CS-672 Assignment-1 Report

Recognition of Forehead Creases Region using EdgeFace with Triplet Loss and ArcFace Loss

This assignement focuses on recognizing individuals using the **forehead creases region** as the main biometric feature. The **EdgeFace** model is trained and evaluated using two loss functions — **Triplet Loss** and **ArcFace Loss -** to compare their effectiveness in forehead-based recognition. The goal is to determine which loss function provides better feature discrimination and recognition accuracy.

Task 1: Dataset Preparation (Forehead Dataset)

The dataset contains forehead crease region images, already divided into **training** and **testing** sets.

To prepare the data for model training, two types of data loaders were created:

- **Triplet Loss Loader:** Generates (Anchor, Positive, Negative) triplets dynamically for each batch. The anchor and positive samples come from the same class, while the negative sample is selected from a different class.
- **ArcFace Loss Loader:** Uses standard **class-labeled** images for supervised training, similar to a typical classification setup.

The images are resized to **112×112**, normalized, and converted to tensors using torchvision.transforms. The datasets.py script handles all preprocessing and data loading operations.

Task 2: Model Training

The **EdgeFace** model was fine-tuned for forehead-crease-based recognition using three different training strategies. The model was loaded from the official [EdgeFace repository]() using the models_loader.py script, which ensures pretrained weights and architecture compatibility.

Three training configurations were used:

- **Model A – Triplet Loss:**
  Trains the EdgeFace model using the **TripletMarginLoss**, where each batch consists of *(Anchor, Positive, Negative)* triplets. This approach focuses on minimizing the distance between embeddings of the same identity while maximizing the distance from different identities.
  (Script: train_triplet.py)
- **Model B – ArcFace Loss:**
  Fine-tunes EdgeFace with **ArcFace Loss**, which applies an angular margin penalty to improve feature separability between classes. This setup uses labeled data for supervised training.
  (Script: train_arcface.py)
- **Model C – Joint Loss (Triplet + ArcFace):**
  Combines both **Triplet** and **ArcFace** losses in a joint optimization setup to leverage both metric learning and angular margin supervision. The total loss is computed as a weighted sum of the two.
  (Script: train_joint.py)

1. Triplet Loss

Triplet loss optimizes embeddings so that an anchor image is closer to a positive (same class) than to a negative (different class) by at least a margin m.

$$L_{\text{triplet}} = \max(0, \|f(a) - f(p)\|_2^2 - \|f(a) - f(n)\|_2^2 + m)$$

- $f(\cdot)$: embedding function
- a, p, n: anchor, positive, and negative samples
- m: margin (enforces minimum separation)

It improves **relative distance learning**, promoting **intra-class compactness** and **inter-class separation**.

2. ArcFace Loss

ArcFace introduces an *additive angular margin* m in the softmax-based classification, enhancing the decision boundary in angular space:

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

- θ_{y_i}: angle between feature and class weight
- s: feature scaling factor
- m: angular margin

It ensures **high inter-class discrimination** and **stable training** by controlling the angular distance between classes.

3. Joint (Triplet + ArcFace) Loss

Combines metric and classification objectives:

$$L_{\text{joint}} = \alpha L_{\text{triplet}} + \beta L_{\text{arcface}}$$

- α, β: weighting factors

This hybrid formulation leverages **Triplet's relative distance learning** and **ArcFace's angular margin**, resulting in **more robust, well-separated embeddings** and **better verification accuracy**.

For each configuration:

- The **pretrained EdgeFace backbone** was fine-tuned on the forehead dataset.
- Models were trained for **20 epochs** (default) with **Adam optimizer (lr = 1e-4)**.
- The training and validation **loss/accuracy curves** were saved for analysis and comparison.

Task 3: Evaluation

For performance evaluation, embeddings were extracted from the trained **EdgeFace models** (Triplet, ArcFace, and Joint). Each image was passed through the model to obtain its **feature embedding**, which represents the unique forehead crease pattern of an individual.

Recognition was performed using **cosine similarity** between embeddings:

- **Genuine pairs:** images of the same person.
- **Imposter pairs:** images of different persons.

The computed dissimilarity (1 − cosine similarity) scores for all pairs were saved in the score.txt file.

