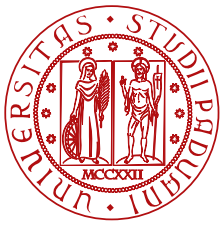


UNIVERSITÀ DEGLI STUDI DI PADOVA

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

CORSO DI LAUREA IN COMPUTER ENGINEERING

Development of a Deep Learning Model with a Novel Loss Function for Facial Recognition based on ArcFace

Relatore
Prof. Nanni Loris

Laureando
Mario Giovanni Peloso

ANNO ACCADEMICO 2023-2024

Data di laurea 12/12/2024

Vorrei ringraziare tutte le persone che mi sono state vicine in questo lungo percorso universitario. Un ringraziamento particolare va ai miei genitori, che mi hanno sempre incoraggiato ad andare avanti senza mai farmi sentire sotto pressione, permettendomi di affrontare tutto con serenità e tranquillità. Un grazie speciale anche alle mie sorelle e a mio fratello, per il loro sostegno costante e prezioso. Un pensiero va anche ai miei zii, cugini e nipoti, che con il loro affetto e supporto mi hanno fatto sentire parte di una famiglia unita. Infine, grazie a tutti i miei amici, che mi hanno accompagnato sia nello studio che nei momenti di svago… forse anche troppo, a volte!

I would like to thank everyone who has supported me throughout this long university journey. A special thanks goes to my parents, who always encouraged me to keep going without ever making me feel under pressure, allowing me to face everything with calmness and peace of mind. I am also especially grateful to my sisters and brother for their constant and invaluable support. My appreciation also extends to my uncles, cousins, and nieces and nephews, whose love and encouragement made me feel part of a close-knit family. Lastly, thank you to all my friends, who have been by my side through both study sessions and times of relaxation... maybe even a bit too much sometimes!

Abstract

This thesis focuses on the development of a novel loss function for deep learning-based facial recognition models. Building upon the foundation of ArcFace, the proposed loss function integrates key concepts from recent advancements, including MagFace, CurricularFace, and AdaFace, to enhance model performance. The ResNet architecture, specifically leveraging pre-trained models, is fine-tuned to accommodate this new loss function. The effectiveness of the resulting models is rigorously evaluated on the IJB-C dataset, with performance metrics highlighting the improvements in recognition accuracy and robustness. This research contributes to the field of facial recognition by offering a more refined loss function that balances identity separation and intra-class compactness, thereby improving model generalization.

Introduction

Facial recognition represents one of the most significant innovations in the realm of biometric recognition, emerging as a fundamental technology in applications that span from the commercial sector to government use. Over the past few decades, the scientific community has focused considerable efforts on improving the reliability and accuracy of these systems, making them essential tools for identifying and authenticating individuals. Unlike other forms of biometrics, such as fingerprint analysis or iris recognition, facial recognition has the advantage of being able to operate on non-cooperative subjects without requiring physical contact with the device. This characteristic makes it particularly suitable for monitoring in high-density environments, such as smart cities, where security and access control are of paramount importance [1]. However, the robustness of facial recognition systems must confront significant challenges posed by adverse environmental conditions, such as uneven lighting, variations in posture, and obstacles like glasses or hats. To overcome these challenges, deep convolutional neural networks (DCNNs) have become central, supported by the adoption of loss functions with angular margins. These functions optimize the representation of facial features, improving the separation between distinct classes and increasing the discriminability of facial embeddings, an aspect that translates into a significant improvement in the overall accuracy of facial recognition systems [2].

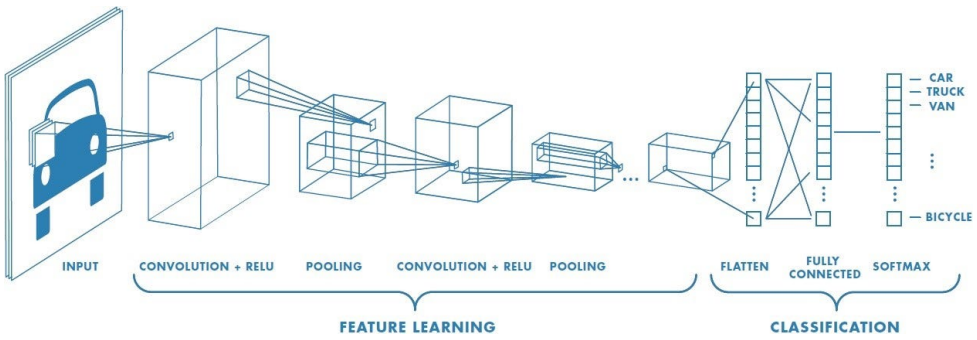


Figure 2: Example of an CNN architecture. The main difference between a CNN and a DCNN is the number of layers for feature learning.

Face Identification and Verification: A Comparison

Within the field of face recognition, two main tasks are distinguished: face identification and face verification. While face identification aims to recognize an individual's identity by comparing a facial image to a predefined set of identities, face verification has a more limited goal, answering a binary question: do two images belong to the same person? In other words, face verification checks whether two facial images represent the same individual, without explicitly identifying who that person is. Despite these conceptual differences, the models used for face verification and face recognition are nearly identical. Both tasks rely on models capable of generating compact and discriminative representations of facial features, often referred to as embeddings. Algorithms like FaceNet [3] introduced the use of embeddings to represent faces in vector space, reducing the distance between representations of the same face and maximizing it between different faces. This innovation has greatly contributed to both face verification and face recognition, setting a new standard in developing robust and precise models for both applications.

Figure 3: Example of face recognition usage in a smart city.

Angular Margin Innovations

The adoption of angular margins has revolutionized the approach to facial recognition, allowing for the overcoming of the limitations of traditional Softmax-based loss functions. For example, ArcFace [4] introduces a constant angular margin between the input and the class vector, deliberately increasing the distance between classes in the feature space. This strategy helps re-

duce the risk of error, especially in uncontrolled contexts where the variability of environmental conditions is high and the robustness of the system is crucial. These techniques demonstrate particular effectiveness when applied to large-scale datasets, where the variety and complexity of the data can be significant. However, creating and annotating such datasets poses a considerable challenge in terms of resources and time, requiring substantial human intervention to ensure label accuracy. To address these issues, variants such as MagFace [5], which adjusts the margin based on image quality, and AdaFace [6], which adapts the margin based on the difficulty of the sample, have been developed. These approaches allow models to dynamically calibrate the angular margin, further enhancing the robustness and precision of facial recognition.

Goals of the Study

My research work aims to develop a new loss function inspired by those that leverage angular margins, such as ArcFace [4], AdaFace [6], CurricularFace [7], and MagFace [5]. The objective is to create a hybrid between MagFace and CurricularFace, integrating the distinctive characteristics of both into a single loss function. This approach seeks to further optimize the performance of facial recognition models, particularly in the domain of deep learning, utilizing the MATLAB environment through the Deep Learning Toolbox and its supporting packages. The models used in this research were taken from the AdaFace repository and subsequently fine-tuned for our specific purposes. Furthermore, the trained models were subjected to the IJB-C [8] test to evaluate the effectiveness and robustness of the new techniques developed.

Contents

List of Figures

1

General Overview of Margin Based Loss Function

Margin-based loss functions have emerged as a pivotal innovation in the realm of deep learning, particularly within facial recognition systems. These functions operate by creating a defined separation, or margin, between classes in the feature space, thereby improving the discriminative power of models. By emphasizing the angular relationships between samples, margin-based approaches enable more robust classification in challenging conditions. This methodology not only enhances the accuracy of facial recognition systems but also paves the way for subsequent advancements in loss function design, ensuring that models can effectively learn from complex and varied datasets.

1.1 Evolution of Loss Functions in Deep Face Recognition

Figure 1.1: Timeline of Loss Function development.

The development of effective loss functions has been a cornerstone for learning discriminative representations in face recognition models. Early deep learning approaches to face recognition relied primarily on generic loss functions, such as Softmax Loss, which maximizes the probability of correct classes without imposing explicit geometric constraints among classes in the embedding space. For instance, Taigman et al.'s work on DeepFace applied Softmax Loss to achieve competitive results in face verification, but without guaranteeing angular separation among classes in the representation space [9]). Although functional for general classification tasks, this approach showed limitations in generalizing to face recognition, where separability among classes in the latent space is fundamental.

Embedding space in this context refers to a high-dimensional vector space where the features or embeddings of facial images are mapped. Each point in this space represents a compressed, numerical version of the face's features, extracted by the neural network. In the embedding space, the distance between any two points reflects the degree of similarity or dissimilarity between the corresponding facial features. Effective loss functions aim to structure this space such that embeddings of the same class (i.e., the same individual's face) cluster together while those of different classes are spaced apart, enhancing the model's ability to discriminate between identities accurately.

As we continue to explore the evolution of loss functions in deep face recognition, understanding the role of embedding space becomes crucial. This understanding helps in appreciating how innovations in loss function design, such as the introduction of angular margins, directly influence the arrangement of classes within this space, thereby improving the model's discriminative power and generalization ability.

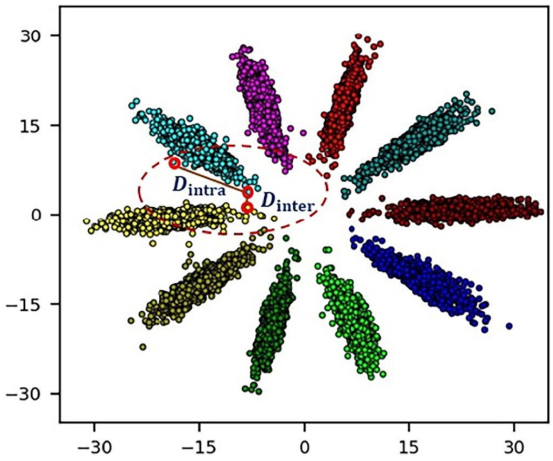


Figure 1.2: The problem of the Softmax Loss is the distance in the embedding space between samples of different class (D_{inter}) that is lower respect to the distance of samples of the same class (D_{intra})

To overcome these limitations, Schroff et al. introduced Triplet Loss in their FaceNet system, one of the first loss functions specifically designed to improve intra-class compactness and inter-class separability [3]. Triplet Loss optimizes triplets of samples (an anchor, a positive, and a negative) by minimizing the distance between the anchor and the positive while maximizing it between the anchor and the negative. This approach yielded more discriminative embeddings but at a high computational cost, as selecting optimal triplets and managing their quantity is computationally demanding, especially on large-scale datasets.

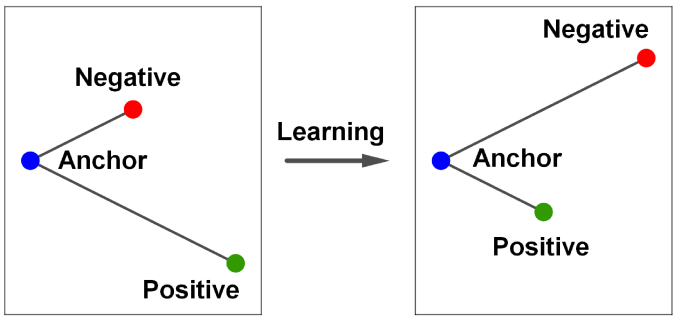


Figure 1.3: During training the model aims at minimizing the distance between the anchor and the positive while maximizing the distance from the anchor to the negative.

In 2016, Wen et al. proposed the Center Loss, which further improved class discrimination by pulling samples closer to a predefined "center" for each class in Euclidean space [10]. Center Loss reduces intra-class variance without explicitly altering inter-class distances. However, it remains limited in ensuring optimal inter-class separation, particularly in high-dimensional embedding spaces used in face recognition.

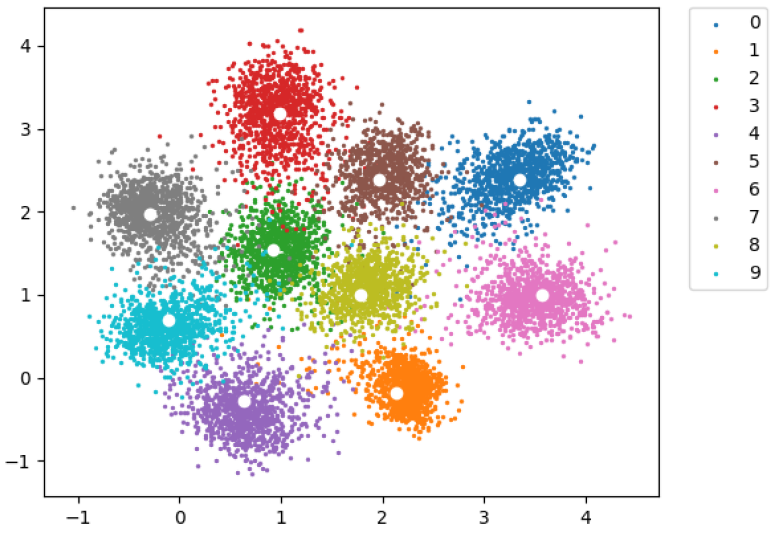


Figure 1.4: Example of how samples are visualized in the Euclidean space around 9 centers.

After the introduction of Center Loss, which reduced intra-class variance by pulling samples closer to class centers in Euclidean space [10], researchers sought methods to enhance inter-class separability further. In 2016, Liu et al. introduced the Large-Margin Softmax (L-Softmax) loss function, which incorporated angular margins directly into the softmax loss [11]. The L-Softmax loss enforced explicit angular margins between classes, guaranteeing more robust class positioning in the latent space. This method marked a significant step forward by explicitly optimizing the decision boundaries to enhance inter-class separability while maintaining intra-class compactness.

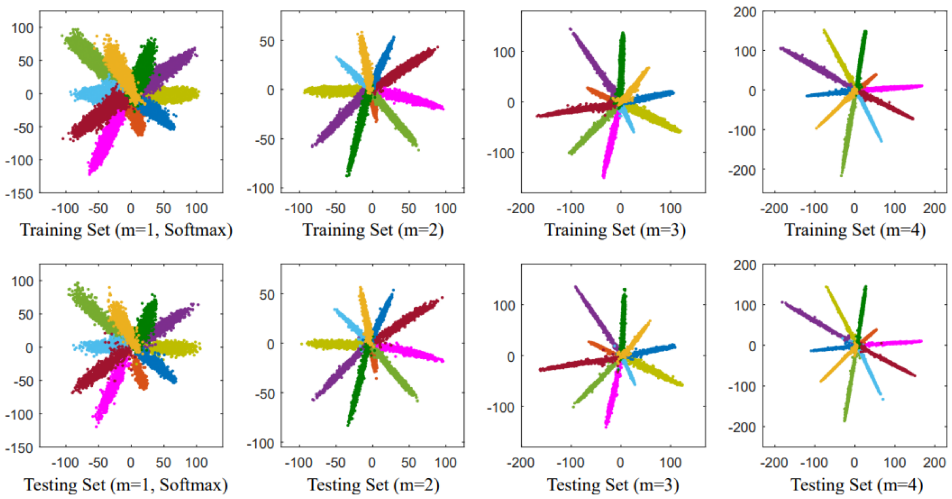


Figure 1.5: Features visualizations (Softmax Loss (m=1) vs. L-Softmax loss (m=2,3,4)) in MNIST dataset

Building upon the foundation laid by L-Softmax, the next advancement came with SphereFace in 2017, also proposed by Liu et al. [12]. SphereFace further refined the concept of angular margins by constraining embedding vectors to lie on a hypersphere and introducing multiplicative angular margins between classes. This approach significantly enhanced angular separation and improved model generalization by optimizing class positions on a hypersphere, leading to superior performance in face recognition tasks.

