論文 / 著書情報
Article / Book Information

Title	Graph Grouping Loss for Metric Learning of Face Image Representations
Author	Nakamasa Inoue
Journal/Book name	2020 IEEE International Conference on Visual Communications and Image Processing (VCIP), , , pp. 152-155
Pub. date	2020, 12
DOI	https://doi.org/10.1109/VCIP49819.2020.9301861
Copyright	(c)2020 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.
Note	This file is author (final) version.

Graph Grouping Loss for Metric Learning of Face Image Representations

Nakamasa Inoue
Tokyo Institute of Technology, Japan
inoue@c.titech.ac.jp

***Abstract*—This paper proposes Graph Grouping (GG) loss for metric learning and its application to face verification. GG loss predisposes image embeddings of the same identity to be close to each other, and those of different identities to be far from each other by constructing and optimizing graphs representing the relation between images. Further, to reduce the computational cost, we propose an efficient way to compute GG loss for cases where embeddings are L_2 normalized. In experiments, we demonstrate the effectiveness of the proposed method for face verification on the VoxCeleb dataset. The results show that the proposed GG loss outperforms conventional losses for metric learning.**

I. Introduction

Face verification is one of the most important research topics in the field of biometric authentication. In the past few decades, signal processing and computer vision techniques have had great success in extracting identity features from face images. In particular, deep convolutional networks have been proven to be effective at embedding images into a vector space, which enables measurement of face similarity by a simple metric such as cosine similarity between vectors.

To optimize network parameters, metric learning has recently attracted attention. Given a set of labeled face images, metric learning aims to assign a small distance between images of the same identity and a relatively large distance between images of different identities. As such, loss functions for metric learning are often designed to directly minimize or maximize distance between images. For example, contrastive loss [1] minimizes distance $d(x_a, x_p)$ and maximizes distance $d(x_a, x_n)$, where x_a is an *anchor* image, x_p is a *positive* image of the same identity as the anchor, and x_n is a *negative* image of a different identity, as shown in Figure 1 (a).

The idea to introduce these three roles of anchor, positive, and negative is widely utilized in metric learning. Triplet loss [2] makes triplets of anchor, positive, and negative images, as shown in Figure 1 (b). Prototypical loss [3] and angle-prototypical loss [4] make prototypes by aggregating positive images. In general, training with triplets or prototypes is more efficient and effective than that with pairs because more than two image embeddings are simultaneously optimized. However, some relation between images remains unused. For example, a connection between the positive image and the negative image is missing in Figure 1 (b). This suggests the idea of a general framework to leverage more dense connections between images by making a graph as shown in Figure 1 (c).

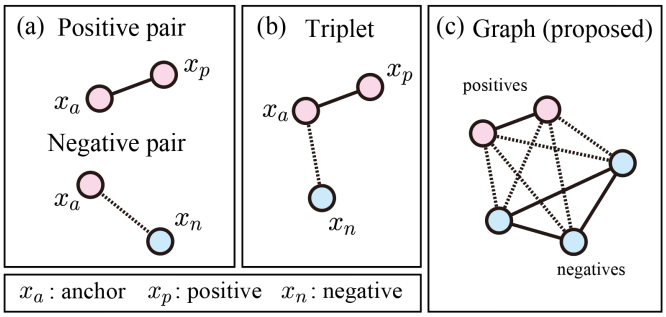

Fig. 1. Illustrations of the relation between images in metric learning. (a) Contrastive loss uses two types of pairs: positive and negative. (b) Triplet loss uses triplets of an anchor, positive, and negative. (c) The proposed GG loss uses a graph representing the general relation between images.

In this paper, we propose a novel loss function, namely Graph Grouping (GG) loss, for metric learning which makes graphs on training images to simultaneously optimize embeddings at each graph. Further, to reduce the computational cost, we propose an efficient way to compute GG loss for cases where embeddings are L_2 normalized. In experiments, we demonstrate that that the proposed loss function outperforms conventional loss functions for metric learning, in terms of face verification performance on the VoxCeleb dataset. In summary, our contributions are three fold.

1) We propose GG loss for metric learning, which optimizes image embeddings for face verification.
2) We propose an efficient way to compute GG loss on L_2-normalized embeddings.
3) We conduct comparison experiments on the VoxCeleb dataset and show that the proposed method outperforms conventional metric learning methods.

II. Related Work

A. Face Identification and Verification

To extract features from images for face identification and verification, researchers have proposed various feature extraction methods in the past few decades. Examples of traditional methods include statistical modeling of heuristic features such as Haar-like features [5]. These methods are often used in lightweight devices. Recent studies have shown that features extracted from deep convolutional networks outperform heuristic features in terms of face identification accuracy.

