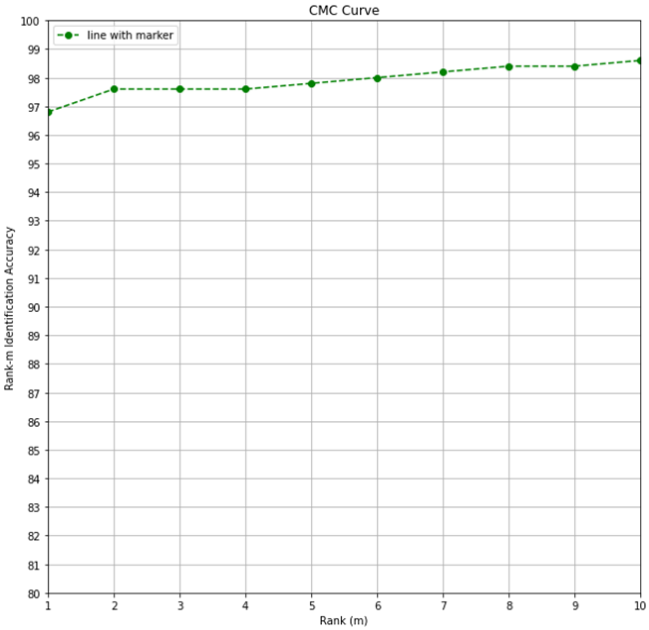


Rank1 Identification accuracy for LightCNN29 using a pre-trained model before training with ArcFace Losses(%) 96.8.

- Contrastive + ArcFace (LightCNN29)

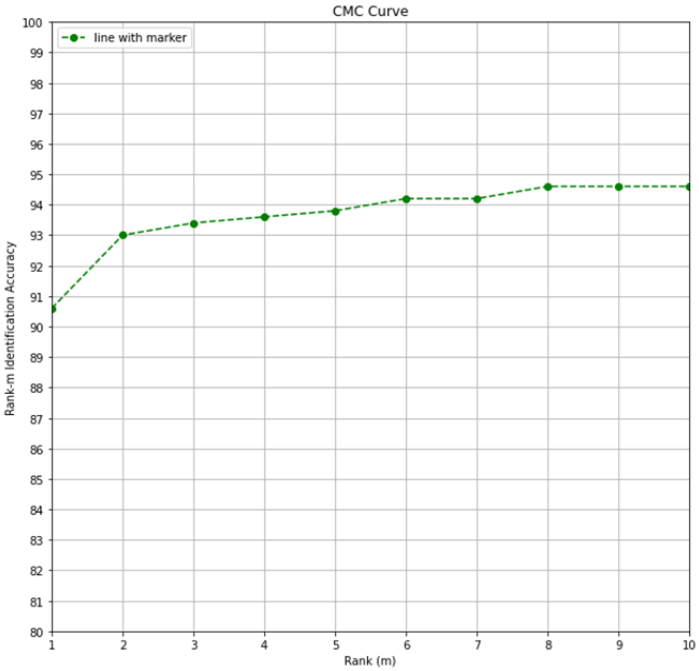


Rank1 Identification accuracy for LightCNN29 using a pre-trained model with Contrastive & Arc Face Losses(%) 75.6.