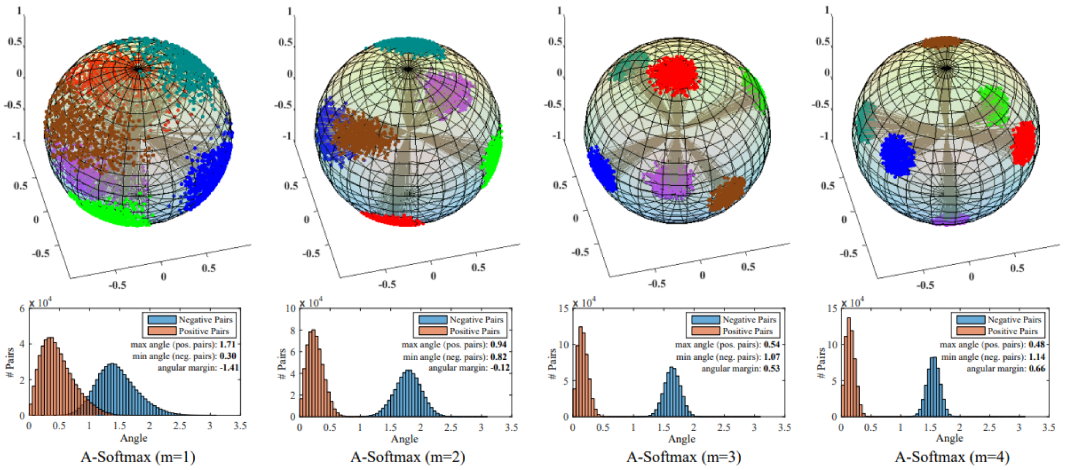


Figure 1.6: Visualization of features learned with different m by using a 6- class subset of the CASIA-WebFace dataset. With larger m the classification margin becomes larger

This progression, culminating in angular-margin-based loss functions, addressed the limitations of previous methods, ushering in a new era of performance and precision in face recognition systems.

1.2 ArcFace Loss

In the field of facial recognition, the introduction of loss functions that leverage angular margins has significantly enhanced models' ability to discriminate between different classes. Among the various functions developed, ArcFace is recognized as the foundation of modern facial recognition systems, serving as a starting point for other advanced loss functions like MagFace, AdaFace, and CurricularFace. Introduced by Deng et al. in 2019, ArcFace represents the first approach to integrate an additive angular margin within the loss function to maximize angular separability between classes [4]. By introducing this angular margin, ArcFace increases the inter-class distance while maintaining compact intra-class clusters, improving both robustness and accuracy.

1.2.1 Mathematical Formulation

ArcFace is based on the classical Softmax Loss but introduces an additive angular margin that enables better discriminability of facial embeddings. Specifically, by adding an angular margin to the angles between the embedding vector and the correct class center, ArcFace succeeds in improving the separability of vectors representing different individuals while ensuring that representations of faces within the same class are tightly clustered [4]. This approach can be visualized as a projection onto a hyperspherical manifold, where the feature embeddings are pushed apart by a fixed angular distance, leading to an improved clustering effect for each identity.

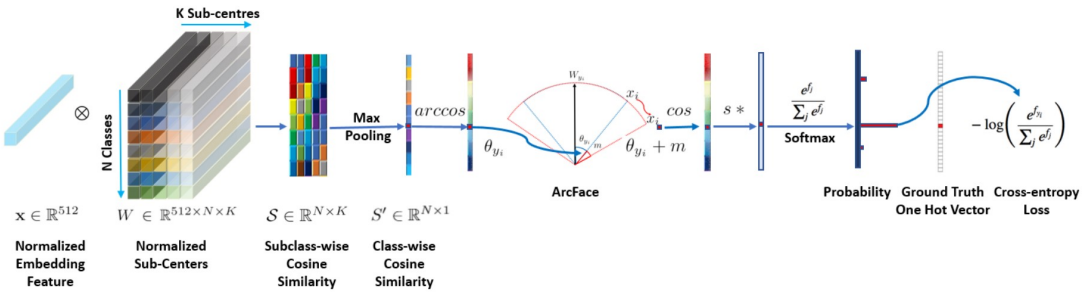


Figure 1.7: Diagram of the ArcFace Loss Mechanism. This figure illustrates the computation of the ArcFace loss function, showing the introduction of an angular margin m between the normalized feature vector and class center to enhance inter-class separability and intra-class compactness. The pipeline proceeds from normalized features and weights through the application of the angular margin, feature re-scaling, softmax calculation, and final cross-entropy loss.

The loss function is defined as follows:

$$L = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{s(\cos(\theta_{y_i}+m))}}{e^{s(\cos(\theta_{y_i}+m))}+\sum_{j\neq y_i}e^{s\cos\theta_j}} \tag{1.1}$$

Where:

- N is the number of samples in the mini-batch.
- θ_{y_i} is the angle between the embedding of sample i and the center of the correct class y_i.
- m is the additive angular margin.
- s is a scaling factor that enhances the angular separability of facial embeddings.

In the ArcFace loss function, the margin m is directly added to the angle between the embedding and the correct class center, thereby creating an artificial angular separation that promotes discrimination between different classes. This approach allows the model to obtain a more robust and discriminative representation of faces compared to the traditional Softmax Loss [4].

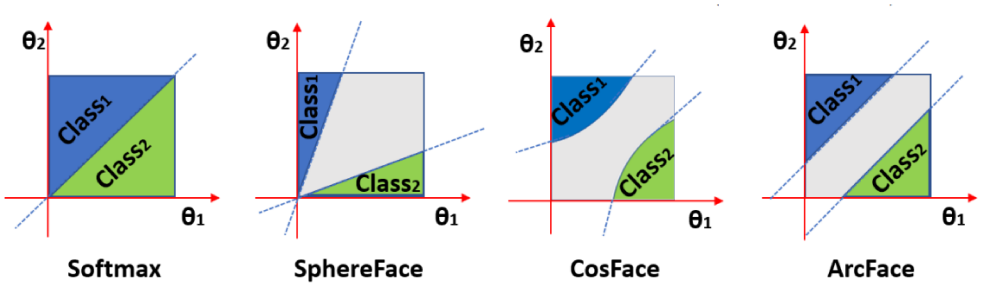
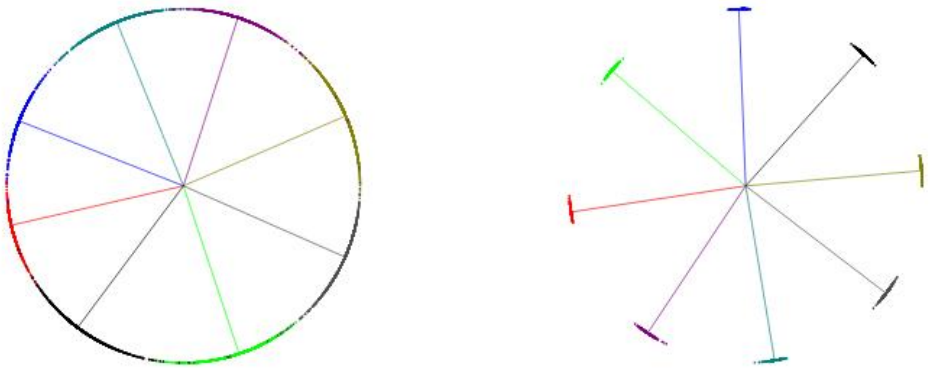


Figure 1.8: This figure illustrates the decision margins for various loss functions (Softmax, SphereFace, CosFace, and ArcFace) used in binary classification. Each plot represents the angle distributions (θ_1 and θ_2) between two classes during training. The shaded regions indicate the decision margins, with the dashed line representing the decision boundary. These margins show how each method modifies the angles between classes to improve class separability.

1.2.2 Empirical Performance

The effectiveness of ArcFace was rigorously evaluated on popular benchmarks such as LFW (Labeled Faces in the Wild), MegaFace, and IJB-C (IARPA Janus Benchmark-C), where it achieved state-of-the-art performance at the time of publication. In detail, a model based on the ResNet-100 architecture trained on the IBUG-500K dataset achieved an impressive accuracy of 99.83% on the LFW dataset and a True Accept Rate (TAR) of 99.08% at a False Accept Rate (FAR) of 10^{-6} on MegaFace, setting new benchmarks in the industry.

These results underscore the robustness of ArcFace in handling large-scale datasets with diverse facial variations, confirming its suitability for real-world facial recognition applications.



(a) Softmax (b) ArcFace

Figure 1.9: Toy examples under the Norm-Softmax and ArcFace loss on 8 identities with 2D features. Dots indicate samples and lines refer to the center direction of each identity. Based on the feature normalization, all face features are pushed to the arc space with a fixed radius. The geodesic distance margin between closest classes becomes evident as the additive angular margin penalty is incorporated [4].

These empirical results demonstrate that the introduction of the angular margin not only enhances the inter-class separability but also ensures tighter intra-class clustering. This characteristic has made ArcFace a foundational approach in face recognition tasks and a precursor to subsequent loss functions like MagFace, AdaFace, and CurricularFace, which further refine the margin-based strategy to address specific challenges such as sample quality and curriculum learning.

1.3 MagFace Loss

MagFace [5] introduces an innovative approach to facial recognition by not only enhancing class discrimination, as methods like ArcFace do, but also by incorporating a mechanism to assess image quality based on the magnitude of facial embeddings. One of the key innovations in MagFace is the integration of a magnitude-aware angular margin and a regularization term linked to the embedding magnitude, allowing for better handling of image variability. This represents a significant improvement over fixed-margin methods, which do not consider the varying quality of input images.

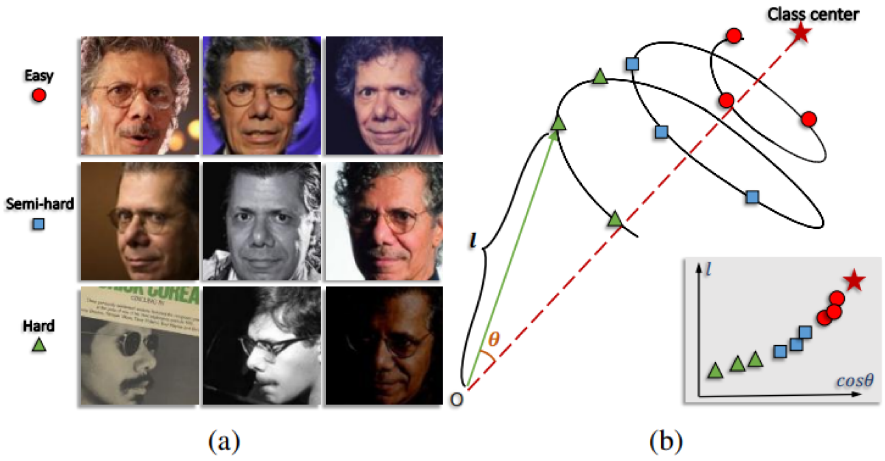


Figure 1.10: MagFace learns for (a) in-the-wild faces (b) a universal embedding by pulling the easier samples closer to the class center and pushing them away from the origin o. As shown in our experiments and supported by mathematical proof, the magnitude l before normalization increases along with feature's cosine distance to its class center, and therefore reveals the quality for each face. The larger the l, the more likely the sample can be recognized.

1.3.1 MagFace Loss formulation

MagFace stands out by incorporating *magnitude-aware learning*, where the embedding magnitude is directly related to image quality. This is achieved through two auxiliary functions: the angular margin $m(a_i)$, which varies according to the embedding magnitude $a_i = \|f_i\|$, and a regularization term $g(a_i)$ [5]. Unlike fixed-margin methods, MagFace adjusts the angular margin proportionally to the embedding magnitude, effectively tying the margin to the quality of the sample. High-quality samples, which generally have larger embedding magnitudes, receive a larger angular margin, enhancing angular separation between classes. Conversely, low-quality samples with smaller magnitudes receive a reduced margin, minimizing the risk of over-penalization.

The MagFace loss function is defined as:

$$L_{\text{Mag}} = \frac{1}{N}\sum_{i=1}^{N} L_i,\ L_i \quad = -\log \frac{e^{s\cdot\cos(\theta_{y_i}+m(a_i))}}{e^{s\cdot\cos(\theta_{y_i}+m(a_i))}+\sum_{j\neq y_i} e^{s\cdot\cos(\theta_j)}} + \lambda_g \cdot g(a_i), \tag{1.2}$$

where:

- N is the number of samples in the mini-batch.
- θ_{y_i} is the angle between the embedding of sample i and the center of the correct class y_i.
- s is the scale factor that amplifies angular separability.
- $m(a_i)$ is the angular margin that increases with embedding magnitude a_i.
- $g(a_i)$ is the regularization function, designed to penalize samples with small magnitudes.
- λ_g balances the regularization term and the classification loss.

The adaptive angular margin $m(a_i)$

1.3.2 Adaptive Angular Margin

One of the core innovations in MagFace is the adaptive angular margin $m(a_i)$, which varies depending on the magnitude of the embedding a_i. This approach addresses the limitations of fixed-margin methods by dynamically adjusting the angular margin based on the sample quality, thereby enhancing the inter-class separability for high-quality samples. The adaptive margin $m(a_i)$ is defined as:

$$m(a_i) = \frac{m_u - m_l}{m_u - m_l} \cdot (a_i - m_l) + m_l$$

Here:

- m_u and m_l are the upper and lower bounds of the margin.
- a_i is the embedding magnitude, which serves as an indicator of the quality of the sample.

High-quality samples, which typically have larger magnitudes a_i, receive a larger angular margin $m(a_i)$. This increases the angular separation between classes for these samples, leading to enhanced class discrimination. Conversely, low-quality samples, with smaller magnitudes, receive a reduced margin, minimizing the risk of over-penalization. This design allows MagFace to adaptively balance the learning process, effectively positioning high-quality embeddings closer to their class centers while managing low-quality samples [5].

1.3.3 Role of the Regularization Term

In addition to the adaptive margin, MagFace introduces a regularization term $g(a_i)$ to enforce a structured distribution of embedding magnitudes, enhancing robustness in high-variability conditions. The regularization function $g(a_i)$ is defined as:

$$g(a_i) = \frac{1}{m_u^2} \cdot a_i + \frac{1}{a_i}$$

This regularizer penalizes embeddings with magnitudes that fall outside a desired range, encouraging the model to produce embeddings that are not only separable across classes but also reflect the quality of the samples. By linking embedding magnitudes to sample quality, this regularization term helps high-quality samples (with large a_i) stay closer to their class centers, while low-quality samples are pulled towards the origin. This approach minimizes the negative impact of low-quality samples on the model, promoting intra-class compactness for reliable recognition even in the presence of noisy data [5].

