CONEFACE: APPROXIMATE PAIRWISE LOSS FOR FACE RECOGNITION

*Zijun Zhuang, Hongtao Lu**

Department of Computer Science and Engineering, Shanghai Jiao Tong University, China
zhuangzijun@sjtu.edu.cn, htlu@sjtu.edu.cn

ABSTRACT

The discriminability of learned face features is the key to a successful face recognition algorithm under the open-set protocol. Recent research exploits well-designed loss functions to penalize the angles between the deep features and their class centers for reaching the purpose of minimizing the intra-class variance and achieves a significant increase in recognition accuracy. In this paper, we proposed an approximate pairwise loss (APL) to encourage inter-class separability as well as intra-class compactness. More specifically, we use cones to approximate the location of hard examples in the feature space which replaces the thorny hard-mining step, and the APL is obtained by calculating the angular distance between the features of training samples and the cone and we named our method ConeFace. Moreover, ConeFace can be easily used together with other SOTA methods to improve the performance with negligible computational overhead. Extensive experiments on Labeled Face in the Wild (LFW), Celebrities in Frontal-Profile in the Wild (CFP), AgeDB-30, and MegaFace datasets show the effectiveness of the proposed ConeFace.

Index Terms— Face Recognition, Metric Learning, Angular Margin Loss

1. INTRODUCTION

Face recognition is the most commonly used biometric authentication in daily life. Since GassianFace [1] first surpasses the human-level on LFW [2], face recognition has become one of the most promising topics in computer vision and artificial intelligence area. In a typical real-world scenario, face recognition is an open-set problem, meaning the identity that appears in the testing phase is never included in the training set. The open-set protocol makes face recognition much more challenging and different from general classification problems, and the discriminability of the feature representation learned from the training set is the key to achieving a model with strong generalization ability. With the development of face recognition technology and the growing amount of face datasets, there are currently two main lines, metric learning-based methods and classification based methods for face recognition. Metric learning-based methods, *e.g.*, Contrastive Loss [3], is exploited to obtain the high inter-class separability. And classification-based method, *e.g.*, CosFace [4] and ArcFace [5], improve the intra-class compactness with the help of the angular margin. However, there are some drawbacks to them. For metric learning-based methods, the combination of pairs is explosive when the training dataset gets larger; what's more, hard or semi-hard example mining is also a challenging problem even in a relatively small dataset which is necessary for the metric learning method to be efficiently trained. The classification-based method does not consider the relation between image pairs, which is very important in the testing phase. In this paper, we proposed a novel loss function called Approximate Pairwise Loss(APL), which considers both inter-class separability and intra-class compactness. It is hard to calculate and optimize the distance between each training pair in the classification-based method, which only focuses on the relationship between the training samples and the class centers. Meanwhile, it is difficult to consider the connections among the classes in the metric learning-based method. The proposed ConeFace can take advantage of both approaches. We use cones to approximate the distribution boundary of each class in the normalized feature space. Thus we utilize the cone to obtain pairwise information of the training samples, which should be a hard task for the classification-based method. Our contributions can be summarized as follows:

1. We proposed a loss function called Approximate Pairwise Loss that can use both classification information and pairwise information; thereby, inter-class separability and intra-class compactness can be optimized simultaneously.
2. The proposed ConeFace does not require additional computing resources and can easily be used with other classification-based methods.
3. We do extensive experiments and the method achieves a good performance on the LFW [2], CFP [6], AgeDB-30 [7] and MegaFace [8] datasets.

2. METHOD

Our proposed ConeFace is based on the angular softmax classification network [9]. However, unlike other classification-

*Corresponding author, also with MOE Key Lab of Artificial Intelligence, AI Institute, Shanghai Jiao Tong University, China

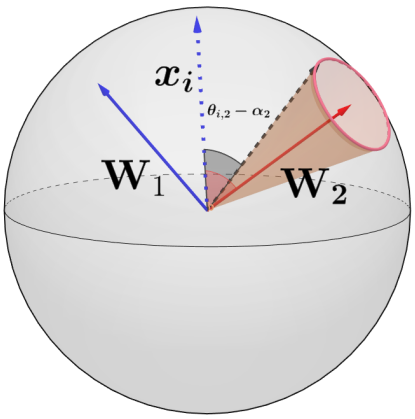


Fig. 1. An illustration of the idea of cone boundary and APL. W_1 and W_2 are the class centers of two identities. x_i is the embedding feature of an image belongs to identity 1. The red cone is the approximate boundary of identity 2 with an angle of α_2. The angle between x_i to the identity 2 is $\theta_{i,2} - \alpha_2$ when calculating APL.

based methods, ConeFace can also enlarge the inter-class separability through our carefully designed approximate pairwise loss (APL). We call this method ConeFace because the APL is obtained by calculating the angular distance between the feature of training samples and the approximate class boundary, and the approximate boundary of each class is a cone in our method.