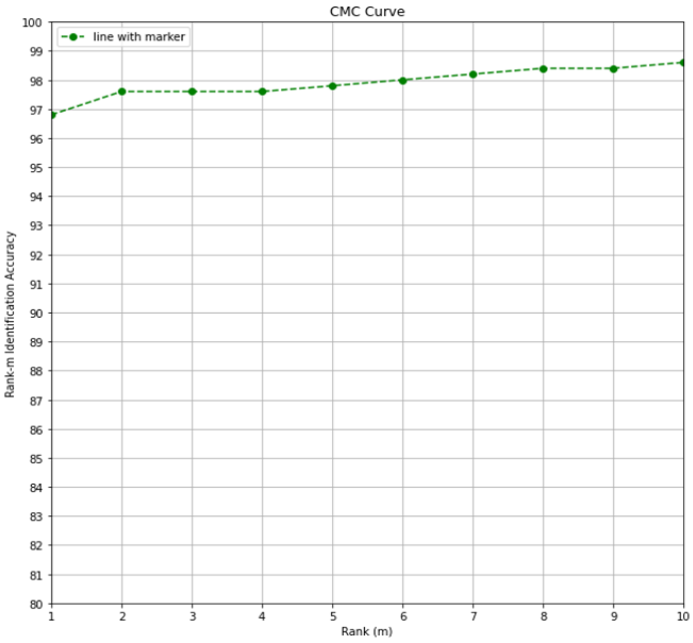


Rank1 Identification accuracy for LightCNN29 using a pre-trained model before training with Contrastive & ArcFace Losses(%) 96.8.

- Contrastive + Variance (LightCNN29)

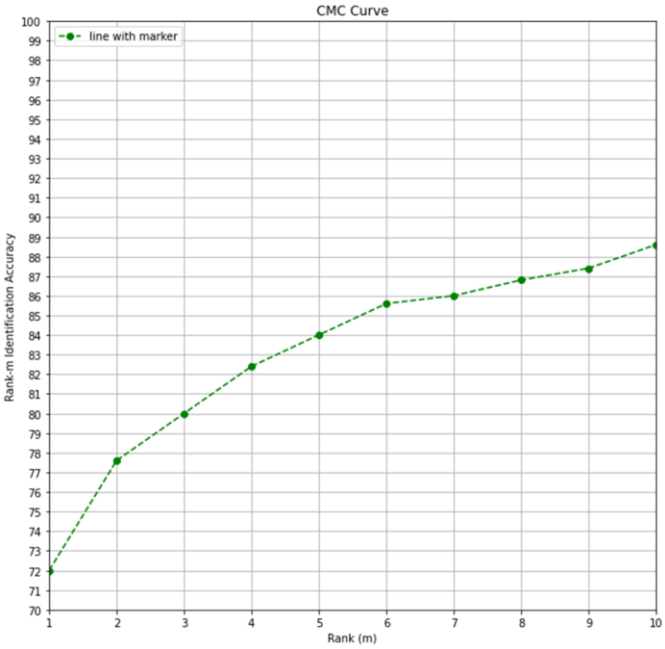


Rank1 Identification accuracy for LightCNN29 using a pre-trained model with Contrastive & Variance Losses(%) 90.60000000000001.



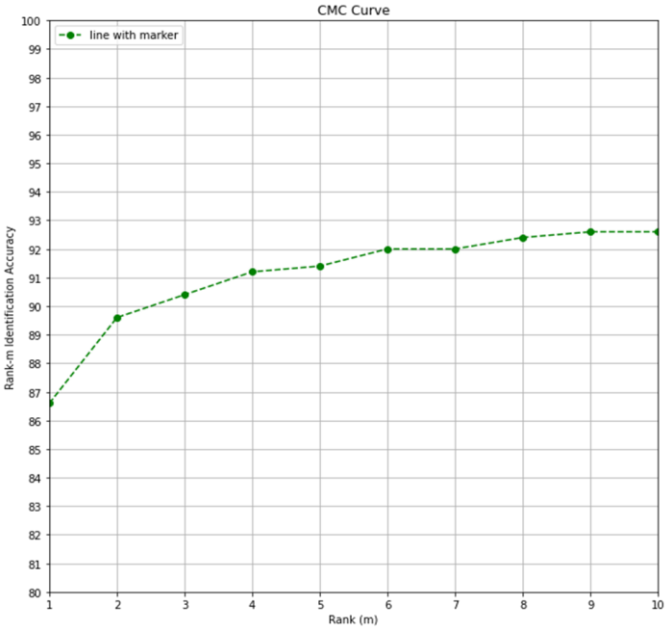
Rank1 Identification accuracy for LightCNN29 using a pre-trained model before training with Contrastive & Variance Losses(%) 96.8.