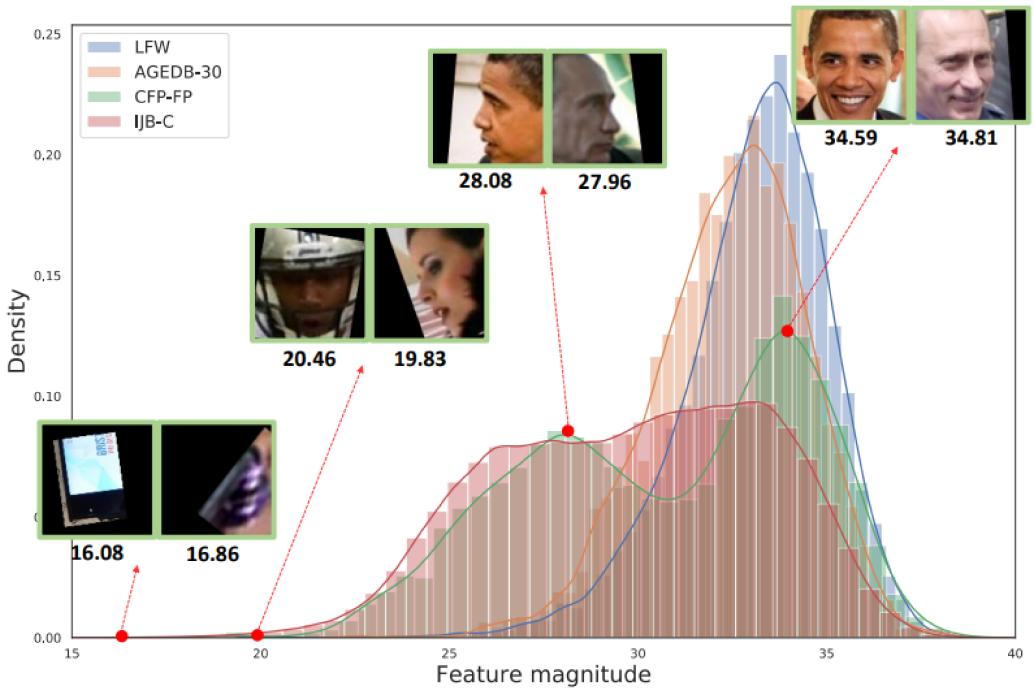


Figure 1.11: Distributions of magnitudes on different datasets

1.3.4 Empirical Performances

To validate the effectiveness of the MagFace loss, we conducted extensive experiments on several popular face recognition benchmarks. In this section, we detail the empirical results obtained and highlight the core advantages of MagFace over traditional fixed-margin methods.

On the LFW dataset, MagFace achieved an accuracy of 99.83%, surpassing fixed-margin approaches such as ArcFace and CosFace, which are frequently used as baselines. These results demonstrate the model's capability to accurately distinguish identities under controlled conditions, underscoring how MagFace's adaptive angular margin optimizes class separation for high-quality samples [5].

In less constrained scenarios, such as those encountered in the IJB-B and IJB-C datasets, MagFace consistently outperformed prior methods in terms of True Accept Rate (TAR) at various False Accept Rate (FAR) levels. Specifically, MagFace achieved a TAR of 94.33% at FAR 10^{-4} on IJB-B and 95.81% on IJB-C, demonstrating robustness against challenging conditions like pose variation, occlusion, and low image quality. These results underscore the effectiveness of MagFace's adaptive approach in handling high variability across samples [5].

1.3.5 Core Advantages and Applicability

MagFace sets a new standard in face recognition by combining adaptive margin management with quality-aware embeddings. This approach enables MagFace to handle samples of varying quality more effectively than fixed-margin methods, enhancing the model's resilience in highly variable environments.

The design of the MagFace regularization function ensures a structured distribution of embeddings, positioning high-quality samples close to their class centers and pushing lower-quality samples towards the origin. This architecture is easily integrable into cosine-similarity-based face recognition systems without significant modifications, making it a versatile solution for real-time applications and high-reliability recognition tasks.

1.4 CurricularFace Loss

Another innovative innovative approach is the CurricularFace loss function, that as introduced to improve the performance of deep learning models in face classification through adaptive curricilum learning [7]. CurricularFace introduces a more dynamic concept respect to the fixed margin of ArcFace, enabling the model to adapt during training by gradually emphasizing more difficult samples. In the initial phase, the model places greater importance on easier samples to facilitate convergence, while later shifting focus to harder samples, emphasizing their update during subsequent iterations [7]. This training progression makes CurricularFace more robust in handling the variability and complexity of the training data.

1.4.1 Mathematical Formulation

The formulation of CurricularFace derives from ArcFace but expands its core idea through dynamic modulation of negative cosine similarity for hard samples. The loss function can be expressed as follows:

$$L = -\log \frac{e^{s\cdot\cos(\theta_{y_i}+m)}}{e^{s\cdot\cos(\theta_{y_i}+m)} + \sum_{j\neq y_i} e^{s\cdot N(t^{(k)},\cos(\theta_j))}}, \tag{1.3}$$

$$N(t^{(k)},\cos(\theta_j)) = \begin{cases} \cos(\theta_j), & \text{if } \cos(\theta_{y_i}+m) \geq \cos(\theta_j), \\ (t^{(k)}+\cos(\theta_j))\cdot\cos(\theta_j), & \text{otherwise.} \end{cases} \tag{1.4}$$

- s is a scalar factor;

- θ_{y_i} is the angle between the weight vector of class y_i and the feature vector x_i;

- $N(t^{(k)},\cos(\theta_j))$ is the modulation function of negative cosine similarity, which varies based on the adaptive parameter $t^{(k)}$.

The key difference from ArcFace lies in the term, $N(t^k,\cos(\theta_j))$ which modulates the negative similarity according to the sample difficulty. For easy samples, the negative similarity is kept close to the original, while for difficult samples, it is altered to increase their importance, making training more efficient in handling complex cases [7].

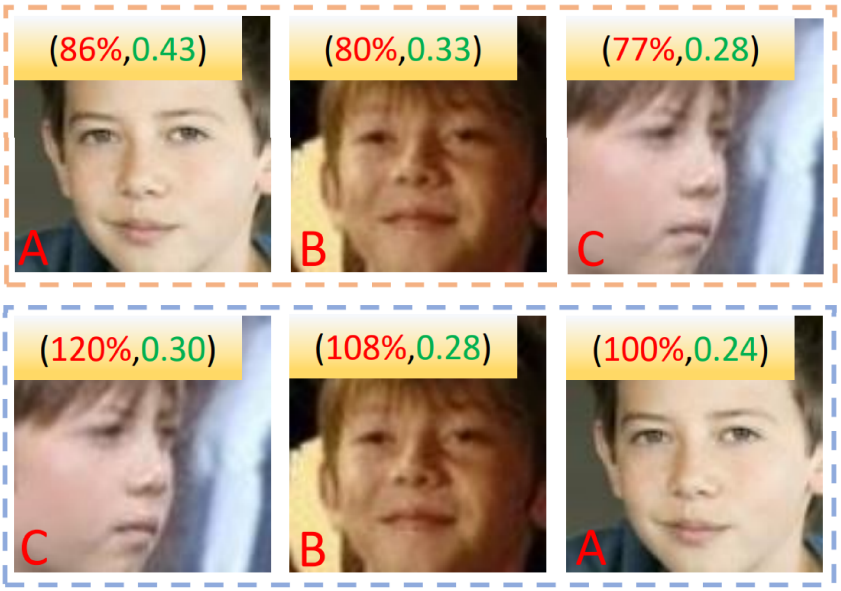


Figure 1.12: Illustrations on (ratio between CurricularFace loss and ArcFace in red, maximum cosθj in green) in different training stages. Top: Early training stage. Bottom: Later training stage

1.4.2 Adaptability Through Parameter t

One of the most significant innovations of CurricularFace is the introduction of the t parameter, which is automatically adapted during training. The value of t controls the degree of attention the model gives to difficult samples. At the start of training, t is low, emphasizing easy samples. As training progresses, t gradually increases, allowing the model to focus more on difficult samples. This adaptive process follows a curriculum learning strategy, similar to how humans learn simple concepts first and then move on to more complex ones [7].

The t parameter is updated using an exponential moving average (EMA) of the positive similarities within the mini-batch:

$$t^{(k)} = \alpha r^{(k)} + (1 - \alpha)t^{(k-1)} \tag{1.5}$$

Where $r^{(k)}$ is the average of the positive cosine similarities in the current mini-batch, and α is a momentum parameter, typically set to 0.99. This approach avoids the need for manual tuning of t and makes the learning process more stable, even in the presence of extreme samples [7].

1.4.3 Pseudocode for CurricularFace Loss Function

Below is the pseudocode that outlines the implementation details of the CurricularFace algorithm.

Algorithm 1 CurricularFace Algorithm

Initialize iteration count $k \leftarrow 0$
Initialize adaptive parameter $t \leftarrow 0$
Initialize margin $m \leftarrow 0.5$
while not converged **do**
    for each sample pair (θ_{y_i}, θ_j) **do**
        if $\cos(\theta_{y_i} + m) \geq \cos(\theta_j)$ **then**
            $N(t, \cos(\theta_j)) \leftarrow \cos(\theta_j)$
        else
            $N(t, \cos(\theta_j)) \leftarrow (t + \cos(\theta_j)) \cdot \cos(\theta_j)$
        end if
    end for
    Compute the loss L
    Compute gradients and update parameters
    Update t using a defined rule
end while
return optimized parameters

This algorithm initializes parameters and enters a loop that continues until convergence is reached. Within each iteration, the algorithm adjusts the penalties for misclassified samples based on their cosine similarities and the adaptive parameter t. The loss L is computed, and model parameters are updated accordingly. The parameter t is updated at the end of each iteration to gradually increase the emphasis on hard samples as the training progresses.

1.4.4 Empirical Results

Empirical evaluations highlight CurricularFace's advantage over well-known margin-based and mining-based loss functions, including ArcFace and MV-Arc-Softmax, across diverse face recognition benchmarks. CurricularFace adaptively emphasizes easy and hard samples at different stages of training, resulting in enhanced feature discrimination on test datasets. Achieving a high accuracy of 99.80% on LFW, CurricularFace performs comparably to ArcFace but outperforms other methods on pose-varied datasets like CFP-FP (98.37%) and CPLFW (93.13%), demonstrating robustness in non-ideal conditions. On the large-scale IJB-B and IJB-C datasets, CurricularFace reaches TARs of 94.8% and 96.1% at a FAR of 10^{-4}, showcasing its capability to handle challenging scenarios with high identity diversity. This performance is attributed to the dynamic adjustment of hard sample penalties and the adaptive tuning of the t parameter, allowing the model to focus more effectively on meaningful samples in later training stages and to converge more stably compared to methods with fixed margins or preset weights for hard samples.

Figure 1.13: Easy and hard examples from two subjects classified by CurricularFace on early and later training stage, respectively. Green box indicates easy samples. Red box indicates hard samples. Blue box means samples are classified as hard in early stage but relabeled as easy in later stage, which indicates samples' transformation from hard to easy during the training procedure.

1.5 AdaFace Loss

AdaFace proposes a novel approach to improving the robustness of face recognition models by dynamically adapting the margin based on image quality. The underlying motivation behind AdaFace is to address the challenge posed by low-quality or unidentifiable images during training, a common problem in real-world face recognition tasks. The model adjusts the margin function based on the feature norm, a quantity derived from the feature representation of the image, to handle samples of varying quality more effectively [6]. The feature norm is computed similarly to the magnitude in MagFace Loss, but its application is different.

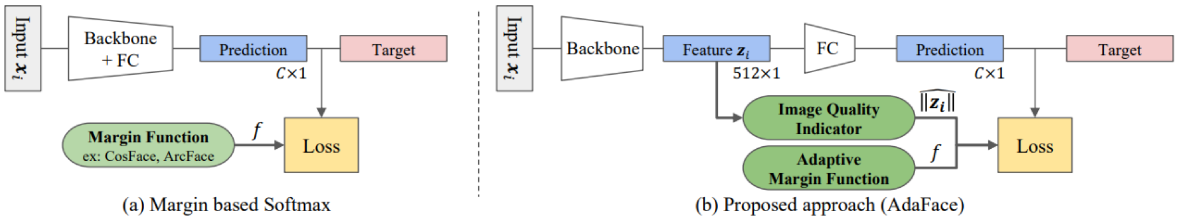


Figure 1.14: Conventional margin based softmax loss vs AdaFace. A framework training pipeline with a margin based softmax loss (a). The loss function takes the margin function to induce smaller intra-class variations.(b) Proposed adaptive margin function (AdaFace) that is adjusted based on the image quality indicator.

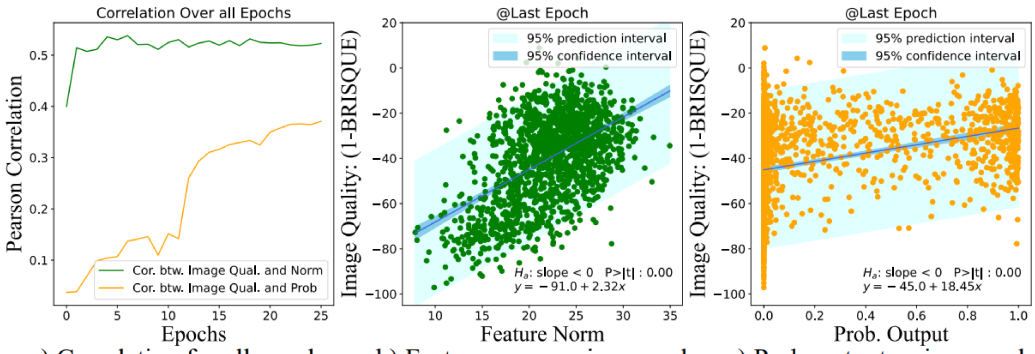
1.5.1 Image Quality Indicator

The key insight of AdaFace is that the feature norm (measure of the magnitude of the feature vector) correlates with image quality: high-quality images tend to produce feature vectors with larger norms, while low-quality images result in smaller norms. AdaFace uses this correlation to adapt the margin in the loss function, with two main goals:

- Emphasize hard samples for high-quality images, pushing the model to focus on difficult decision boundaries.

- De-emphasize hard samples for low-quality images, preventing the model from overfitting to noisy or unrepresentative data.

To achieve this, AdaFace defines two margin components: an angular margin and an additive margin, both are dependent from the feature norm $||z_i||$ [6].



a) Correlation for all epochs b) Feature norm vs img. qual. c) Prob. output vs img. qual.

Figure 1.15: Correlation between Feature Norm and Image Quality across Training Epochs. This figure demonstrates how the feature norm $||z_i||$ correlates with image quality, measured by BRISQUE (Blind/Referenceless Image Spatial Quality Evaluator), a score where higher values indicate lower image quality. Pearson's correlation coefficient is used to show the strength of the linear relationship between feature norm and image quality, ranging from -1 (strong negative) to 1 (strong positive).
(a) The green curve shows the increasing correlation of feature norm with image quality over training epochs, confirming feature norm as a reliable indicator of quality. The orange curve shows a weaker correlation between the probability P_{y_i} (confidence for the true class) and image quality.
(b) Scatter plot of feature norm vs. image quality, showing a positive relationship: higher norms correlate with higher quality images.
(c) Scatter plot P_{y_i} of vs. image quality, illustrating a weaker, non-linear relationship, supporting feature norm as a more effective quality indicator.