Constraining the distance between samples that do not belong to the same category in the feature space and requiring the distance larger than a certain margin is the most intuitive and effective method to improve inter-class separability. However, as the dataset grows, the number of sample pairs grows in $\mathcal{O}(n^2)$, which makes it even harder to do hard mining from the training set. So we need to use some approximate methods to simplify these problems without too much degradation of the algorithm's performance.

Suppose the number of identities of the training set is N, the batch size is M, $x_i \in \mathbb{R}^d$ is the i-th feature embedding in the batch with y_i the corresponding label, and $\boldsymbol{W} = \{\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_N\}$ is the weight of the last fully connected layer, where $\boldsymbol{w}_i$ can be explained as the class center of the i-th identity. The angular softmax loss [9] can be presented as follows:

$$L_{\text{A-softmax}} = \frac{1}{M}\sum_{i=1}^{M} -\log \frac{e^{\cos\theta_{i,y_i}}}{\sum_{j=1}^{N} e^{\cos\theta_{i,j}}} \tag{1}$$

where $\theta_{i,j}$ is the angle between x_i and $\boldsymbol{w}_j$. the final classification result is only related to $\theta_{i,j}$.

The hard examples are usually those features with a large θ_{i,y_i}. Given an identity k as an example, the features vectors of its samples should be wrapped in a high-dimensional cone C_k. The axis of cone C_k is $\boldsymbol{w}_k$, and the generatrix is the feature vector x_l where $l = \arg\max_i \{\theta_{i,y_i}\}_{y_i=k}$, *i.e.*, the feature vector most far away from the class center $\boldsymbol{w}_k$. This high-dimensional cone is a boundary of the identity k. Figure. 1 gives a direct observation of the relationship between the class center, features, and cone boundary. However, it is still tough to get this boundary since we need to infer all the samples in the training set and then calculate the angles of all the training samples to their class center to update the boundary of all identities. Therefore, we propose a method to approximate the boundary by iterating on each batch during the training, just like training the general neural networks.

For simplicity, we use the angle between the generatrix and axis to represent cone boundary. We use a vector $\boldsymbol{\alpha} \in \mathbb{R}^N$ to record the boundary and initialize it as $\boldsymbol{\alpha}^\mathsf{T} = \{0, 0, \ldots, 0\}$, which means the initial angle between the generatrix and axis of high-dimensional cone is $0°$. Then, in a batch, samples of some identities will be trained. At this time, we record the angle between these samples and their respective class centers. This is very easy to do because the output logits are naturally the cosine value of the angle between the sample and each identity in normalized softmax. After we get the angle between the training samples in this batch and their respective class centers, $\theta_{i,j}$ for example, where $y_i = j$. We can use θ_{i,y_i} to update the boundary $\alpha_{y_i}^t$ at step $t+1$. The update rule is as follow:

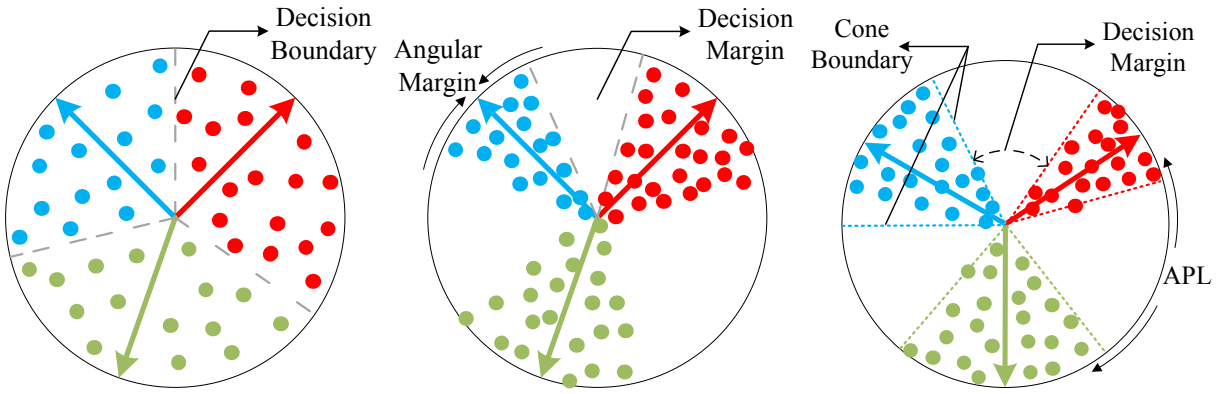
$$\alpha_{y_i}^{t+1} = \begin{cases} \theta_{i,y_i} & \text{If } \theta_{i,y_i} \geq \alpha_{y_i}^t \\ \dfrac{1}{2}(\theta_{i,y_i} + \alpha_{y_i}^t) & \text{If } \theta_{i,y_i} < \alpha_{y_i}^t \end{cases}, \tag{2}$$