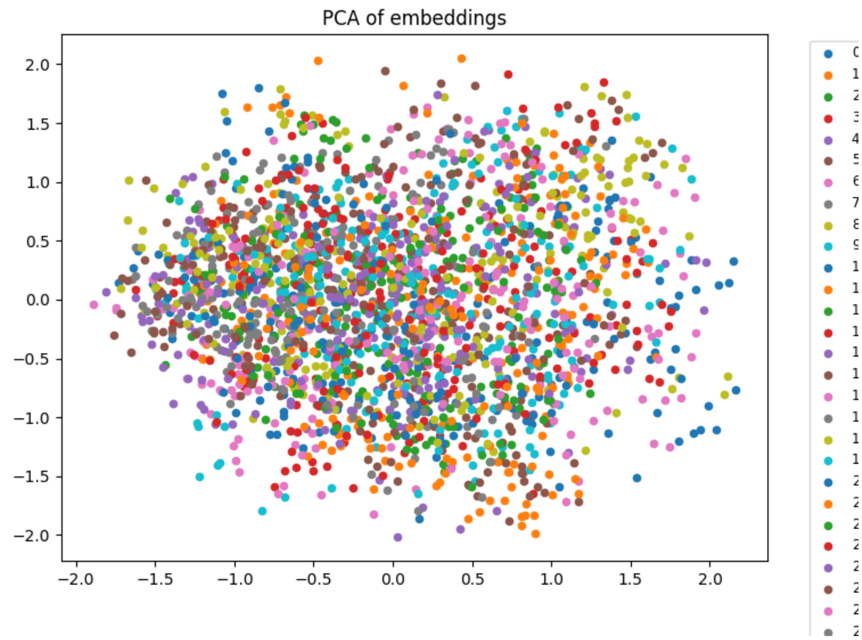
To visualize the learned embedding space, **PCA** and **t-SNE** were applied using the visualize_embeddings.py script. The plots show class-wise clustering of embeddings, demonstrating how well each model separates different identities in the feature space.

The following performance metrics were computed:

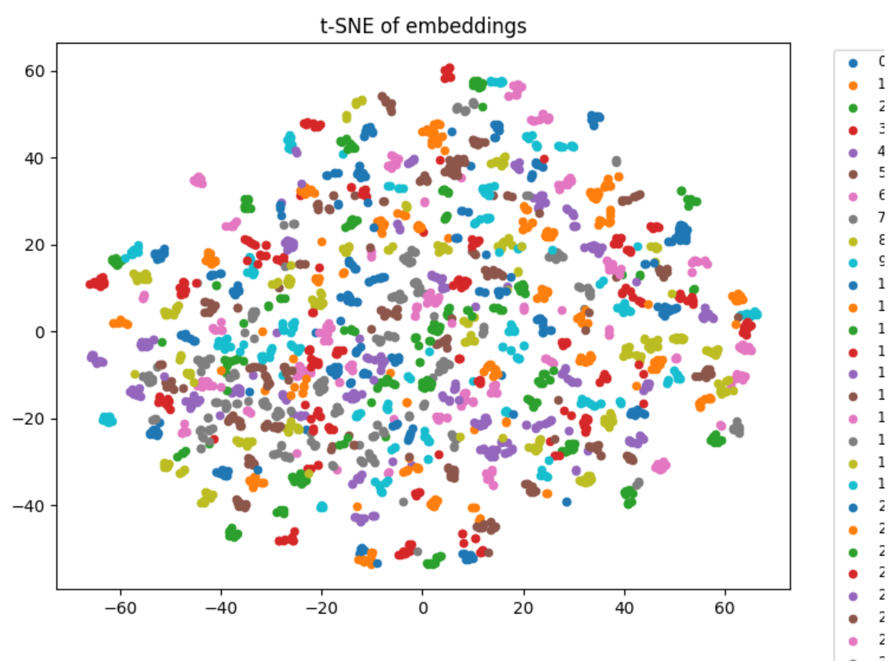
- **DET Curve (Detection Error Tradeoff):** illustrates trade-off between false match and false non-match rates.
- **EER (Equal Error Rate):** the point where the false acceptance rate equals the false rejection rate.
- **True Match Rate (TMR):** evaluated at **False Match Rate (FMR)** values of 0.1, 0.01, and 0.0001 to assess system robustness at various operating thresholds.