1.5.2 Mathematical Formulation

The loss function of AdaFace it is very similar, like the other loss functions presented in this chapter to ArcFace Loss but with a different formulation of the margin. The formula for the adaptive margin function in AdaFace is given by:

$$f(\theta_j, m)_{\text{AdaFace}} = \begin{cases} s\left(\cos(\theta_{y_i} + g_{\text{angle}}) - g_{\text{add}}\right) & \text{if } j = y_i \\ s\cos(\theta_j) & \text{if } j \neq y_i \end{cases} \tag{1.6}$$

The margin is modified depending on the image quality. If $j = y_i$ (correct class), both angular and additive margins are applied. If $j \neq y_i$ (incorrect class), only the original cosine similarity is used.

$$g_{\text{angle}} = -m \cdot \widehat{\|\mathbf{z}_i\|} \tag{1.7}$$

$$g_{\text{add}} = m \cdot \widehat{\|\mathbf{z}_i\|} + m \tag{1.8}$$

These functions define how the angular (g_{angle}) and additive (g_{add}) margins adapt based on the normalized feature norm $\widehat{\|\mathbf{z}_i\|}$.

$$\widehat{\|\mathbf{z}_i\|} = \text{clip}\left(\frac{\|\mathbf{z}_i\| - \mu_z}{\sigma_z/h}, -1, 1\right) \tag{1.9}$$

The norm $\|\mathbf{z}_i\|$ is adjusted using the mean μ_z and standard deviation σ_z of the norms across the batch, and then clipped between -1 and 1.

$$\mu_z^{(k)} = \alpha\mu_z^{(k)} + (1-\alpha)\mu_z^{(k-1)} \tag{1.10}$$

$$\sigma_z^{(k)} = \alpha\sigma_z^{(k)} + (1-\alpha)\sigma_z^{(k-1)} \tag{1.11}$$

These formulas are used to stabilize the mean and the standard deviation by using an exponential moving average [6].

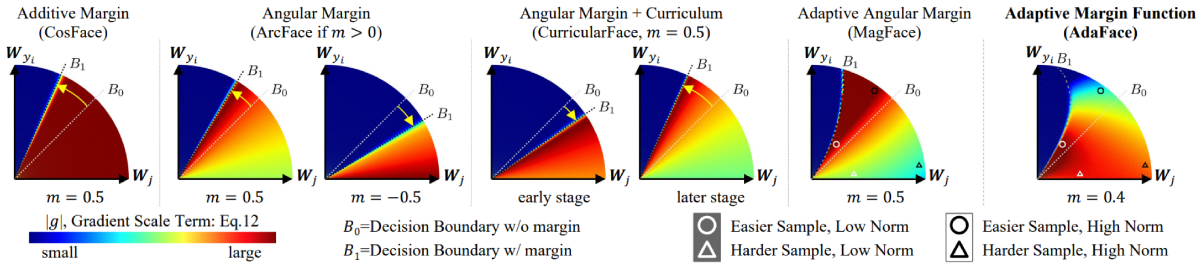


Figure 1.16: Comparison of Different Margin-Based Approaches in Feature Space. This figure provides a visual comparison of various margin-based approaches, including the angular margin of ArcFace, the adaptive angular margin of MagFace, and the quality-adaptive margin function of AdaFace. The arcs and decision boundaries represent how each method positions samples based on both quality and difficulty.
AdaFace's margin function is shown to adjust adaptively according to the feature norm (indicating image quality), placing greater emphasis on higher-quality, hard samples while down-weighting unidentifiable samples with low feature norms. The illustration highlights how AdaFace leverages adaptive components g_{angle} and g_{add} to dynamically shift the decision boundaries based on sample quality, unlike fixed-margin approaches.

1.5.3 Empirical Results

Empirical evaluations demonstrate AdaFace's adaptability and superior performance across high- and low-quality face recognition benchmarks. On high-quality datasets, such as LFW, AdaFace achieves a notable accuracy of 99.83%, performing on par with state-of-the-art methods while maintaining robustness without sacrificing accuracy. In mixed-quality datasets like IJB-C, AdaFace achieves a True Accept Rate (TAR) of 96.89% at a False Accept Rate (FAR) of

10^{-4}, marking a significant reduction in error rates, especially when compared to competitive methods. This boost in performance is attributed to AdaFace's quality-adaptive margin, which dynamically adjusts the emphasis on hard samples based on image quality, thereby improving discriminative capacity. Furthermore, on low-quality datasets such as TinyFace, AdaFace achieves a Rank-1 accuracy improvement of 3.5% on average across key performance metrics. These empirical results underscore the efficacy of using feature norm as a proxy for image quality, enabling AdaFace to deliver top-tier results even under challenging conditions without additional computational overhead. This adaptability makes AdaFace particularly suitable for applications where image quality is inconsistent, such as in surveillance or low-resolution settings.

2

Development of a Novel Loss Function for Face Recognition

In face recognition, model accuracy and robustness are significantly impacted by the choice of loss function during training. Although existing loss functions, such as ArcFace, MagFace, and CurricularFace, have improved recognition performance, they still face challenges with variations in image quality and difficulty levels. This chapter introduces a novel loss function that combines the strengths of MagFace and CurricularFace.

The main concept is to merge the adaptive margin approach from CurricularFace with the feature norms of MagFace, aiming to develop a more robust model capable of handling diverse image qualities. This integration addresses the limitations of current loss functions, particularly their sensitivity to noisy data and fixed margin settings, thereby enhancing generalization across various datasets.

All developments were implemented in MATLAB, allowing for seamless integration with pre-trained models from the AdaFace repository. Fine-tuning these models with the new loss function aims to improve face recognition across different backgrounds and qualities. This chapter will detail the mathematical formulation of the loss function, the selection of pre-trained models, the training datasets employed, and the experimental setup for validation.

2.1 Mathematical Formulation

The novel loss function proposed in this chapter is designed to enhance the robustness and adaptability of face recognition models by integrating multiple concepts derived from existing loss functions. The overall loss is expressed as follows:

$$L = L_{ce} + L_{reg} \tag{2.1}$$

where L is the total loss, L_{ce} represents the cross-entropy loss, and L_{reg} denotes the magnitude regularization term.

2.1.1 Cross-Entropy Loss

The cross-entropy loss serves as the foundational component of our model, measuring the discrepancy between the predicted class probabilities and the true labels. This loss function is critical to the performance of face recognition models and is also a key element in the previously discussed models such as ArcFace. It is defined mathematically as:

$$L_{ce} = -\log \frac{e^{s\cdot\cos(\theta_{y_i}+m_{adaptive})}}{e^{s\cdot\cos(\theta_{y_i}+m_{adaptive})} + \sum_{j\neq y_i} e^{s\cdot N(t^{(k)},\cos(\theta_j))}} \tag{2.2}$$

In this equation, s represents the scaling factor applied to the cosine similarity values, while θ_{y_i} denotes the angle corresponding to the true class label y_i. The adaptive margin $m_{adaptive}$ is integrated into the loss, allowing the model to dynamically adjust based on the quality of the facial embeddings.

2.1.2 Magnitude and Clamping

The calculation of the magnitude of the facial embeddings and the subsequent clamping of this magnitude is done to ensure it remains within a predefined range. The raw magnitude, $magnitude_{raw}$, is calculated as the Euclidean norm of the facial embedding vectors, and is given by:

$$magnitude_{raw} = \|\mathbf{x}_i\|_2 \tag{2.3}$$

where $\mathbf{x}_i$ is the facial embedding of sample i, and $\|\mathbf{x}_i\|_2$ denotes its L_2-norm, which computes the magnitude of the embedding vector in Euclidean space. To prevent extreme magnitudes that might affect the stability of the model, we apply clamping to restrict the magnitude within predefined upper and lower bounds, denoted by m_u and m_l, respectively. The clamped magnitude is expressed as:

$$magnitude_{clamped} = \max\left(\min\left(magnitude_{raw}, m_u\right), m_l\right) \tag{2.4}$$

This clamping ensures that the magnitude is neither too large nor too small, thereby preventing numerical instability during training and improving the model's ability to generalize across different input qualities.

2.1.3 Adaptive Margin Calculation

To ensure that the model can adapt to varying image qualities, we calculate the adaptive margin based on the feature norms of the embeddings. This is expressed as:

$$m_{adaptive} = \left(\frac{margin_u - margin_l}{magnitude_u - magnitude_l}\right) \cdot (magnitude_{clamped} - magnitude_l) + margin_l \tag{2.5}$$

Here, $margin_u$ and $margin_l$ represent the upper and lower bounds of the margin, respectively; $magnitude_u$ and $magnitude_l$ are upper and lower bounds of the magnitude while $magnitude_{clamped}$ is derived from the normalized magnitude of the facial embeddings. This dynamic margin effectively adjusts the difficulty level for each sample, depending on its quality, thereby allowing the model to focus on more challenging instances without compromising overall performance. This formulation is derived from the code repository of MagFace [5].

2.1.4 Magnitude Regularization

In addition to the cross-entropy loss, we incorporate a magnitude regularization term to further enhance the model's capability in handling variations in embedding magnitudes. The magnitude regularization is computed as follows:

$$L_{reg} = \lambda \cdot \text{mean}\left(\frac{1.0}{magnitude_u^2} \cdot magnitude_{clamped} + \frac{1.0}{magnitude_{clamped}}\right) \tag{2.6}$$

In this equation, λ is a hyperparameter that controls the weight of the regularization term, guiding the model's sensitivity to the magnitudes of the embeddings. By penalizing embeddings based on their magnitude, the model is encouraged to produce more consistent and reliable representations across different input samples.

2.1.5 Penalty Mechanism for Difficult Samples

To address the challenge of difficult samples, we implement a penalty mechanism for incorrect classes, which is articulated mathematically as:

$$N(t^{(k)}, \cos(\theta_j)) = \begin{cases} \cos(\theta_j), & \text{if } \cos(\theta_{y_i} + m_{adaptive}) \geq \cos(\theta_j), \\ (t^{(k)} + \cos(\theta_j)) \cdot \cos(\theta_j), & \text{otherwise.} \end{cases} \tag{2.7}$$

This mechanism, inherited from the approach used in CurricularFace, ensures that when the cosine similarity of an incorrect class exceeds the adjusted threshold defined by the adaptive margin, a penalty is applied. The parameter $t^{(k)}$ represents the threshold that influences the penalty applied to the incorrect classes, thereby making the model more sensitive to challenging examples that are often misclassified.

2.1.6 Updating Parameter t

To further refine the model's performance, the parameter t is updated after each iteration as follows:

$$t = 0.01 \cdot \text{mean}(cos_{y_i}) + 0.99 \cdot t \tag{2.8}$$

This update rule, as described in CurricularFace, allows the model to adaptively adjust the penalty threshold based on the average cosine similarity of the correct class embeddings. This mechanism effectively maintains a balance between sensitivity and robustness across the training process, enabling the model to handle difficult samples without over-penalizing them.

Algorithm 2 Custom Loss Function with MagFace and CurricularFace Adjustments

Initialize iteration count $k \leftarrow 0$
Initialize adaptive margin parameters: mag_u, mag_l, m_u, m_l
Initialize regularization weight λ
Initialize parameter $t \leftarrow 0$
while not converged **do**
  for each input batch (X, T) **do**
    Compute model forward pass: $X, state \leftarrow \text{forward}(net, X)$
    Extract embeddings $X \leftarrow \text{squeeze(X)}$
    Calculate embedding magnitudes: $mag\_raw \leftarrow ||X||_2$
    Clamp magnitudes: $mag\_c \leftarrow \text{clamp}(mag\_raw)$
    Compute adaptive margin $m\_adaptive \leftarrow \frac{m\_u - m\_l}{mag\_u - mag\_l} \cdot (mag\_c - mag\_l) + m\_l$
    Compute $mag\_reg \leftarrow \frac{1}{mag\_u^2} \cdot mag\_c + \frac{1}{mag\_clamped}$
    Compute magnitude loss $mag\_loss \leftarrow \lambda \cdot \text{mean}(mag\_reg)$
    Retrieve classification layer weights W and biases $b \leftarrow 0$
    $X \leftarrow \text{fullyconnect}(X, W, b)$
    Normalize embeddings $X \leftarrow X/||X||_2$
    $cos\_theta \leftarrow X$
    $cos\_theta\_m \leftarrow cos(\theta_{y_i}) \cdot cos(m\_adaptive) - sin(\theta_{y_i}) \cdot sin(m\_adaptive)$
    $cond\_mask \leftarrow (cos\_theta - cos(\pi - m\_adaptive)) \leq 0$
    $keep\_val \leftarrow cos\_theta - layer.mm$
    $cos\_theta\_m[cond\_mask] \leftarrow keep\_val[cond\_mask]$
    Initialize mask $mask \leftarrow$ true for all incorrect classes
    Retrieve correct class cosines cos_yi
    Create penalty mask $p\_mask \leftarrow (cos\_yi + 0.5 \leq cos\_theta)$ & $mask$
    Apply penalty to difficult samples:
    $cos\_theta[p\_mask] \leftarrow (t + cos\_theta[p\_mask]) \cdot cos\_theta[p\_mask]$
    $t \leftarrow 0.01 \cdot \text{mean}(cos\_yi) + 0.99 \cdot t$
    Set output logits $output \leftarrow cos\_theta$
    Update logits for correct classes $output \leftarrow cos\_theta\_m$
    $Z \leftarrow s \cdot output$
    $logit\_softmax \leftarrow \text{softmax}(Z)$
    $loss\_ce \leftarrow \text{crossentropy}(logit\_softmax, T)$
    $loss \leftarrow loss\_ce + mag\_loss$
    $gradients \leftarrow \text{dlgradient}(loss, weights, parameters)$
  end for
end while
return optimized parameters

2.2 Pre-trained Models and Fine-tuning

In my research, I utilized two pre-trained models from the AdaFace repository, each based on different ResNet architectures: ResNet-100 trained on the WebFace12M dataset and ResNet-50 trained on WebFace4M. Before importing these models into MATLAB, I had to perform tracing of the PyTorch models, a necessary step for compatibility with MATLAB's conversion tools. This tracing process essentially involves running the model with sample inputs in PyTorch to record its execution graph, transforming it into a static model that MATLAB can interpret. This process is required by the Deep Learning Toolbox Converter for PyTorch Models in order to properly import and modify the architecture.

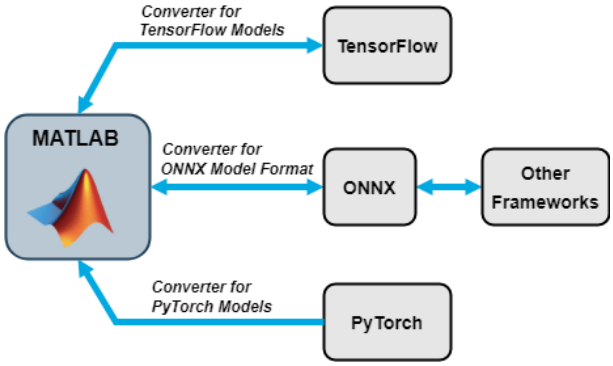


Figure 2.1: format of models that can be imported into MATLAB

Once the models were successfully imported into MATLAB using the Deep Learning Network Designer, I encountered some challenges with the structure of the final layers. One specific issue was that certain layers, such as the flatten operation, were not recognized by MATLAB. To resolve this, I replaced the flattening operation with a Global Average Pooling layer ('global2daverage'), which achieves similar functionality while being compatible with MATLAB's framework. I also customized the last layer to output the desired embedding for face recognition, adapting the architecture to align with the requirements of my loss function.

Furthermore, I applied final fine-tuning steps to adjust the input size and included a dropout layer to enhance the model's robustness during training, ensuring that the network remains resilient to overfitting. Through this approach, I was able to finalize a pre-trained architecture that is now tailored to support my novel loss function.