- SemiHard Triplet (LightCNN29)



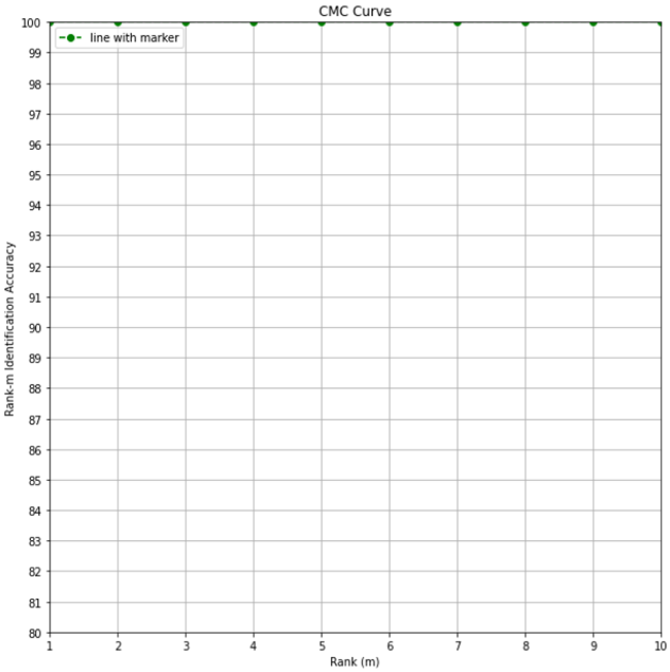
Rank1 Identification accuracy for LightCNN29 using a pre-trained model with Semi Hard Triplet Loss(%) 72.0.

- SupCon with SNR Dist (LightCNN29)



Rank1 Identification accuracy for LightCNN29 using a pre-trained model with Sup Contrastive Loss with SNR distance(%) 86.6.

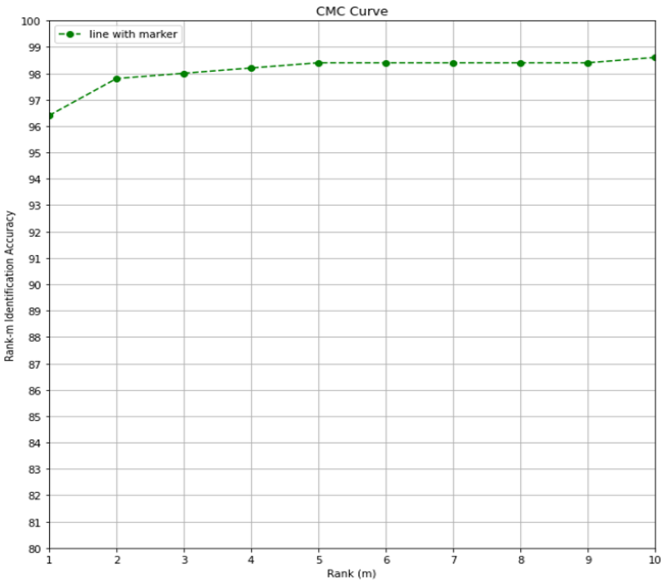
- Contrastive + Variance (Compact Convolution Transformer)



Rank1 Identification accuracy for CCT using Pre-trained Init model with Contrastive & Variance Losses(%) 100.0.