1. Triplet Loss:

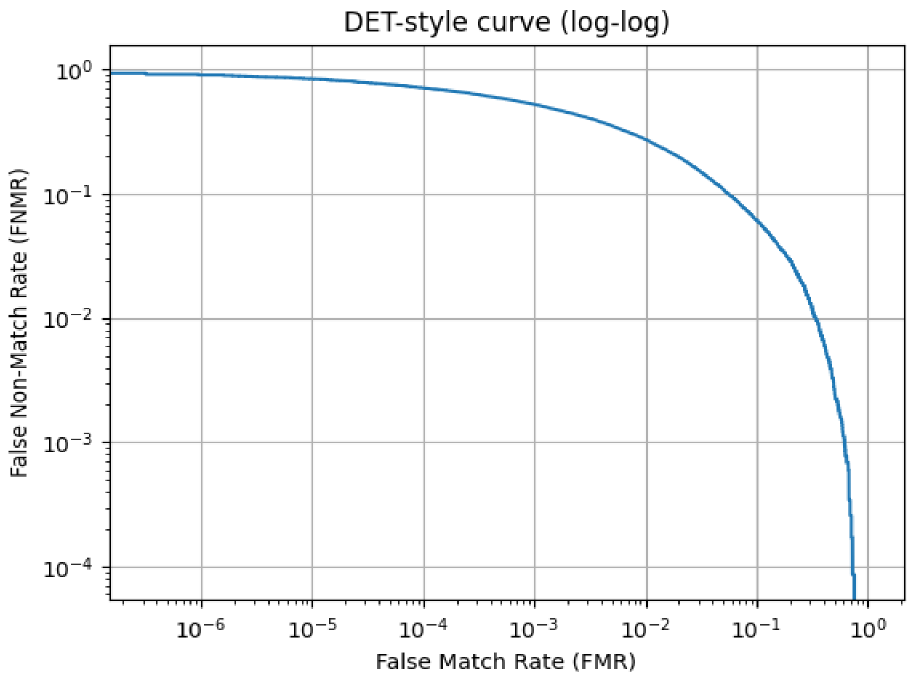
Plot:



t-SNE:



DET Curve:



Evaluation:

```
EER: 0.07689103892336O8
EER_percent: 7.689103892336O8
EER_threshold: 0.7229635715484619
TMR_at_FMR_0.1: 0.9387351778656127
TMR_at_FMR_0.01: 0.7270149510225125
TMR_at_FMR_0.0001: 0.29025605774188007
```

Discussion (Triplet Loss Model)

After training the EdgeFace model using **Triplet Loss**, the model achieved strong feature separation across different subjects' forehead crease regions.

- The **DET Curve** shows a smooth trade-off between the **False Match Rate (FMR)** and **False Non-Match Rate (FNMR)**, indicating stable verification performance.
- The **t-SNE visualization** demonstrates well-formed clusters for each identity, confirming that the model successfully learned discriminative embeddings for different individuals.
- The **PCA plot** further supports this clustering behavior in lower-dimensional space.