2.3 Training Datasets

In my research, as previously mentioned, I utilized neural networks pre-trained with AdaFace, which were then further fine-tuned using newly developed loss functions. The pre-training phase involved several datasets, including WebFace4M, WebFace12M, and MS1MV2, chosen for their wide coverage of identities and facial attributes. For the fine-tuning process that I conducted, I worked primarily with MS1MV3 and WebFace4M. The following sections will describe these datasets in more detail, highlighting their unique characteristics and their relevance to my experiments.

2.3.1 MS1MV2, MS1MV3

The MS1MV2 and MS1MV3 datasets are two widely used benchmarks for face recognition tasks, both derived from the original MS-Celeb-1M dataset. These datasets have become essential for training deep learning models in facial recognition due to their large scale and variety of identities[13].

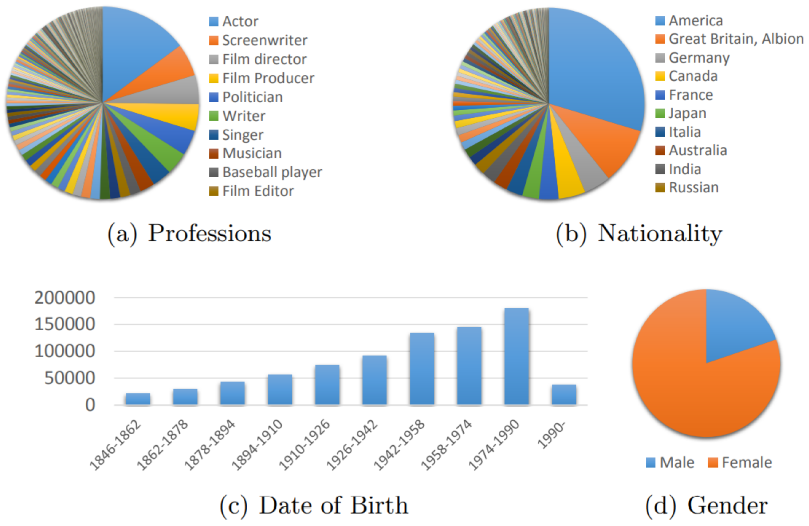


Figure 2.2: Distribution of faces inside the original MS-Celeb-1M dataset

- MS1MV2: The MS1MV2 dataset is a cleaned version of the original MS-Celeb-1M, addressing issues of noise in the data. MS-Celeb-1M originally contained around 10 million images of 100,000 celebrities, but it was notorious for containing a high percentage of mislabeled or low-quality images. MS1MV2 was created to clean up these inconsistencies, leading to a refined dataset with around 5.8 million images and 85,000 identities. The cleaning process involved both manual and automated steps to reduce the noise level, ensuring a higher quality of data for training. It has been extensively used in training models

for face verification and identification due to its scale and diversity, which cover a wide range of poses, lighting conditions, and facial expressions [13].

- MS1MV3: The MS1MV3 dataset is a further improved version of MS1MV2, offering even cleaner data with higher quality annotations. The primary objective in creating MS1MV3 was to remove duplicate images, as well as further reduce label noise, which was still present in MS1MV2. MS1MV3 retains approximately the same number of identities as MS1MV2 but improves the accuracy of the labels and reduces overlapping images. This dataset has been particularly useful for training models that are evaluated on high-stakes benchmarks like IJB-C and LFW, where even small errors in annotation can significantly affect performance [13].

Both MS1MV2 and MS1MV3 have been instrumental in developing state-of-the-art face recognition models and are commonly used in conjunction with advanced loss functions like ArcFace and AdaFace.

2.3.2 WebFace260M

The WebFace260M dataset is one of the largest benchmarks for facial recognition, offering diversity in faces, poses, and demographic attributes. It was created by gathering images of celebrities from Freebase and IMDB, resulting in a collection of over 265 million images from approximately 4 million identities. Not all subjects had publicly available images, so the final dataset included varying numbers of images per identity based on popularity [14].

To clean the noisy images collected from the web, the CAST (Cross-Architecture Self-Training) framework was used. This involved a ResNet-100 model trained with ArcFace on the MS1MV2 dataset, which filtered noisy data via clustering. This iterative process reduced noise and created a cleaner subset called WebFace42M (42 million images, 2 million subjects), with less than 10% noise, a significant improvement over datasets like MegaFace2 and MS1M, which had noise levels of 30%-50% [14].

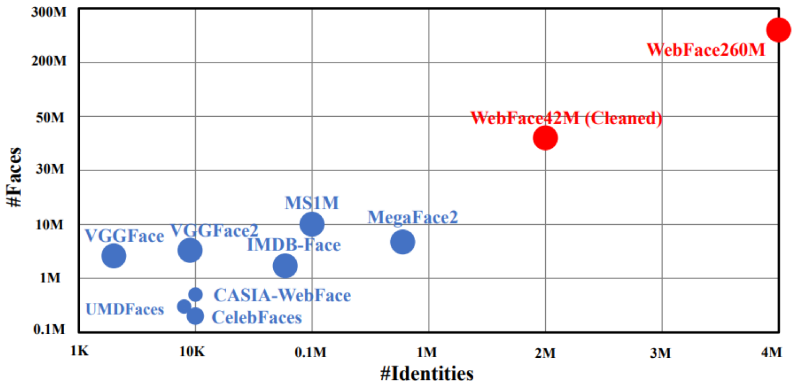
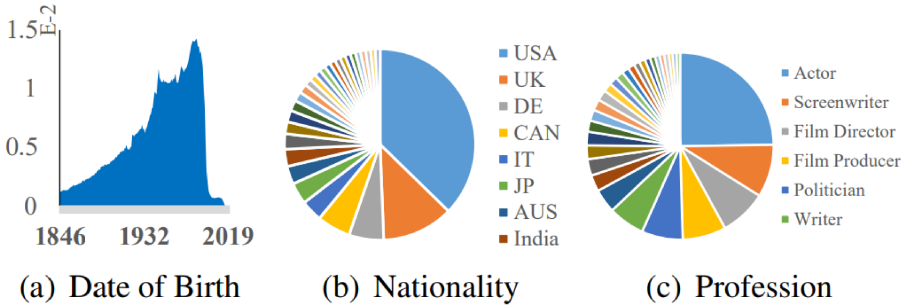


Figure 2.3: Comparisons of identities and faces between WebFace dataset and others public training set.

For preprocessing, facial alignment was done using RetinaFace, selecting only the largest, most prominent face in each image. This process ensured the dataset was clean and ready for large-scale facial recognition experiments.

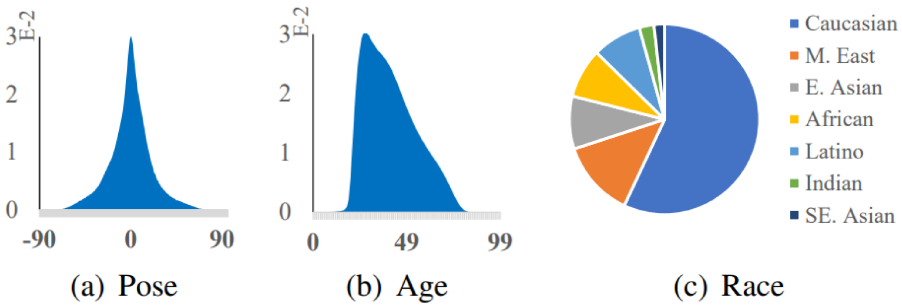
In my research, I had to deal with two subsets of WebFace42M, WebFace12M and WebFace4M, to train ResNet-100 and ResNet-50 networks. These subsets provided a good balance between dataset size and computational complexity. The diversity in poses, age, and demographics allowed the models to generalize well, particularly for real-world facial recognition challenges. These subsets were highly suitable for testing on benchmarks like IJB-C.

WebFace42M includes detailed face annotations such as pose, age, gender, race, and accessories (e.g., glasses), making it a rich dataset for deep learning. Models trained on these subsets can perform well even in challenging conditions due to this demographic diversity.



(a) Date of Birth (b) Nationality (c) Profession

Figure 2.4: Date of birth, nationality and profession of WebFace260M



(a) Pose (b) Age (c) Race

Figure 2.5: Pose (yaw), age and race of WebFace42M

2.4 Experimental Setup

The model training was conducted on a high-performance supercomputer cluster ”Blade” at the D.E.I. department of the University of Padua, equipped with Nvidia RTX 3090 and A40 GPUs. This setup was essential to handle the computational demands of training a deep learning model on a large-scale dataset such as WebFace260M and its variants. MATLAB, specifically its Deep Learning Toolbox, was the primary environment for this implementation, facilitating data handling, visualization, and custom function development. Critical support functions, such as minibatchqueue, enabled efficient processing of mini-batches, optimizing memory use and training speed.

For the base model architecture, a pre-trained ResNet-50 was fine-tuned using a combination of the MagFace and CurricularFace loss functions to enhance facial feature discrimination. The model processed input images resized to [112, 112], producing a 512-dimensional embedding vector for each image. Since the pre-trained models from the AdaFace repository were trained on images in BGR format, I incorporated a custom function within the minibatchqueue to convert images from RGB to BGR by rearranging the color channels. Training parameters were carefully chosen, including a mini-batch size of 256 (Thanks to the 48 GB memory of the A40 GPU, I set the minibatchsize to 192 when using an RTX 3090), a total of 30 epochs, an initial learning rate of 0.01, a momentum parameter of 0.9, and a learning rate drop factor of 0.5 applied every 6 epochs. Training the ResNet-50 on this setup required approximately two weeks to reach convergence.

Key hyperparameters for the loss function were set based on recommendations from the official MagFace repository to ensure optimal model performance. Specifically, the lower margin $margin_l$=0.4, upper margin $margin_u$=0.8, minimum magnitude $magnitude_l$=5, and maximum magnitude $magnitude_u$=10 followed these guidelines. Additionally, the adaptive margin calculation was implemented according to the repository’s proposed formula, where the margin adjusts in response to embedding magnitudes. This margin dynamic introduces a proportional penalty based on the model’s confidence, ensuring that embeddings with lower magnitudes receive a smaller margin, and higher-magnitude embeddings receive a larger one, directly influencing the classification boundary.

Further, the loss function was designed to combine MagFace’s regularization on embedding magnitudes with CurricularFace’s penalization for difficult samples. This design helps manage the model’s response to challenging cases by adaptively updating the t parameter based on the cosine values for correctly classified samples, a key feature of CurricularFace. In each epoch, the model’s loss and accuracy were tracked to monitor convergence, and model weights were saved as checkpoints to secure progress and allow for recovery in case of interruptions. This ro-

bust approach ensured consistent model improvement and alignment with state-of-the-art facial recognition techniques.

Figure 2.6: Value of the loss during training

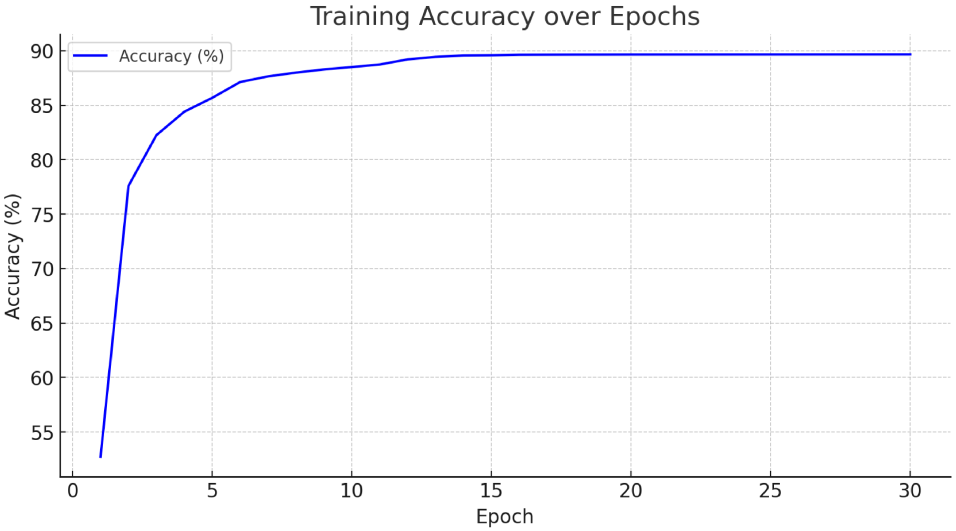


Figure 2.7: Percentage of accuracy value during training

3

Test IJB-C (IARPA Janus Benchmark-C)

This chapter describes the IJB-C benchmark, used to evaluate the performance of the developed models. The IJB-C benchmark is recognized as one of the most comprehensive and challenging in the field of facial recognition, due to its ability to include realistic scenarios with high variability in images, such as different lighting conditions, poses, and facial qualities.

Throughout the chapter, the verification and identification protocols required by the benchmark are introduced, providing a clear definition of how the models should be tested. Additionally, the process of embedding aggregation for templates is illustrated, which is essential to combine the representations of images belonging to the same individual. Two aggregation strategies are discussed in detail: the simple mean of embeddings and an advanced method called ERS (Enhanced Representation by Subsampling), which takes into account the recognizability of individual images.

This chapter focuses on describing the techniques and experimental setups employed, laying the groundwork for the detailed analysis of the results, which will be presented in the following chapter.

3.1 ROC Curve and AUC Value

The ROC curve is a graphical representation of the trade-off between the True Positive Rate (TPR) (sensitivity) and the False Positive Rate (FPR) (1-specificity) as the decision threshold varies. The TPR measures the proportion of correctly identified true positives relative to the total number of actual positives, while the FPR measures the proportion of false positives relative to the total number of actual negatives.

To construct an ROC curve:

- Compute similarity scores (e.g., cosine similarity) between template pairs.
- Apply different decision thresholds to classify pairs as matches (positive) or non-matches (negative).
- For each threshold, calculate the TPR and FPR values, which form the points on the ROC curve.

An ideal ROC curve approaches the upper left corner of the graph, where the TPR reaches 100% and the FPR is 0%, indicating a perfect model [15].

Inst#	Class	Score	Inst#	Class	Score
1	p	.9	11	p	.4
2	p	.8	12	n	.39
3	n	.7	13	p	.38
4	p	.6	14	n	.37
5	p	.55	15	n	.36
6	p	.54	16	n	.35
7	n	.53	17	p	.34
8	n	.52	18	n	.33
9	p	.51	19	p	.30
10	n	.505	20	n	.1

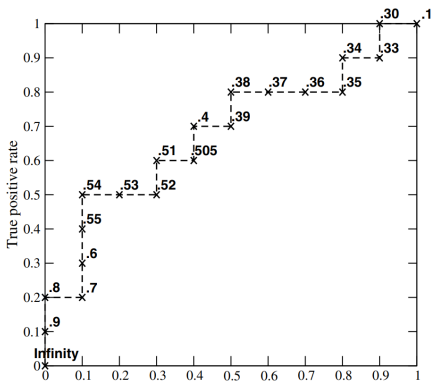


Figure 3.1: This figure demonstrates the construction of a Receiver Operating Characteristic (ROC) curve. The table above lists instances with their respective scores and ground-truth classes (positive or negative). By progressively lowering the decision threshold and recalculating the True Positive Rate (TPR) and False Positive Rate (FPR) for each threshold, a ROC curve is generated. The x-axis represents the FPR, while the y-axis represents the TPR.

3.2 Interpretation of AUC

The Area Under the Curve (AUC) is a scalar value representing the area under the ROC curve. It provides a summary of a model's performance across all possible thresholds:

- AUC = 1.0: Perfect performance; the model always distinguishes between positives and negatives.
- AUC = 0.5: No discriminative ability; equivalent to random guessing.
- AUC > 0.7: Indicates acceptable discrimination.
- AUC > 0.8: Represents good discrimination.
- AUC > 0.9: Signifies excellent discrimination [16].