$\theta_{i,y_i} \geq \alpha_{y_i}^t$ means we find a new sample that can represent the boundary more accurate, so we will update the new boundary $\alpha_{y_i}^{t+1} = \theta_{i,y_i}$. However, when $\theta_{i,y_i} < \alpha_{y_i}^t$, the approach would be a bit different. We know that as the training progresses, the intra-class compactness will increase, so the boundary $\boldsymbol{\alpha}$ will gradually become smaller, and the previous boundary may be out of date. So even if θ_{i,y_i} is smaller than $\alpha_{y_i}^t$, we will still update the boundary. We will take the average of the new angle and the previous boundary as the new boundary angle. By applying the update rule, we can get the approximate boundary iteratively.

Based on the above assumptions and approximations, we can assert that the hard examples in a certain identity are mostly distributed near the high-dimensional cone that wraps the identity. Thus, calculating the distance of other samples to this cone can approximate the pairwise relationship between the other samples and the hard examples in this identity. The approximate pairwise loss (APL) thus defined as follows:

$$L_{\text{APL}} = \frac{1}{M}\sum_{i=1}^{M} -\log \frac{e^{s\cos(\theta_{i,y_i})}}{e^{s\cos(\theta_{i,y_i})} + \sum_{j=1,j\neq y_i}^{N} e^{s\phi_{i,j}}}, \tag{3}$$

where $\phi_{i,j} = \cos(\theta_{i,j} - k\alpha_j)$. Clearly, $\phi_{i,j}$ is the cosine distance from the feature x_i to the j-th identity cone when $k = 1$,



(a) Normalized softmax loss (b) ArcFace [5] (c) ConeFace

Fig. 2. An illustration of embedding features trained by different loss functions, arrows in color indicate the class centers of different identities. (a) The features learned by Normalized softmax loss lie near the decision boundaries without margin between two identities. (b) ArcFace [5] enforce the features to be close to their class centers which improve the intra-class compactness and generate decision margin between two identities. (c) Our proposed ConeFace also generates decision margin between two identities. The dashed lines in color is the cone boundaries of the corresponding class centers. The approximate pairwise loss will pull the features from different identities apart and increase the inter-class separability.

and k determines the weight of cone loss. s is a hyperparameter introduced by CosFace [4] that can make the training more stable. We can see the difference of our proposed ConeFace with other method and our unique advantage of improving the inter-class separability in Figure. 2.

2.1. Combined with other Methods

ConeFace can be easily combined with other large angular margin methods such as CosFace and ArcFace. Here are two examples of the combined loss functions:

ConeFace + CosFace

$$L_{\text{cone+cos}} = \frac{1}{M}\sum_{i=1}^{M} -\log \frac{e^{s(\cos(\theta_{i,y_i})-d)}}{e^{s(\cos(\theta_{i,y_i})-d)} + \sum_{j=1,j\neq y_i}^{N} e^{s\phi_{i,j}}}, \tag{4}$$

ConeFace + ArcFace

$$L_{\text{cone+arc}} = \frac{1}{M}\sum_{i=1}^{M} -\log \frac{e^{s(\cos(\theta_{i,y_i}+d))}}{e^{s(\cos(\theta_{i,y_i}+d))} + \sum_{j=1,j\neq y_i}^{N} e^{s\phi_{i,j}}}, \tag{5}$$

3. EXPERIMENTS

3.1. Implementation Details

Preprocessing. We use the preprocessing method commonly used in face recognition task [4, 5]. All the images in the training set and test set are aligned by MTCNN [10] through the five-point face landmarks. Then the images are cropped into 112×112. Each pixel in images ($112 \times 112 \times 3$) is normalized by subtracting 127.5 then dividing by 128 so that the pixel can be represented in the interval $(-1, 1)$.