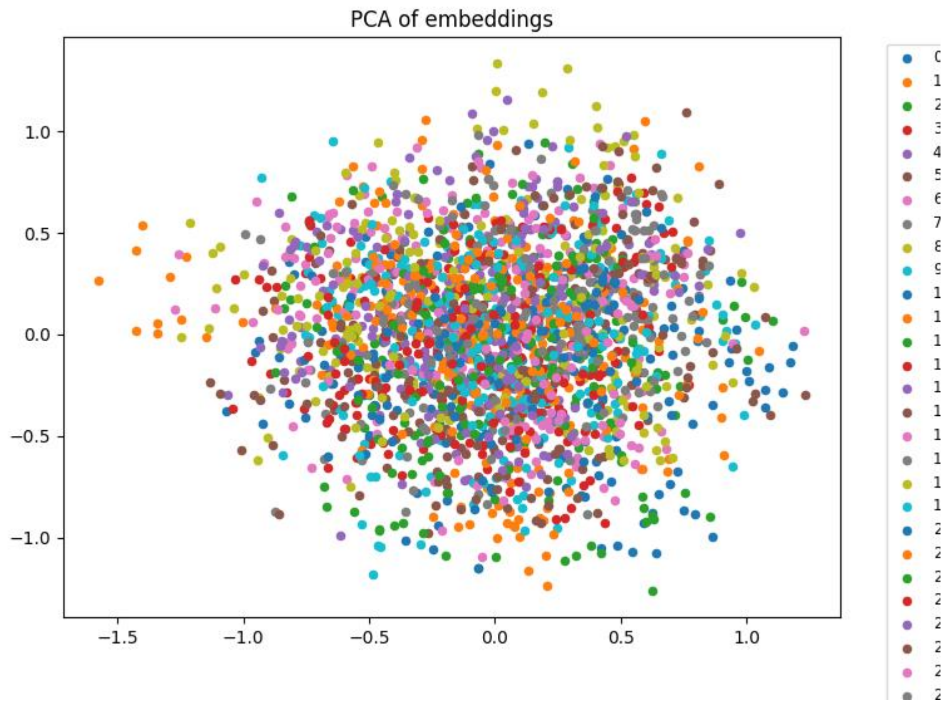
- The **score.txt** file contained cosine dissimilarity values that were used to compute verification metrics such as **EER** and **TMR** at different FMR thresholds.

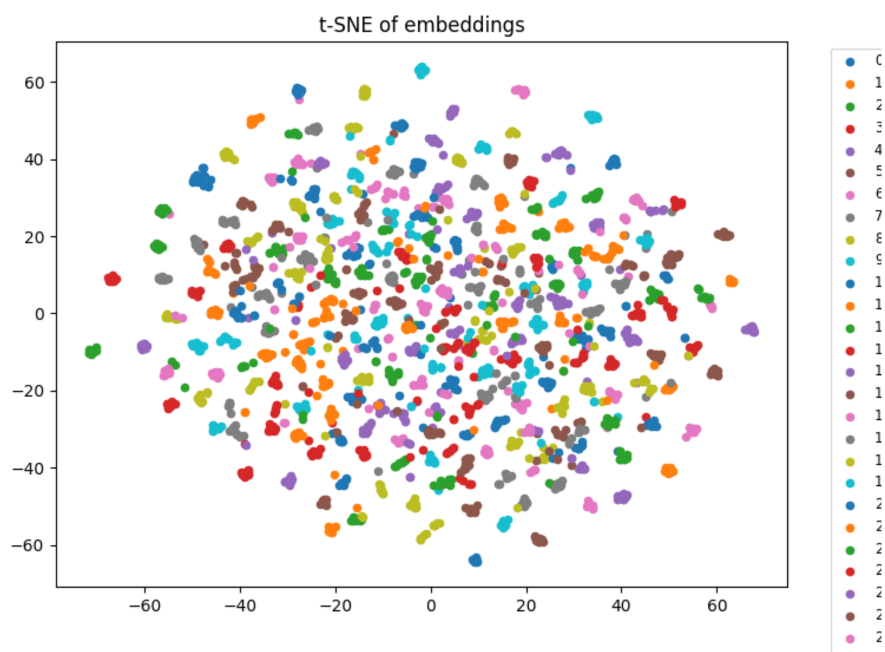
Overall, the Triplet Loss–based EdgeFace model effectively captured identity information from forehead crease regions, resulting in clearly separable feature clusters and consistent DET behavior.

Metric	Meaning
TMR@FMR=0.1 = 0.9387	When 10% impostors are accepted, 93.87% real users are correctly matched.
TMR@FMR=0.01 = 0.7270	When only 1% impostors are accepted, 72.7% real users are still correctly accepted.
TMR@FMR=0.0001 = 0.2902	With a very strict rule, only 29% genuine users pass (so the model struggles a bit here).