In face recognition, the AUC is particularly valuable for comparing different models or configurations, as it provides a single metric summarizing their effectiveness over all thresholds.

3.3 Structure of the IJB-C Dataset

The IARPA Janus Benchmark–C (IJB-C) dataset represents one of the most challenging and comprehensive benchmarks currently available for evaluating face recognition systems under real-world, unconstrained conditions. As an extension of the IJB-B dataset, IJB-C includes an expanded variety of media and protocols to facilitate the evaluation of face detection, verification, and identification tasks.

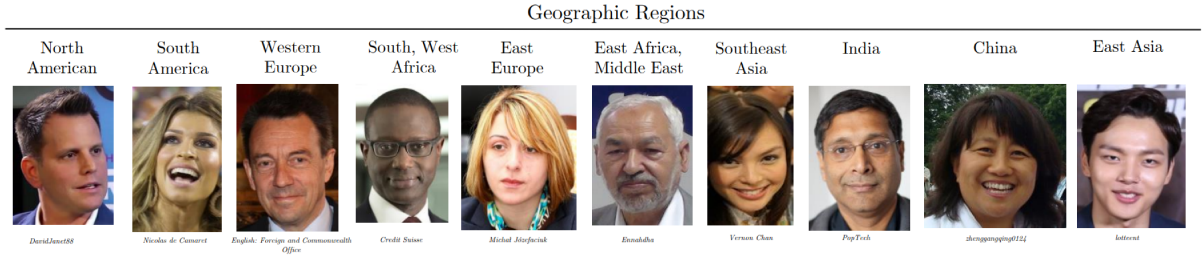


Figure 3.2: Examples of subjects included in IJB-C from various geographic regions.

The IJB-C dataset is composed of:

- Images: A total of 31,334 images, including 21,294 face images and 10,040 non-face images. This diverse collection represents a wide range of facial variations, from controlled frontal faces to in-the-wild images with varying quality, occlusions, and head poses.
- Video Frames: 117,542 frames extracted from 11,779 video clips, capturing natural variations in facial expressions, pose, and environmental factors. These frames enable a more dynamic assessment of recognition models.

- Subjects: The dataset includes 3,531 unique identities, spanning different demographics and backgrounds. This diversity ensures that models are tested against a broad set of identities, which is critical for real-world applicability.

- Annotations: IJB-C provides manually annotated bounding boxes for faces in both images and frames, along with covariate labels such as occlusion and pose angles. These annotations allow for more granular performance analysis and facilitate the evaluation of specific factors affecting recognition accuracy [13].

Figure 3.3: Annotation Labels included within IJB-C

3.4 Verification and Identification Protocols

The IJB-C dataset offers well-defined protocols for both verification and identification tasks, each of which is structured to evaluate different aspects of face recognition:

- Verification: the task is to determine whether two face images belong to the same individual. This is achieved by comparing pairs of face templates and calculating similarity scores (e.g., cosine similarity). The verification protocol is particularly useful for applications where confirming an identity match is critical, such as in access control systems.

- Identification: the objective is to recognize an individual's identity from a gallery set of known subjects. This task simulates scenarios where a query face is compared against a database of identities, as would occur in security or surveillance applications. Identification involves both closed-set (where the query face is guaranteed to match a subject in the gallery) and open-set (where the query face may not have a match in the gallery) scenarios.

These protocols provide a comprehensive framework for evaluating both the discriminative power and the robustness of face recognition models [13].

3.5 Key Metrics: TAR@FAR and AUC

To evaluate model performance on the IJB-C dataset, two primary metrics are employed: True Acceptance Rate at a given False Acceptance Rate (TAR@FAR) and Area Under the Curve.
The TAR@FAR metric measures the proportion of correctly verified identity matches (True Acceptances) at specific False Acceptance Rate (FAR) thresholds. For instance, TAR@FAR=0.001 represents the percentage of true positives achieved when the false positive rate is constrained to 0.1%. This metric is particularly valuable in security-sensitive applications, where minimizing false acceptances is crucial, as it reflects the model's ability to balance high accuracy with strict false positive control.
The AUC, as discussed previously, serves as an aggregate measure of the model's discriminative ability across all thresholds, capturing its overall capacity to distinguish between true and false matches. A higher AUC value suggests that the model is generally better at this task, making it an essential benchmark for comparing model performance on IJB-C.
By combining TAR@FAR and AUC, the IJB-C benchmark provides a comprehensive assessment of a model's performance under challenging conditions. These metrics allow for the analysis of both accuracy and robustness, providing insight into how well a model can balance security and usability in real-world applications. In particular, TAR@FAR emphasizes the model's effectiveness in maintaining low false acceptance rates without sacrificing true positive rates, a key requirement in practical deployments [17].

3.6 Embedding Aggregation: Mean and ERS

In template-based face recognition, the aggregation of embeddings plays a crucial role in generating robust representations of an individual's identity across multiple images. This section compares two embedding aggregation strategies: the simple mean method and the Enhanced Representation by Subsampling (ERS) approach. Each method has its unique advantages and limitations, which impact the final performance of the face recognition model.

3.6.1 Simple Mean Aggregation

The simplest method of embedding aggregation is to compute the arithmetic mean of embeddings obtained from multiple images of the same individual. By averaging these embeddings,

the model generates a single, unified template that captures the shared characteristics of the subject.
While straightforward and computationally efficient, the mean aggregation method is sensitive to outliers. Images that suffer from low quality, occlusions, or extreme variations in pose can distort the averaged embedding, leading to a less accurate representation of the subject's identity. This limitation often results in decreased recognition accuracy, especially when templates are constructed from a mixture of high- and low-quality images [13].

3.6.2 Enhanced Representation Strategy (ERS)

The *Embedding Recognizability Score (ERS)* strategy provides an innovative approach to improving face recognition by evaluating the recognizability of individual embeddings. Unlike conventional mean aggregation methods, ERS introduces a measure of embedding quality based on its distance from a reference centroid, representing low-quality or "Unrecognizable Images" (UI). This method does not require additional training or manual annotations, making it versatile and easily adaptable to both single-image and set-based face recognition tasks [17].

ERS Process and Definition

The ERS strategy begins by defining a reference centroid, called the *UI Centroid (UIC)*, which represents the average embedding of unrecognizable images. These images can be obtained through:

- Clustering embeddings derived from artificially degraded datasets.
- Utilizing large-scale in-the-wild face datasets, such as WIDERFace[18], to identify a cluster of embeddings corresponding to low-quality images.

The normalized mean embedding of these images serves as the UIC, denoted as f_{UI}.
For a given embedding f_i, its *Embedding Recognizability Score* is calculated as:

$$e_i = 1 - \langle f_{\text{UI}}, f_i \rangle$$

where $\langle f_{\text{UI}}, f_i \rangle$ is the cosine similarity between the embedding and the UIC. Higher ERS values indicate better recognizability, while lower values signify embeddings that are closer to the UIC and thus harder to recognize.

Experimental results demonstrate that embeddings with high ERS scores are strongly correlated with higher image quality, minimal occlusions, and frontal poses. Conversely, embeddings with low ERS values often correspond to images affected by blurriness, occlusions, or extreme variations in pose, making them difficult to recognize [17].

Figure 3.4: This collage showcases artificial degradations applied to images from the LFW [19] dataset. The first and third images are original samples, while the second is blurred with Gaussian filtering, and the fourth has added Gaussian noise.

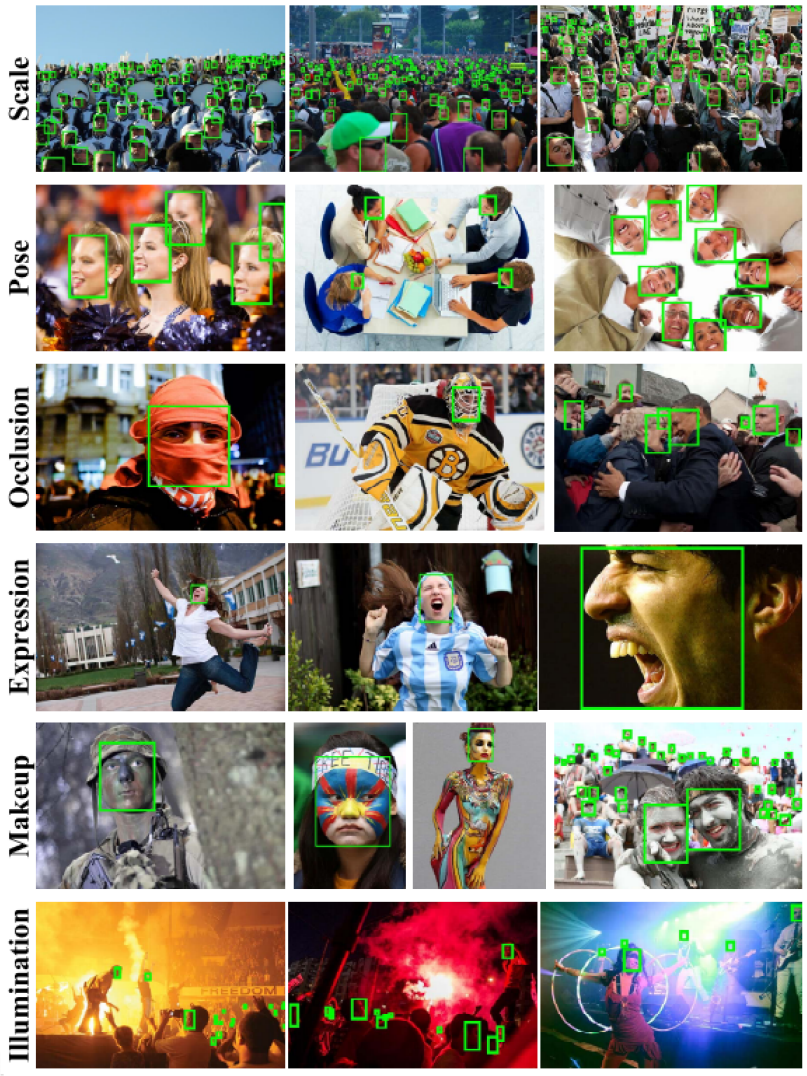


Figure 3.5: Representation of the images of the WIDERFace set. This dataset has a high degree of variability in scale, pose, occlusion, expression, appearance and illumination.

ERS in Set-Based Face Recognition

In set-based face recognition, where each set θ_i contains multiple images of the same person, ERS can be utilized during the template aggregation phase:

1. Extract embeddings f_i^l for all images in the set θ_i.

2. Compute ERS for each embedding e_i^l.

3. Perform a weighted aggregation of the embeddings using their ERS values as weights:

$$f_i = \frac{\sum_{l=1}^{|\theta_i|} w(e_i^l) f_i^l}{\sum_{l=1}^{|\theta_i|} w(e_i^l)}$$

   where $w(e_i^l)$ is a weighting function based on ERS.

This aggregation strategy ensures that high-quality embeddings contribute more to the final template representation, enhancing robustness against noisy or low-quality images [17].

Advantages of ERS

The ERS strategy offers several benefits:

- By prioritizing high-quality embeddings and penalizing low-quality ones, ERS reduces the impact of noise, occlusions, and extreme variations in pose.

- ERS can be applied to set-based recognition systems without additional training.

- By incorporating recognizability thresholds, ERS minimizes false matches and improves decision reliability, particularly in challenging scenarios.

Empirical results demonstrate that ERS significantly outperforms simple mean aggregation, especially in scenarios with substantial variations in image quality. By focusing on embedding recognizability, ERS provides a robust and scalable solution for face recognition tasks [17].

Algorithm 3 ERS-Based Template Aggregation

Load image embeddings and template IDs
Load UIC (Unrecognizable Image Centroid)
Iterate Over Templates
Identify unique template IDs
for all templates t **do**
  Retrieve embeddings for images in template t
  Normalize image embeddings and UIC
  Compute cosine similarity for each embedding:

$$\text{cosine\_similarity} \leftarrow \frac{\text{dot}(f_i, f_{UI})}{\|\|f_i\|\| \cdot \|\|f_{UI}\|\|}$$

  Compute ERS:

$$e_i \leftarrow 1 - \text{cosine\_similarity}$$

  Filter Embeddings
  Select valid embeddings:

$$\text{valid\_indices} \leftarrow \{i : e_i \geq \gamma\}$$

  Retrieve valid embeddings and ERS scores
  Aggregate Embeddings
  if no valid embeddings **then**
    Aggregate with mean:

$$\text{template\_feature} \leftarrow \text{mean}(f_i)$$

  else
    Normalize ERS scores:

$$e_i' \leftarrow \frac{e_i}{\sum e_i}$$

    Compute weighted aggregation:

$$\text{template\_feature} \leftarrow \sum (e_i' \cdot f_i)$$

  end if
  Store normalized template feature
end for
Save Results
Save all aggregated and normalized template features

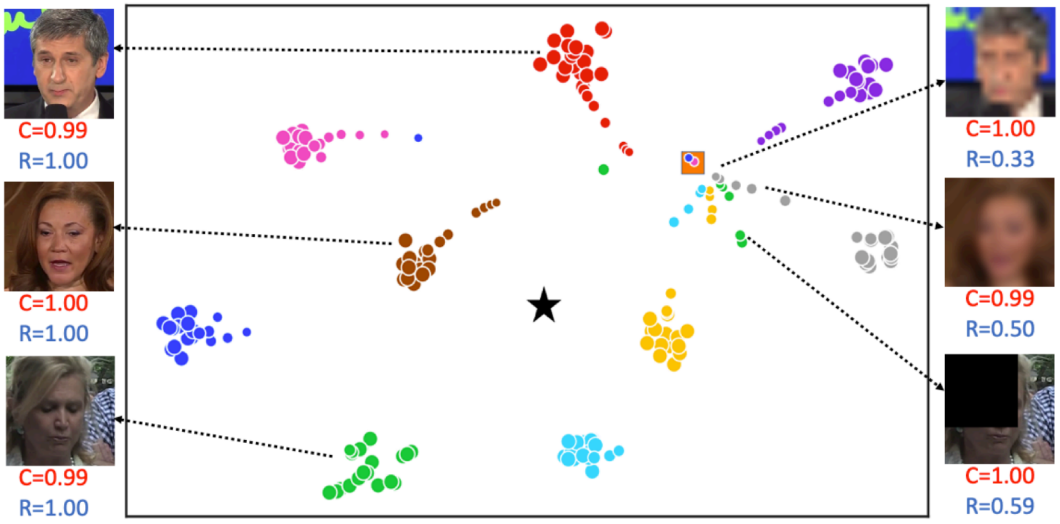


Figure 3.6: This image illustrates the application of the Enhanced Representation by Subsampling (ERS) strategy to identify high-quality embeddings within clusters of face images. Each cluster represents embeddings extracted from different identities, color-coded for clarity. The metrics displayed (C = Confidence, R = Reliability) highlight the robustness of the ERS method in selecting representative embeddings, even when low-quality or noisy data is present. The dotted lines connect selected embeddings to their respective subjects, demonstrating the filtering process for creating robust identity templates.