Network Setup. Pytorch [11] is used as the framework to implement ConeFace. We use ResNet [12] with 50 layers as the network architecture. We also use the BN [13]-Dropout [14]-FC-BN after the last convolutional layer, L inputs, IR blocks and SE module [15] which are all introduced in ArcFace [5].

Training. We use the CASIA-WebFace [16] and MS-Celeb-1M [17] datasets to train the model. The CASIA-WebFace dataset has about 10 thousand individuals and 0.5 million face images in total. The MS-Celeb-1M dataset contains about 10 million images from 100 thousand identities. Learning rate is 0.1 and drops to one tenth at 10, 16 epoch. Training is stopped at 18 epoch. k is set to be 0.3.

Testing. We test our proposed ConeFace on four widely-used face recognition benchmarks: LFW [2], CFP [6], AgeDB-30 [7] and MegaFace challenge [8, 18]. We carry out the testing under two protocols. Under the small protocol, CASIA-WebFace is used as the training set, and under the large protocol, MS-Celeb-1M is used. LFW, CFP-FP, AgeDB-30 are tested under the small protocol, and MegaFace is tested under both small and large protocols.

3.2. Experiments on LFW, CFP-FP and AgeDB-30

LFW [2] dataset contains $13,233$ images from 5749 different people. CFP [6] dataset includes 7000 face images from 500 different identities. Each person has 10 frontal images and 4 profile images . CFP-FP means the face verification is between frontal and profile. AgeDB-30 [7] includes 16,488 images from 568 people with accurate age label.

Method	LFW	CFP-FP	AgeDB-30
Softmax	98.01	92.93	90.93
ConeFace w/o warmup	97.68	90.39	88.48
ConeFace w/ warmup	**98.57**	**93.11**	**91.42**

Table 1. The face verification accuracy(%) comparison between ConeFace and Softmax. ConeFace with warmup will train the model without the APL in first two epochs as the warmup. The performance of ConeFace will surpass softmax loss after introduced the warmup mechanism.

Method	LFW	CFP-FP	AgeDB-30
Softmax	98.01	92.93	90.93
Triplet [19]	98.98	91.90	89.98
SphereFace [9]	99.11	94.38	91.70
CosFace [4]	99.51	95.44	94.56
ArcFace [5]	99.53	95.56	95.51
ConeFace	98.57	93.11	91.42
ConeFace + ArcFace	**99.62**	**95.57**	**95.84**

Table 2. The face verification accuracy(%) of different method on LFW, CFP-FP and AgeDB-30 datasets. ConeFace + ArcFace means we use the combined loss function of ConeFace and ArcFace as eq. 5 in Sec. 2.1.

We Need Warm-Up : In our early experiment, we found an interesting phenomenon that the ConeFace can not converge to an ideal minimum if we train it from scratch. The testing accuracy on LFW and other datasets are lower than the method that only uses the softmax loss. So we think the ConeFace needs some warm-up. The warm-up here is different from the commonly used method [12] that uses a small learning rate at a very early stage and then uses the standard learning rate so that the algorithm can be more stable and converge to a more optimized point. We use the same idea to reduce the algorithm's instability at the early stage since the APL may be useless or even harmful to the network if the model is not representative enough. We manually set the loss produced by APL to be zero, which means we only use the softmax loss at the first two training epochs. The comparison is demonstrated in Table. 1, and we can found there is an excellent improvement in the performance of our method with the help of the warm-up. In all the experiments, we do the warm-up step if not explicitly stated.

Comparison with State-of-the-Arts: Table. 2 shows the experimental results on LFW, CFP-FP, and AgeDB-30 datasets compared with the State-of-the-Arts methods. Our proposed ConeFace can be easily combined with other methods, and the experiments proved that ConeFace does improve the performance of the algorithm with which it is used together. When we use ConeFace together with ArcFace, it outperforms all the SOTA methods on all three datasets.

Method	Protocol	Id (%)	Ver (%)
FaceNet [19]	Large	70.50	86.47
ArcFace [5]	Large	81.03	**96.98**
CosFace [4]	Large	**82.72**	96.65
RegularFace [20]	Large	75.61	91.13
L-Softmax [21]	Small	67.13	80.42
ArcFace [5]	Small	**77.50**	**92.34**
CosFace [4]	Small	77.11	89.88
RegularFace [20]	Small	70.23	84.07
ArcFace†	Large	79.38	95.46
ConeFace + ArcFace	Large	**80.93**	**96.55**
ArcFace†	Small	73.81	87.70
ConeFace + ArcFace	Small	**75.02**	**88.51**

Table 3. Face identification and verification evaluation of different methods on MegaFace Challenge 1. "Id" refers to the Rank-1 face identification accuracy, and "Ver" refers to the face verification TAR at 10^{-6} FAR. The ArcFace† and our method are trained under the same setting for fair comparison.