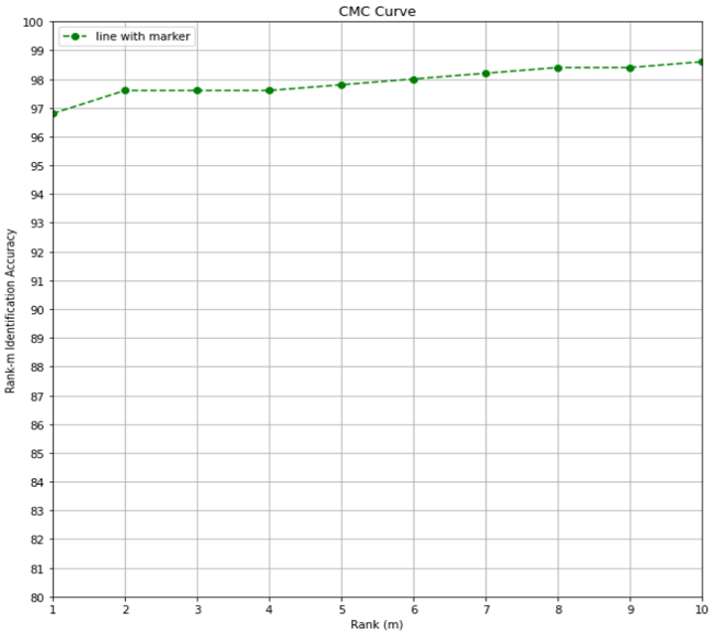
b. Using fine-tuned models with different loss combinations:

- Contrastive + Variance (LightCNN29)



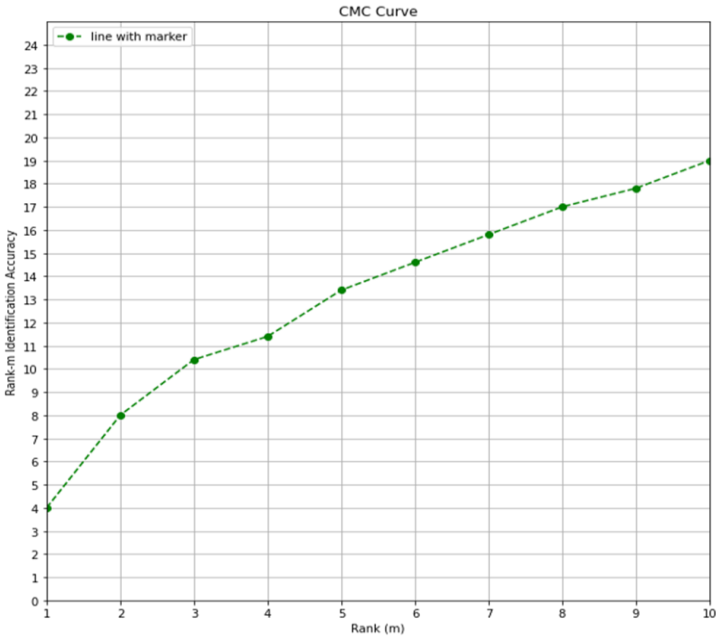
Rank1 Identification accuracy for LightCNN29 using a pre-trained model after fine tuning with Contrastive & Variance Losses(%) 96.39999999999999.

Rank10 Identification accuracy for LightCNN29 using a pre-trained model after fine tuning with Contrastive & Variance Losses(%) 98.6.



Rank1 Identification accuracy for LightCNN29 using a pre-trained model before training with Contrastive & Variance Losses(%) 96.8.
Rank10 Identification accuracy for LightCNN29using a pre-trained model before training with Contrastive & Variance Losses(%) 98.6.