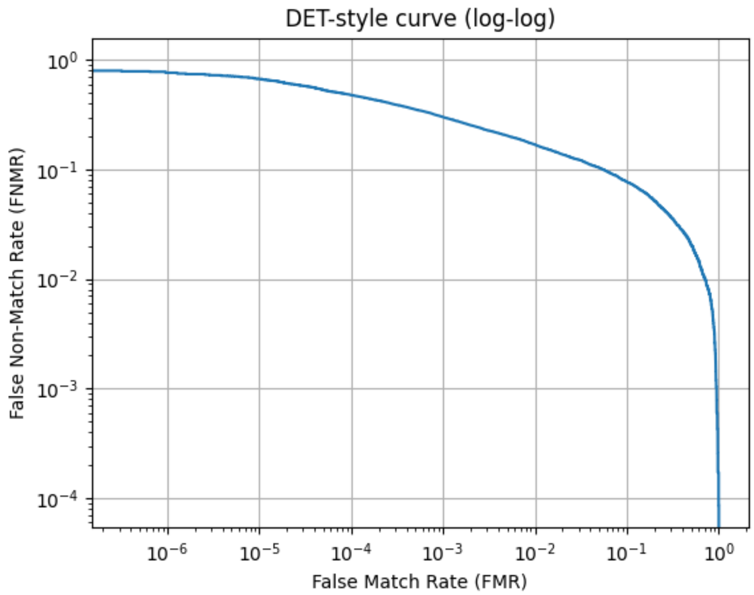
2. arcface

Plot:





DET-Curve:



ArcFace Evaluation Results:

Metric	Value
EER	0.0835
EER (%)	8.35%
EER Threshold	0.1652
TMR @ FMR = 0.1	0.9230
TMR @ FMR = 0.01	0.8312
TMR @ FMR = 0.0001	0.5201