3.3. Experiments on MegaFace

We compare our method with many state-of-the-art methods, and the result can be found in Table. 3. In the face identification task, we give out the Rank-1 accuracy of these methods, and in the face verification task, we test the TAR under the limitation of 10^{-6} FAR. To improve the performance of the model and prove that our proposed ConeFace can help the SOTA methods, we trained our model together with the ArcFace [5]. The ArcFace† is the version of ArcFace implemented under the Pytorch framework by ourselves and trained with the same settings such as initialization, learning rates, *etc*. as our ConeFace for the fairness of the comparisons. Our ConeFace can consistently improve the performance of the ArcFace for identification and verification on MegaFace Challenge both under the small and large protocols.

4. CONCLUSION

In this paper, we propose an Approximate Pairwise Loss (APL) function, which can exploit both classification information and pairwise information efficiently without intuitively sampling pairwise data. The proposed ConeFace based on APL can be used together with other methods to further improve the model's inter-class separability and intra-class compactness. Comprehensive experiments on several widely used face benchmarks show that the proposed method can consistently improve the performance of the existing state-of-the-art methods.

5. ACKNOWLEDGEMENT

This paper is supported by NSFC (62176155), Shanghai Municipal Science and Technology Major Project (2021SHZ DZX0102) and NSFC (61772330)

6. REFERENCES

[1] Chaochao Lu and Xiaoou Tang, "Surpassing human-level face verification performance on lfw with gaussianface," in *Twenty-ninth AAAI conference on artificial intelligence*, 2015.

[2] Gary B. Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller, "Labeled faces in the wild: A database for studying face recognition in unconstrained environments," Tech. Rep. 07-49, University of Massachusetts, Amherst, October 2007.

[3] Sumit Chopra, Raia Hadsell, Yann LeCun, et al., "Learning a similarity metric discriminatively, with application to face verification," in *CVPR (1)*, 2005, pp. 539–546.

[4] Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Dihong Gong, Jingchao Zhou, Zhifeng Li, and Wei Liu, "Cosface: Large margin cosine loss for deep face recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 5265–5274.

[5] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou, "Arcface: Additive angular margin loss for deep face recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4690–4699.

[6] S. Sengupta, J. C. Chen, C. Castillo, V. M. Patel, and D. W. Jacobs, "Frontal to profile face verification in the wild," in *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2016.

[7] Stylianos Moschoglou, Athanasios Papaioannou, Christos Sagonas, Jiankang Deng, Irene Kotsia, and Stefanos Zafeiriou, "Agedb: the first manually collected, in-the-wild age database," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2017, pp. 51–59.

[8] Ira Kemelmacher-Shlizerman, Steven M Seitz, Daniel Miller, and Evan Brossard, "The megaface benchmark: 1 million faces for recognition at scale," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4873–4882.

[9] Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song, "Sphereface: Deep hypersphere embedding for face recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 212–220.

[10] Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, and Yu Qiao, "Joint face detection and alignment using multitask cascaded convolutional networks," *IEEE Signal Processing Letters*, vol. 23, no. 10, pp. 1499–1503, 2016.

[11] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer, "Automatic differentiation in pytorch," 2017.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[13] Sergey Ioffe and Christian Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[14] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[15] Jie Hu, Li Shen, and Gang Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7132–7141.

[16] Dong Yi, Zhen Lei, Shengcai Liao, and Stan Z Li, "Learning face representation from scratch," *arXiv preprint arXiv:1411.7923*, 2014.

[17] Yandong Guo, Lei Zhang, Yuxiao Hu, Xiaodong He, and Jianfeng Gao, "Ms-celeb-1m: Challenge of recognizing one million celebrities in the real world," *Electronic imaging*, vol. 2016, no. 11, pp. 1–6, 2016.

[18] Aaron Nech and Ira Kemelmacher-Shlizerman, "Level playing field for million scale face recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 7044–7053.

[19] Florian Schroff, Dmitry Kalenichenko, and James Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 815–823.

[20] Kai Zhao, Jingyi Xu, and Ming-Ming Cheng, "Regularface: Deep face recognition via exclusive regularization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 1136–1144.

[21] Weiyang Liu, Yandong Wen, Zhiding Yu, and Meng Yang, "Large-margin softmax loss for convolutional neural networks.," in *ICML*, 2016, vol. 2, p. 7.