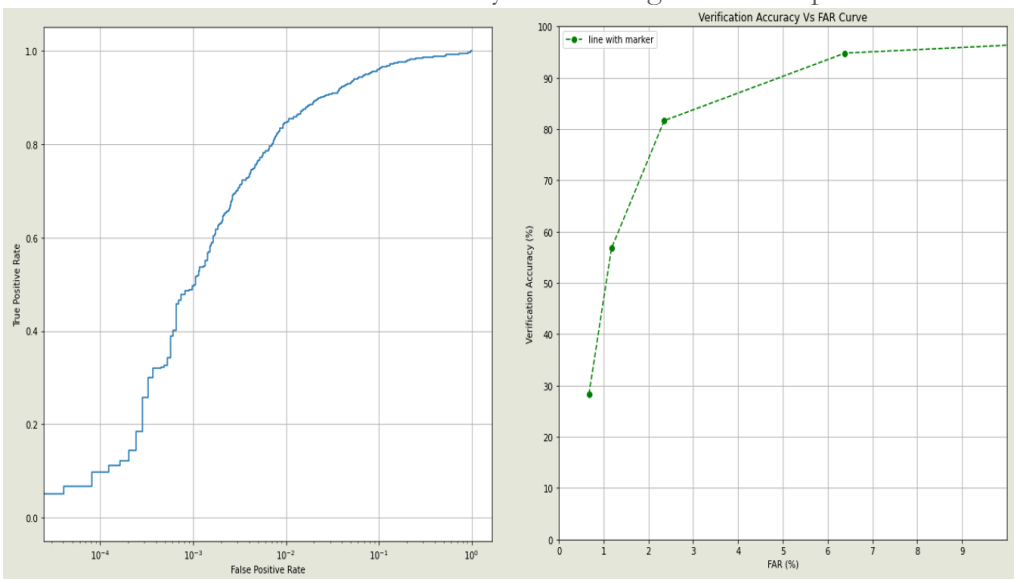
- Contrastive + Variance (Compact Convolution Transformer)



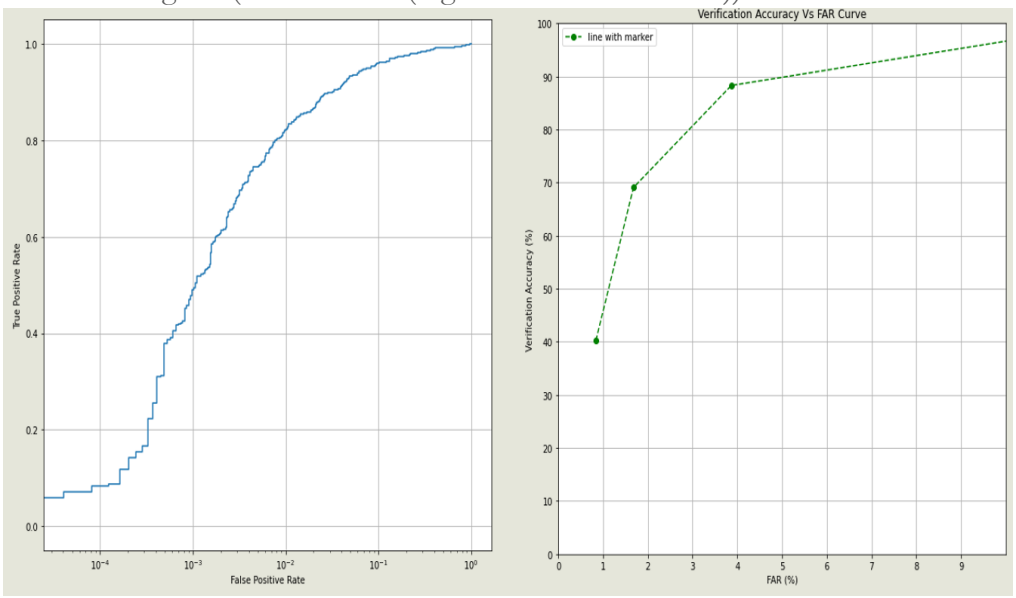
Rank1 Identification accuracy for CCT using a pre-trained model after fine-tuning with Contrastive & Variance Losses(%) 4.0.
Rank10 Identification accuracy for CCT using a pre-trained model after fine-tuning with Contrastive & Variance Losses(%) 20.0.