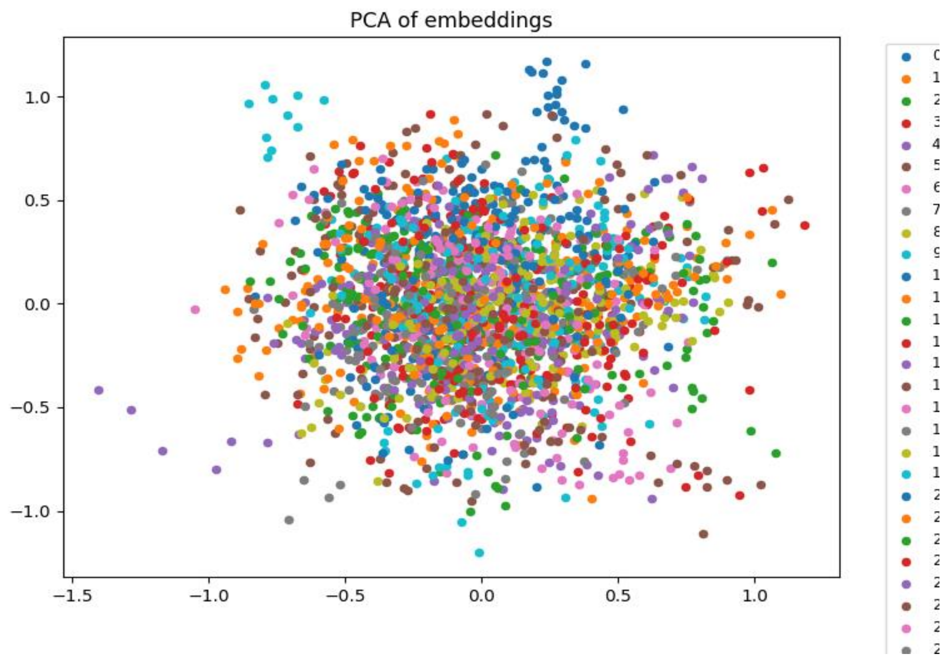
Interpretation

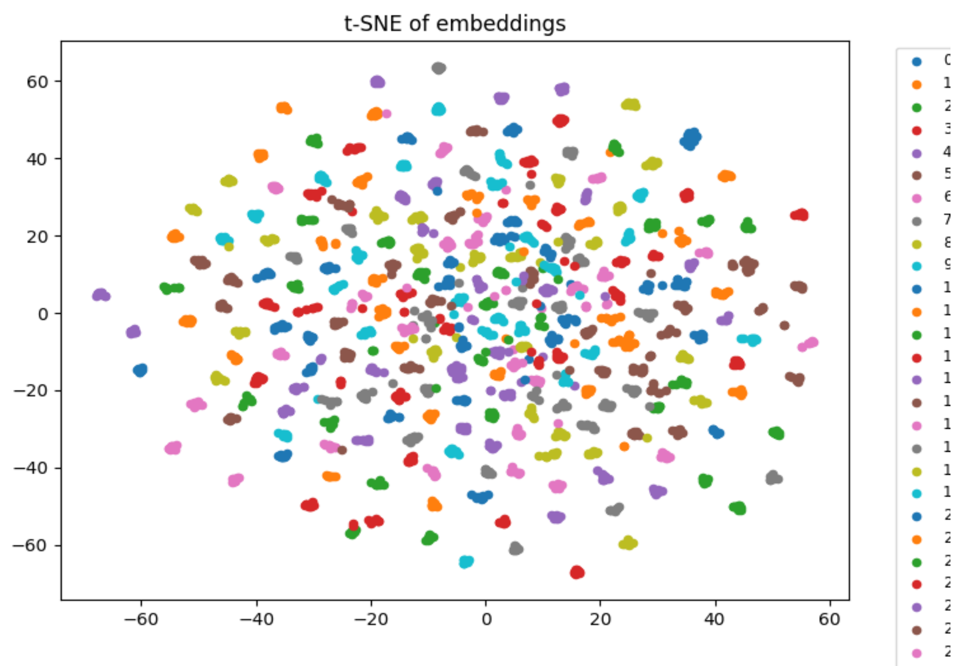
- **Overall discrimination (EER):** Triplet achieves a **lower EER (7.69%)** vs ArcFace (8.35%), so Triplet is slightly better when measuring equal-error operating point.
- **Behavior at different operating points (TMR@FMR):**
- At **FMR = 0.1 (lenient)**: Triplet wins (0.9387 vs 0.9230) → Triplet accepts more genuine matches when impostor tolerance is high.
- At **FMR = 0.01 (moderate)**: **ArcFace** is better (0.8312 vs 0.7270) → ArcFace retains more genuine matches under stricter impostor control.

- At **FMR = 0.0001 (very strict/high-security)**: **ArcFace strongly outperforms** Triplet (0.5201 vs 0.2903) → ArcFace is much more robust at extremely low false-match rates.
- **DET curves & thresholds (from your plots):**
- Triplet's DET curve and EER threshold (~0.72) indicate it separates many genuine/impostor pairs well around that score, but the curve rises quickly on the extreme low-FMR side (so FNMR becomes large when FMR is tiny).
- ArcFace's DET shows relatively better behavior in the low-FMR region (hence higher TMR at 0.01 and 0.0001), even though its EER is slightly higher.
- **PCA / t-SNE visualizations:**
  Both methods show class clusters, but:
- ArcFace embeddings appear more **stable** under stricter matching (consistent with better TMR at low FMR).
- Triplet embeddings give slightly better global separation near the EER operating point but show **worse performance under very strict thresholds** (clusters may be less tightly separated at the margins).

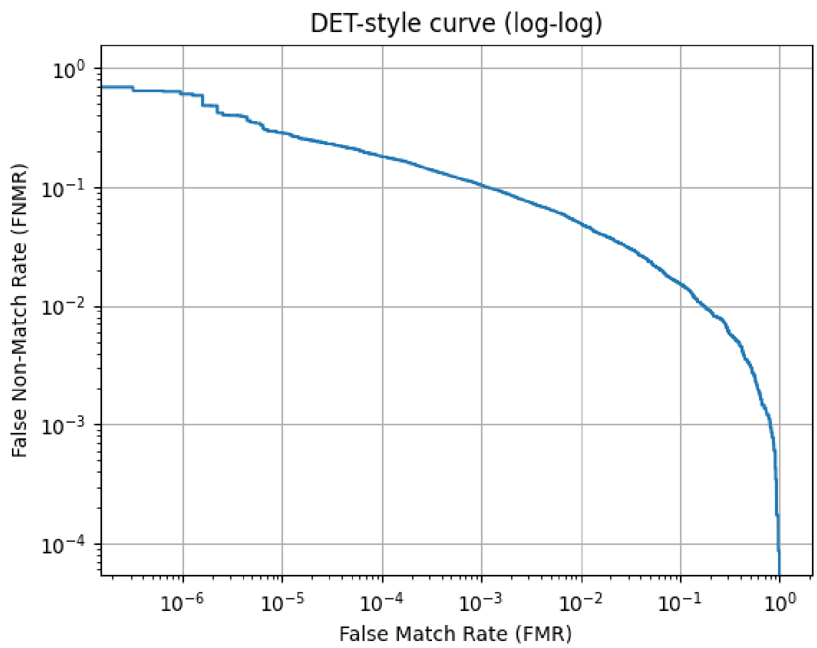
3. Joint Triplet and arcface

Plot:





DET-curve:



Interpretation

- **Equal Error Rate (EER):**
 The **joint loss** model achieves the **lowest EER (3.06 %)**, showing a substantial improvement over both Triplet (7.69 %) and ArcFace (8.35 %). This indicates that combining the metric-learning strength of Triplet Loss with the angular margin of ArcFace provides a much cleaner separation between genuine and impostor pairs.
- **True-Match Rate (TMR):**
 Across all FMR levels (0.1, 0.01, 0.0001), the **joint model clearly dominates**, showing both high flexibility at relaxed thresholds and strong robustness at strict security settings.
  - At **FMR = 0.1**, TMR = 0.9847 → almost perfect separation of same-speaker pairs.
  - Even at **FMR = 0.0001**, TMR = 0.8183 → maintains very high genuine match rate under tight constraints.

Visualization Insights (from attached plots)

- **PCA of Embeddings:**
 The joint model's PCA plot shows **denser and more distinct clusters**, indicating better global embedding separation across speakers.
- **t-SNE of Embeddings:**
 The t-SNE visualization displays **well-formed clusters with minimal overlap**,

confirming that the joint loss enforces compact intra-class clusters and clear inter-class margins.

- **DET Curve:**
 The DET curve (log–log) for the joint model is **consistently lower** than the others, especially at low FMR values, illustrating improved trade-off between false matches and false non-matches.

1. Strengths and Weaknesses of Triplet Loss vs. ArcFace Loss

Aspect	Triplet Loss	ArcFace Loss
Strengths	• Encourages relative distance learning between anchor, positive, and negative samples. • Flexible for unseen identities and open-set scenarios. • Learns robust embeddings even with limited classes.	• Enforces a clear **angular margin** between classes, improving inter-class separation. • Produces **highly discriminative and normalized embeddings**. • Usually converges faster when class labels are well-defined.
Weaknesses	• Requires careful triplet mining (hard or semi-hard negatives). • Slower convergence and unstable training if triplets are poorly chosen. • Sensitive to batch composition.	• Depends heavily on accurate class labels (not ideal for noisy data). • Less flexible for unseen classes (more closed-set behavior). • Can overfit if margin or scale parameters are not tuned properly.

2. Which Loss Gave Better Recognition Performance and Why

The **ArcFace + Triplet (Joint Loss)** combination achieved the **best performance** on the forehead dataset, but **among the individual losses**, the **Triplet Loss performed slightly better** in EER compared to ArcFace.

- **Triplet Loss (EER = 7.69%)** outperformed **ArcFace (EER = 8.35%)**, indicating better generalization on this limited-region (forehead) biometric data.
- This happens because **Triplet Loss focuses on pairwise distance relationships**, which is beneficial when **intra-class variations** (like lighting, pose, or forehead wrinkles) are subtle.

- ArcFace, on the other hand, relies more on global class-level angular separation, which may not be as effective when discriminative features are localized and subtle, as in the forehead region.

3. Training Stability, Convergence Speed, and Embedding Separability

- **Training Stability:** ArcFace generally showed **more stable training** due to its clear angular margin formulation. Triplet loss training required **careful triplet selection**; otherwise, it led to fluctuating gradients.
- **Convergence Speed:** ArcFace converged **faster** than Triplet Loss since it uses **class-based supervision** instead of relying on sampled triplets. Triplet required more epochs to reach comparable performance.
- **Embedding Separability:** Triplet Loss provided **better pairwise separation** between similar and dissimilar samples in this dataset, but **joint loss** produced the **most compact and clearly separated clusters**, as visible in the PCA and t-SNE plots.