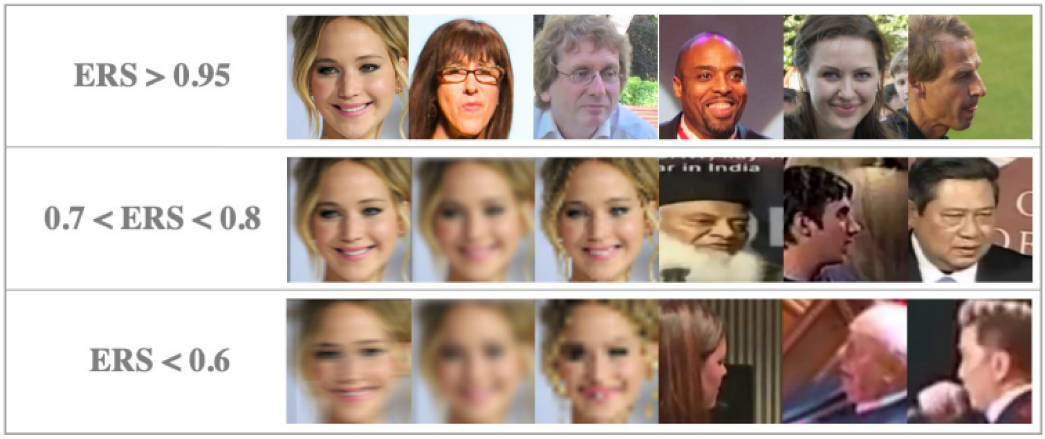


Figure 3.7: This figure depicts how the ERS strategy evaluates the quality of embeddings based on the ERS score. The rows represent different thresholds of ERS scores:
- ERS > 0.95: High-quality embeddings with minimal noise and clear identity representation.
- 0.7 < ERS < 0.8: Moderate-quality embeddings with slightly degraded clarity or consistency.
- ERS < 0.6: Low-quality embeddings with significant noise or distortion.
This visualization demonstrates how ERS prioritizes embeddings with higher scores for aggregation, effectively discarding noisy or unreliable embeddings to improve the robustness of the final template.

3.7 Implementation of IJB-C Benchmark Testing

The test was implemented using several custom MATLAB scripts designed to handle various stages of the IJB-C protocol, from embedding generation to template aggregation, pair matching, and performance evaluation. The images required to perform the test, along with the text files necessary for generating templates and conducting comparisons, were provided by Matteo Grandin, the corporate representative.

Initially, I generated embeddings for each face in the dataset using trained neural network model, ensuring consistency in image sizing and format to facilitate uniform embedding extraction. Each image from the dataset was processed, creating a feature matrix that captured critical facial characteristics for each image.

Following this, the individual embeddings were aggregated at the template level to represent each unique identity. Aggregation methods, as discussed before, had a significant impact on the model's performance in identifying face pairs.

Once template-level features were established, cosine similarity scores between each pair of templates were computed. This comparison relied on labels specifying pairs to be evaluated, allowing for precise calculation of similarities based on the dot product between template feature vectors. The similarity scores for each template pair were stored for performance assessment, forming a core component of the ROC and AUC calculations.

Through this multi-step approach, the model's ability to perform on a widely recognized benchmark was rigorously assessed, providing clear insights into the influence of feature aggregation strategies on recognition accuracy.

Algorithm 4 Test Pipeline

Step 1: Compute Facial Embeddings
Load image paths and template IDs
Import trained model
for all images i **do**
    Preprocess image: ensure RGB, resize to 112×112
    Compute embedding using `predict`
    Store embedding in matrix
end for
Save embeddings to file
Step 2: Compute Template-Level Features
Load facial embeddings and template data
Identify unique template IDs
for all templates t **do**
    Retrieve embeddings for images in template t
    Aggregate embeddings with mean or ERS startegy
    Normalize template feature
end for
Save normalized template features
Step 3: Compute Cosine Similarities
Load template pair labels and normalized template features
for all template pairs (t_1, t_2) **do**
    Compute cosine similarity:

$$\text{similarity} \leftarrow \frac{\text{dot}(t_1, t_2)}{\|\|t_1\|\| \cdot \|\|t_2\|\|}$$

end for
Save cosine similarities to file
Step 4: Compute ROC Curve and TAR@FAR
Load cosine similarities and true labels
Sort similarities and labels by score
Initialize TP, FP, FPR, and TPR
for all thresholds **do**
    Update TP and FP counts
    Compute $FPR \leftarrow FP/N$, $TPR \leftarrow TP/P$
end for
Remove duplicate FPR values and calculate AUC:

$$\text{AUC} \leftarrow \int \text{TPR}\, d\text{FPR}$$

Plot ROC curve
for all target FAR values **do**
    Interpolate TAR@FAR using:

$$\text{TAR@FAR} \leftarrow \text{interp1(FPR, TPR, target FAR)}$$

end for
return ROC curve, AUC, and TAR@FAR values

4

Evaluation and Analysis of Face Recognition Models

In this final chapter, I present the results obtained using the IJB-C test. Initially, I analyze the performance using simple mean aggregation for embedding vectors. Subsequently, I incorporate the ERS strategy to evaluate its impact on the results. Finally, I compare these outcomes with the state-of-the-art performance and provide a detailed discussion on the findings in the context of this comparison.

4.1 Baseline Results with Mean Embedding Aggregation

The models evaluated include ArcFace, MagFace, and the novel loss function, all trained on ResNet-100 architectures for 30 epochs. Additionally, a ResNet-50 model trained with the novel loss function was included in the evaluation to explore its performance on a lighter architecture. Performance was measured using TAR@FAR metrics across various thresholds and visualized through ROC curves.

The results demonstrate that the best model overall remains the one trained with MagFace loss on the ResNet-100 architecture. However, the ResNet-50 model trained with the novel loss function exhibited competitive performance, particularly at lower FAR thresholds, showcasing its potential for scenarios where computational efficiency is a priority.

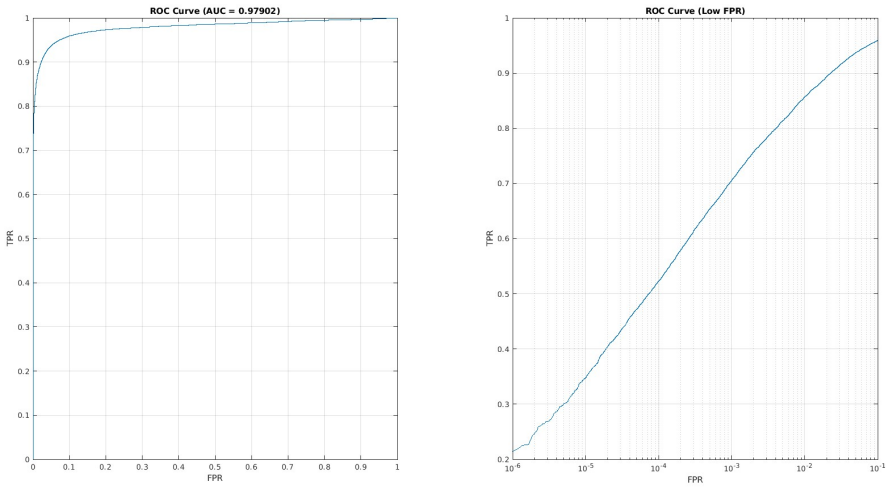


Figure 4.1: ROC curves showcasing the performance of ArcFace.

TAR@FAR	Value %
1.0×10^{-6}	21.43
1.0×10^{-5}	34.81
1.0×10^{-4}	52.23
1.0×10^{-3}	70.56
1.0×10^{-2}	85.64
1.0×10^{-1}	95.89

Table 4.1: TAR@FAR performance metrics for ArcFace on the IJB-C dataset.

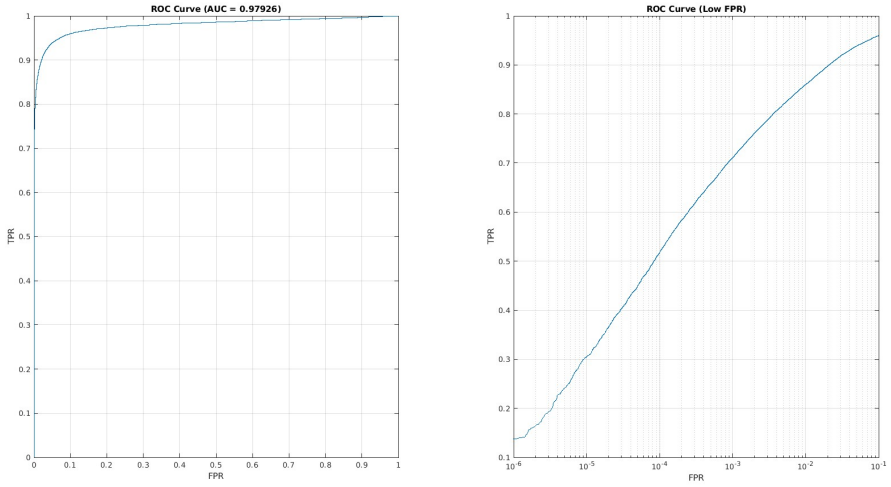


Figure 4.2: ROC curves showcasing the performance of MagFace.

TAR@FAR	Value %
1.0×10^{-6}	13.75
1.0×10^{-5}	30.52
1.0×10^{-4}	51.73
1.0×10^{-3}	71.04
1.0×10^{-2}	86.01
1.0×10^{-1}	95.97

Table 4.2: TAR@FAR performance metrics for MagFace on the IJB-C dataset.

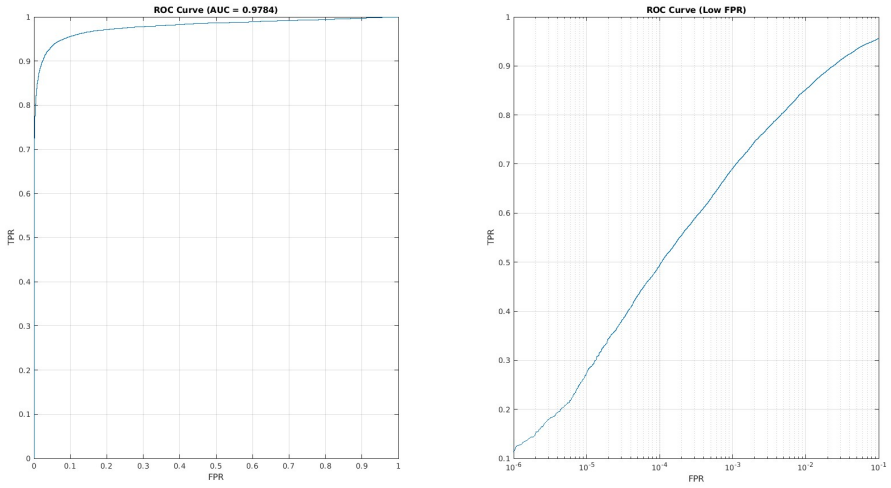


Figure 4.3: ROC curves showcasing the performance of the novel Loss function.

TAR@FAR	Value %
1.0×10^{-6}	21.01
1.0×10^{-5}	34.85
1.0×10^{-4}	51.76
1.0×10^{-3}	69.94
1.0×10^{-2}	85.32
1.0×10^{-1}	95.54

Table 4.3: TAR@FAR performance metrics for the novel Loss function on the IJB-C dataset.

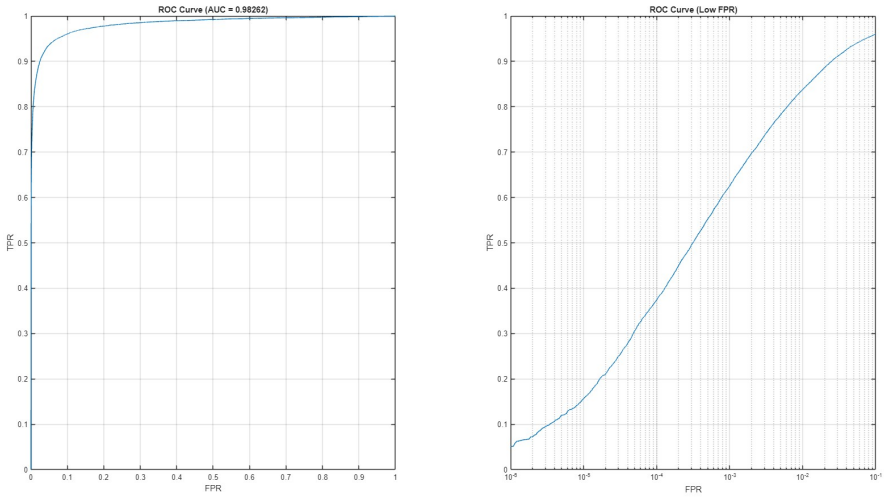


Figure 4.4: ROC curves showcasing the performance of the novel Loss function with ResNet-50 architecture

TAR@FAR	Value %
1.0×10^{-6}	5.05
1.0×10^{-5}	15.63
1.0×10^{-4}	37.47
1.0×10^{-3}	62.59
1.0×10^{-2}	83.79
1.0×10^{-1}	96.05

Table 4.4: TAR@FAR performance metrics for the novel Loss function on the ResNet-50 architecture evaluated on the IJB-C dataset.

4.2 Baseline Results with ERS Embedding Aggregation

The models evaluated include ArcFace, MagFace, and the novel loss function, trained on ResNet-100 architectures for 30 epochs. Additionally, a ResNet-50 model trained with the

novel loss function was included in the evaluation to assess its performance in comparison. For the testing phase on the IJBC dataset, embeddings were aggregated using the Enhanced Representation Strategy (ERS), designed to optimize feature representation. Following the recommendations of the reference paper [17], the gamma value was set to 0.60, as this configuration consistently achieved the best performance. Performance was assessed using TAR@FAR metrics across various thresholds and visualized through ROC curves. This section highlights the improvements achieved by ERS compared to traditional aggregation methods, demonstrating its effectiveness. While the model trained with MagFace loss consistently outperformed others across most metrics, the ResNet-50 model trained with the novel loss function showed competitive results, particularly in the lower FAR ranges. These results underline the potential of the novel loss function when combined with a lightweight architecture such as ResNet-50, offering a viable alternative for applications requiring a balance between efficiency and performance.

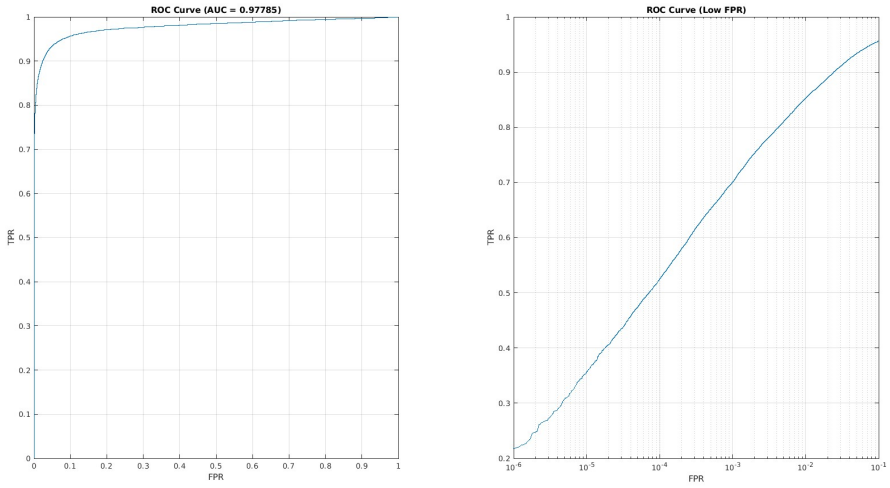


Figure 4.5: ROC curves showcasing the performance of ArcFace with ERS.