2. **On DFW dataset:**
    a. **Using Fine-tuned models with different loss combinations:**
        - Contrastive (LightCNN29)

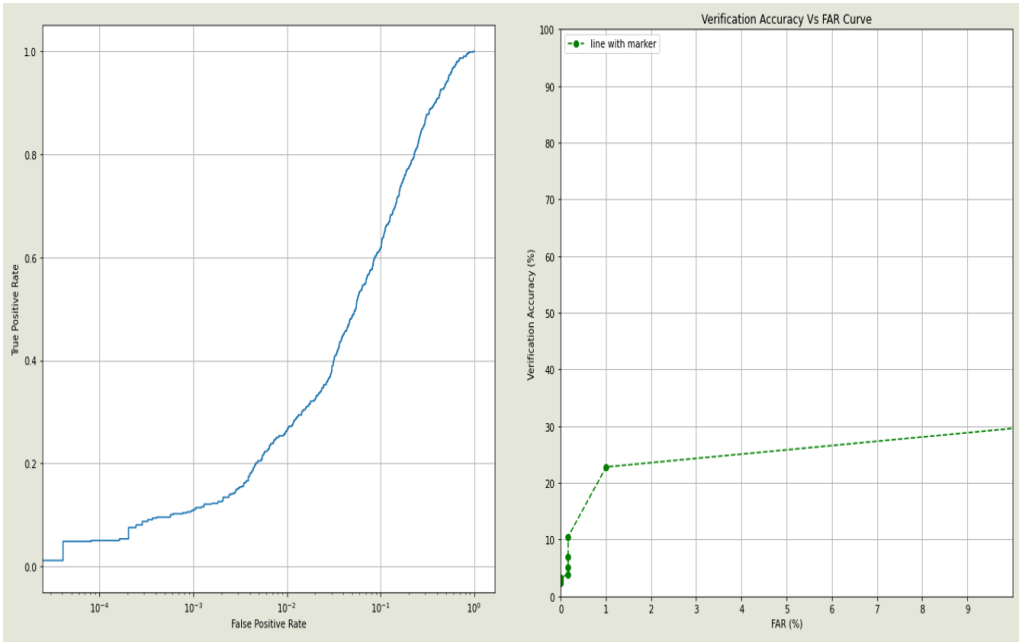
Here we have shown for 1% FAR by considering a few samples.



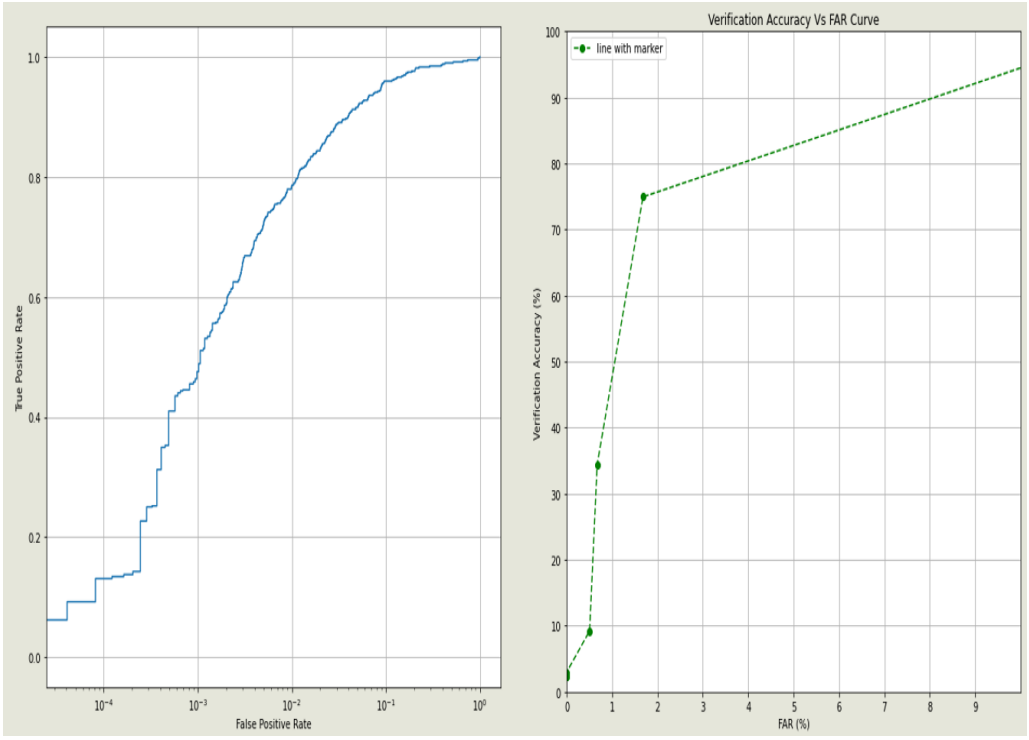
        - Center + Angular (LAttCNN29 (Light Attention CNN 29))



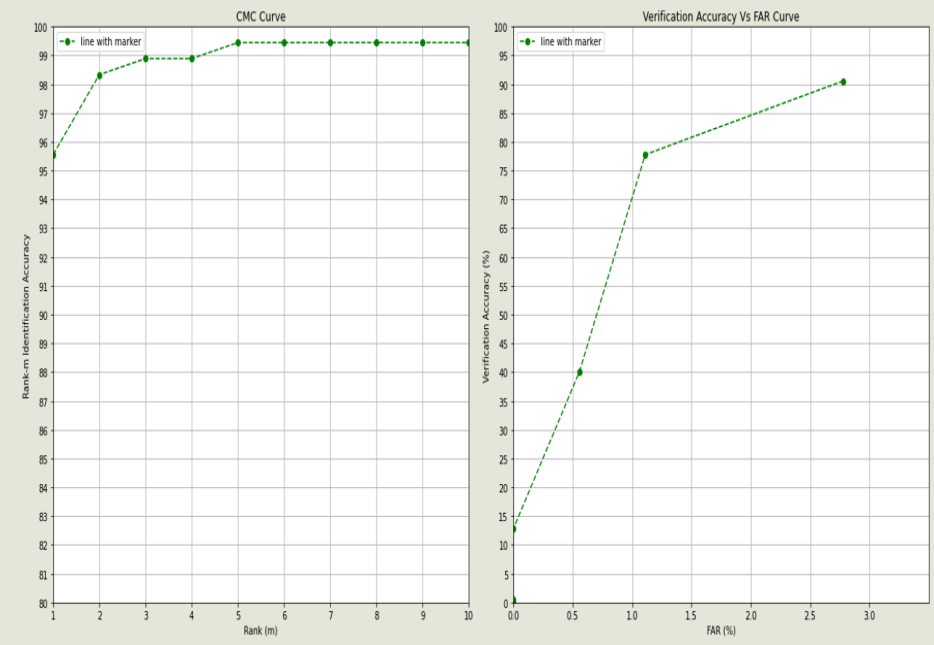
- Cosine Augmented Center loss (LAttCNN29 (Light Attention CNN 29))



- ArcFace + Variance Loss (LAttCNN29 (Light Attention CNN 29))



3. **On DFW dataset:**
    a. **Using Fine-tuned models with different loss combinations:**
        - Contrastive (LightCNN29)



Rank1 Identification accuracy for LightCNN29 using a pre-trained model after fine tuning with Contrastive Loss for Plastic Dataset(%) 95.55555555555556.
Rank5 Identification accuracy for LightCNN29 using a pre-trained model after fine tuning with Contrastive Loss for Plastic Dataset(%) 99.44444444444444.
Rank10 Identification accuracy for LightCNN29 using a pre-trained model after fine tuning with ContrastiveLoss for Plastic Dataset(%) 99.44444444444444.

Conclusion:

The datasets LFW, Plastic Surgery Faces and DFW (Protocol1) are small sample datasets. So training from scratch or using pre-trained initializations of LightCNN29 didn't give good results. The loss function we proposed, Cosine Augmented Center Loss, didn't give satisfactory results. The best results were found by fine-tuning the pre-trained LightCNN29 model updated with Attention Modulated MFM, using Contrastive Loss and variants.

Further exploring architectures such as DenseNet or ViT for fine-tuning along with loss function updates can give better results (rank1 accuracy & verification accuracy).