TAR@FAR	**Value (%)**
1.0×10^{-6}	22.67%
1.0×10^{-5}	38.12%
1.0×10^{-4}	54.68%
1.0×10^{-3}	71.64%
1.0×10^{-2}	85.77%
1.0×10^{-1}	95.60%

Table 4.5: TAR@FAR performance metrics for ArcFace on the IJB-C dataset with ERS.

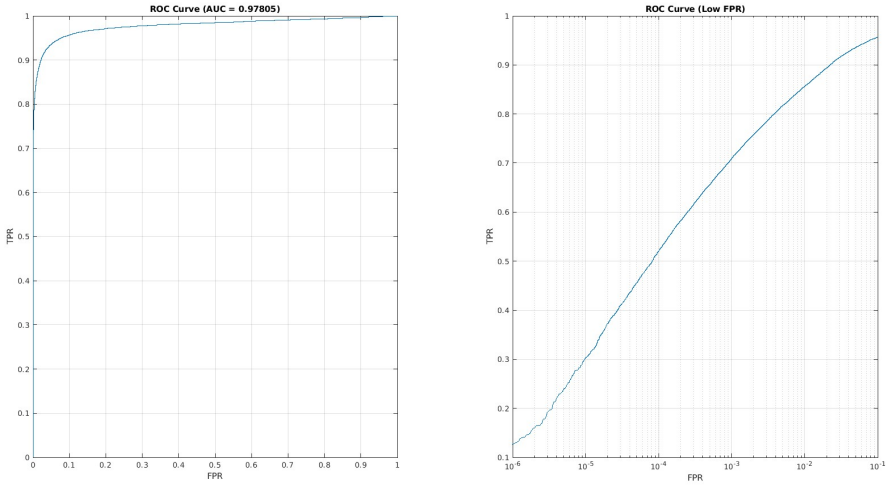


Figure 4.6: ROC curves showcasing the performance of MagFace with ERS.

TAR@FAR	Value (%)
1.0×10^{-6}	16.08%
1.0×10^{-5}	35.84%
1.0×10^{-4}	55.50%
1.0×10^{-3}	72.48%
1.0×10^{-2}	86.21%
1.0×10^{-1}	95.79%

Table 4.6: TAR@FAR performance metrics for MagFace on the IJB-C dataset with ERS.

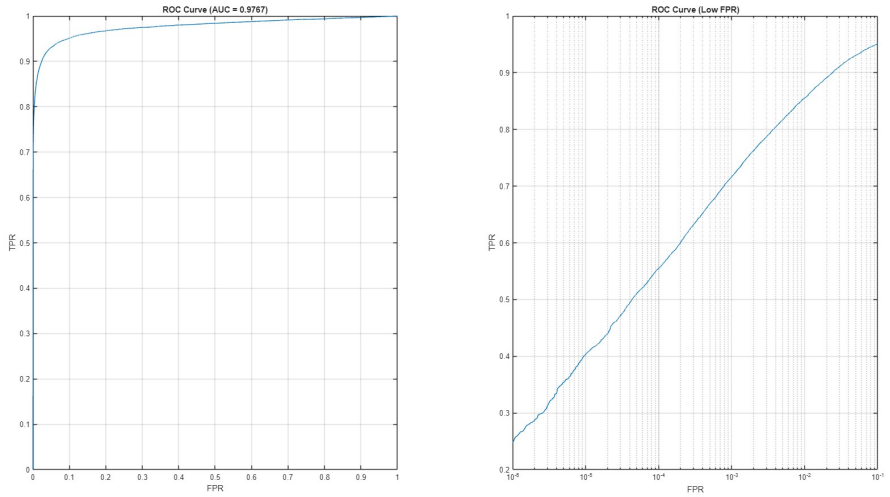


Figure 4.7: ROC curves showcasing the performance of the novel Loss function with ERS.

TAR@FAR	Value (%)
1.0×10^{-6}	24.65%
1.0×10^{-5}	40.28%
1.0×10^{-4}	55.48%
1.0×10^{-3}	71.64%
1.0×10^{-2}	85.54%
1.0×10^{-1}	95.12%

Table 4.7: TAR@FAR performance metrics for the novel Loss function on the IJB-C dataset with ERS.

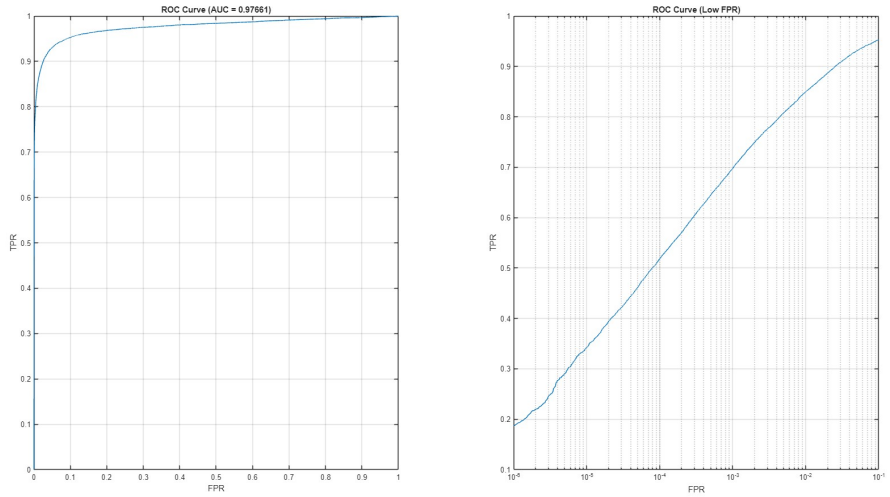


Figure 4.8: ROC curves showcasing the performance of the novel Loss function with ERS and ResNet-50 architecture.

TAR@FAR	Value %
1.0×10^{-6}	7.82
1.0×10^{-5}	24.19
1.0×10^{-4}	45.82
1.0×10^{-3}	67.36
1.0×10^{-2}	85.24
1.0×10^{-1}	96.22

Table 4.8: TAR@FAR performance metrics for the novel Loss function with ERS strategy on the IJB-C dataset.

4.3 Comparison with State-of-the-Art Models

Model	Train Data	TAR@FAR=1.0×10^{-4}
ArcFace ($m = 0.50$)	MS1MV2	96.03%
CurricularFace	MS1MV2	96.1%
MagFace	MS1MV2	95.97%
AdaFace ($m = 0.4$)	MS1MV3	97.09%

Table 4.9: Performance comparison of state-of-the-art models on the IJB-C dataset, reporting TAR@FAR metrics [6].

As shown in this table, the performance of the state-of-the-art models described in the referenced paper significantly surpasses that of the models developed in this research.

4.4 Analysis of Results and Potential Sources of Systematic Errors

The results obtained during this research highlight the presence of systematic errors in the evaluated models. Based on a detailed analysis, several potential causes for these discrepancies have been identified:

1. **Pre-trained Model Import Issues**: One of the most likely sources of error stems from the import of pre-trained models from Python to MATLAB. As detailed in the relevant section of this thesis, the conversion process required modifications to the model architecture, particularly replacing layers such as the flatten operation with alternatives like global average pooling. These changes, while necessary to ensure compatibility, may have altered the feature representation and contributed to the observed performance degradation.

2. **Test Dataset Limitations**: The IJB-C dataset, although a widely used benchmark for face recognition, is not immune to potential imperfections that can affect evaluation outcomes. It is important to note that the dataset was developed and provided by an external organization, and as such, it may inherit some issues typical of large-scale, externally curated datasets. These issues might include mislabeled samples and unbalanced identity class distributions. Such anomalies can introduce noise into the evaluation process, potentially leading to biased or less reliable assessments of a model's performance. For example, mislabeled samples might result in incorrect predictions being unfairly penalized, while uneven distributions of identities can lead to models being optimized for majority classes

at the expense of minority ones. These factors highlight the importance of careful dataset scrutiny when interpreting benchmarking results.

3. **Complexity of the Novel Loss Function**: The newly developed loss function, designed to integrate the strengths of MagFace and CurricularFace, is computationally more complex and involves multiple hyperparameters, such as the adaptive margin, angular penalties, max and min value for magnitude and margin, the variation of the t parameter and regularization terms. While initial experiments showed promising results, the fine-tuning process for these parameters requires more extensive study to achieve the optimal balance. This complexity might have hindered the model's ability to generalize effectively during testing.

4. **Training Conditions and Computational Constraints**: The training process was carried out on computational resources with specific constraints, such as limited time availability on high-performance clusters. These restrictions may have prevented sufficient exploration of hyperparameter spaces, resulting in suboptimal training of the models. Moreover, the fixed architecture (e.g., ResNet-100) may not have fully exploited the capabilities of the proposed loss function.

4.5 Conclusions

The systematic errors identified in this research emphasize the importance of optimizing both the training pipeline and the evaluation process. While the novel loss function demonstrates potential for improving face recognition performance, further refinements are necessary to mitigate issues arising from computational complexity, parameter sensitivity, and dataset inconsistencies. Future work will involve addressing these limitations to achieve a more robust and generalizable model.

4.6 Future Works

This thesis has explored the development and evaluation of novel loss functions for face recognition models, leveraging pre-trained architectures and fine-tuning strategies. However, there are several promising directions for future work to build upon the foundations laid in this study. A critical next step would be to train a neural network from scratch rather than relying on pre-trained models imported into MATLAB. This approach would provide deeper insights into how the model architecture and loss function interact during training, without the biases introduced by pre-trained weights. Training from scratch also offers the opportunity to fully tailor the model

to the specific characteristics of the dataset, potentially unlocking better performance on challenging benchmarks like IJB-C.
Another area for improvement lies in the systematic exploration of hyperparameter configurations.
Furthermore, testing the proposed loss function on a wider range of datasets, including those with diverse demographic and environmental conditions, would provide a more comprehensive understanding of its generalization capabilities. This could also include exploring synthetic or augmented data to simulate edge cases that are underrepresented in current benchmarks.
Finally, integrating the novel loss function into state-of-the-art architectures and frameworks beyond MATLAB, such as PyTorch or TensorFlow, could facilitate broader adoption and allow for comparisons with the latest advancements in the field.

Bibliography

[1] W. Ali, W. Tian, S. U. Din, D. Iradukunda, and A. A. Khan, "Classical and modern face recognition approaches: A complete review," *Multimedia Tools and Applications*, vol. 80, no. 3, pp. 4825–4880, 2021, issn: 1573-7721. doi: 10.1007/s11042-020-09850-1. [Online]. Available: https://doi.org/10.1007/s11042-020-09850-1.

[2] Y. Kortli, M. Jridi, A. A. Falou, and M. Atri, "Face recognition systems: A survey," *Sensors*, vol. 20, no. 2, p. 342, 2020, Special Issue on Biometric Systems. doi: 10.3390/s20020342. [Online]. Available: https://doi.org/10.3390/s20020342.

[3] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, Jun. 2015, pp. 815–823. doi: 10.1109/cvpr.2015.7298682. [Online]. Available: http://dx.doi.org/10.1109/CVPR.2015.7298682.

[4] J. Deng, J. Guo, J. Yang, N. Xue, I. Kotsia, and S. Zafeiriou, "Arcface: Additive angular margin loss for deep face recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 10, pp. 5962–5979, Oct. 2022, issn: 1939-3539. doi: 10.1109/tpami.2021.3087709. [Online]. Available: http://dx.doi.org/10.1109/TPAMI.2021.3087709.

[5] Q. Meng, S. Zhao, Z. Huang, and F. Zhou, *Magface: A universal representation for face recognition and quality assessment*, 2021. arXiv: 2103.06627 [cs.CV]. [Online]. Available: https://arxiv.org/abs/2103.06627.

[6] M. Kim, A. K. Jain, and X. Liu, *Adaface: Quality adaptive margin for face recognition*, 2023. arXiv: 2204.00964 [cs.CV]. [Online]. Available: https://arxiv.org/abs/2204.00964.

[7] Y. Huang, Y. Wang, Y. Tai, *et al.*, *Curricularface: Adaptive curriculum learning loss for deep face recognition*, 2020. arXiv: 2004.00288 [cs.CV]. [Online]. Available: https://arxiv.org/abs/2004.00288.

[8] B. Maze, J. C. Adams, J. A. Duncan, *et al.*, "Iarpa janus benchmark - c: Face dataset and protocol," *2018 International Conference on Biometrics (ICB)*, pp. 158–165, 2018. [Online]. Available: https://api.semanticscholar.org/CorpusID:28375094.

[9] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "Deepface: Closing the gap to human-level performance in face verification," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1701–1708. doi: 10.1109/CVPR.2014.220.

[10] Y. Wen, K. Zhang, Z. Li, and Y. Qiao, "A discriminative feature learning approach for deep face recognition," in *Computer Vision – ECCV 2016*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds., ser. Lecture Notes in Computer Science, vol. 9911, Springer, Cham, 2016, pp. 499–515. doi: 10.1007/978-3-319-46478-7_31. [Online]. Available: https://doi.org/10.1007/978-3-319-46478-7_31.

[11] W. Liu, Y. Wen, Z. Yu, and M. Yang, *Large-margin softmax loss for convolutional neural networks*, 2017. arXiv: 1612.02295 [stat.ML]. [Online]. Available: https://arxiv.org/abs/1612.02295.

[12] W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, and L. Song, *Sphereface: Deep hypersphere embedding for face recognition*, 2018. arXiv: 1704.08063 [cs.CV]. [Online]. Available: https://arxiv.org/abs/1704.08063.

[13] Y. Guo, L. Zhang, Y. Hu, X. He, and J. Gao, *Ms-celeb-1m: A dataset and benchmark for large-scale face recognition*, 2016. arXiv: 1607.08221 [cs.CV]. [Online]. Available: https://arxiv.org/abs/1607.08221.

[14] Z. Zhu, G. Huang, J. Deng, *et al.*, *Webface260m: A benchmark unveiling the power of million-scale deep face recognition*, 2021. arXiv: 2103.04098 [cs.CV]. [Online]. Available: https://arxiv.org/abs/2103.04098.

[15] T. Fawcett, "An introduction to roc analysis," *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861–874, 2006, ROC Analysis in Pattern Recognition, issn: 0167-8655. doi: https://doi.org/10.1016/j.patrec.2005.10.010. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S016786550500303X.

[16] J. A. Hanley and B. J. McNeil, "The meaning and use of the area under a receiver operating characteristic (roc) curve," *Radiology*, vol. 143, no. 1, pp. 29–36, 1982. doi: 10.1148/radiology.143.1.7063747.

[17] S. Deng, Y. Xiong, M. Wang, W. Xia, and S. Soatto, *Harnessing unrecognizable faces for improving face recognition*, 2021. arXiv: 2106.04112 [cs.CV]. [Online]. Available: https://arxiv.org/abs/2106.04112.

[18] S. Yang, P. Luo, C. C. Loy, and X. Tang, *Wider face: A face detection benchmark*, 2015. arXiv: 1511.06523 [cs.CV]. [Online]. Available: https://arxiv.org/abs/1511.06523.

[19] 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," University of Massachusetts, Amherst, Technical Report 07-49, 2007.

Acknowledgments

This thesis greatly benefited from the invaluable contributions of Matteo Grandin, whose expertise in data management and thoughtful guidance on the implementation of the IJB-C test played a fundamental role in shaping this work.
Special thanks are also due to Professor Nanni Loris for his steadfast support and mentorship throughout this journey. His deep knowledge, constructive feedback, and thoughtful suggestions provided essential direction and inspiration at every stage of the project.