978-1-7281-8068-7/20/$31.00

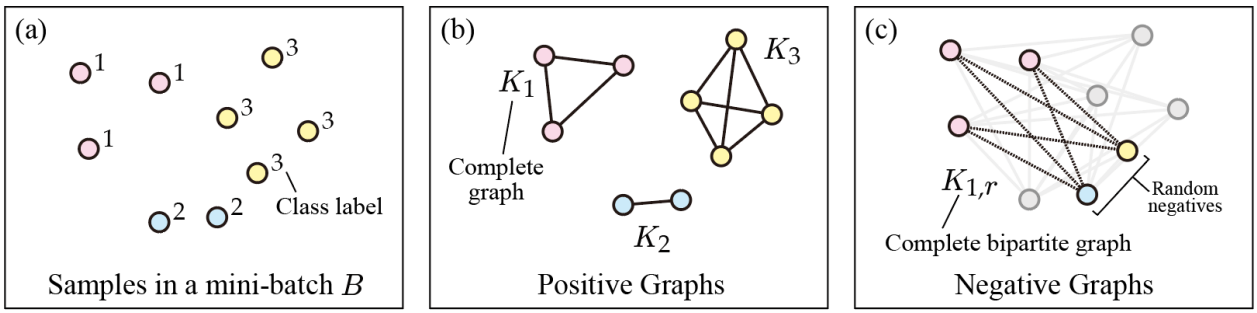

Fig. 2. Graphs defined on a mini-batch B. (a) Samples in a mini-batch. Labels for three classes (identities) are illustrated. (b) Positive graphs: the complete graphs K_c for each class label c are constructed. (c) Negative graphs: the complete bipartite graphs $K_{c,r}$ between the class c and the r-th random negative sets ($r = 1, 2, \cdots, R$) are constructed. The proposed GG loss predisposes edge weights for K_c to be small and those for $K_{c,r}$ to be relatively large.

For instance, the final pooling layer of ResNet [6] or SE-ResNet [7] is often utilized to extract features.

To train these networks, a large-scale labeled dataset is required. Examples of publicly available datasets include MS-Celeb-1M [8], VGGFace2[9], and VoxCeleb[10]. These datasets provide face images or videos for more than 5,000 identities, and are usually large enough to optimize parameters.

B. Metric Learning

Metric learning is a framework for finding an optimized metric space. Previous studies have proposed definitions of loss functions based on metric learning for training neural networks. The simplest ones are contrastive loss [1] and triplet loss [2]. They make pairs or triplets of images to predispose image embeddings to be close or to be far from each other. For face identification, losses based on cosine similarity, such as ArcFace [11] and CosFace [12], often improve the performance. Their formulation can be viewed as an extension of softmax loss. For speaker verification, further extensions including prototypical loss [3] and angle-prototypical loss [4] are also known to be effective.

III. Proposed Method

A. Notation and Settings for Face Verification

Given two face images x and x', the goal of face verification is to determine whether two images are of the same identity. In this paper, we assume that sets of identities for training and testing are disjoint. This setting is the same as that of speaker verification for audio signals [10] and is also similar to zero-shot image recognition [13]. Note that this is more difficult than standard face recognition, where the identity sets for training and testing are the same.

More precisely, a training set $\mathcal{D}$ consists of pairs of a face image $x \in X$ and its identity label $y \in Y$, where X is a set of images and Y is a set of identities for training. A testing set $\mathcal{T}$ consists of pairs of a face image $x \in X$ and its identity label $z \in Z$, where Z is another set of identities, i.e., $Y \cap Z = \emptyset$. Note that, in the testing phase, distance (or similarity) between embeddings $\phi(x), \phi(x')$ is used to determine whether two images are of the same identity. These embeddings are extracted from a hidden layer of a neural network, e.g., from the final pooling layer of ResNet18 [6]. The rest of this section presents a method for training a neural network $\mathcal{N}_\theta$ on $\mathcal{D}$ to extract embeddings $\phi(x)$, where θ is a set of network parameters.

B. Graph Grouping Loss for Metric Learning

The goal of metric learning for face verification is to learn a metric $d(x, x')$ which assigns a small distance between images of the same identity. Assuming that network parameters are iteratively updated by using mini-batch sampling, the proposed GG loss is defined on a mini-batch $B = \{(x_i, y_i) : i = 1, 2, \cdots, N\}$.

Our main idea is to define two types of graphs on B: *positive* graphs and *negative* graphs. Here, positive graphs have edges between images from the same identity and negative graphs have edges between images from the different identities, as illustrated in Figure 2. Based on these graphs, GG loss predisposes the edges of the positive graphs to be short and those of the negative graphs to be long. Specifically, the loss is calculated in the following four steps.

1) **Constructing Positive Graphs.** Let $C = \bigcup_i \{y_i\}$ be a unique set of labels on B. For each label $c \in C$, a positive graph $K_c = (V_c, E_c)$ is constructed by

$$V_c = \{x_i : y_i = c, (x_i, y_i) \in B\}, \quad (1)$$
$$E_c = \{(u, v) : u, v \in V_c, u \neq v\}, \quad (2)$$

where V_c is a set of nodes and E_c is a set of edges. Note that K_c is an undirected complete graph as shown in Figure 2 (b).

2) **Constructing Negative Graphs.** For each label $c \in C$, a negative graph $K_{c,r} = (V_{c,r}, E_{c,r})$ is constructed by

$$V_{c,r} = V_c \cup R_r, \quad (3)$$
$$E_{c,r} = \{(u, v) : u \in V_c, v \in R_r\}, \quad (4)$$

where R_r is a random subset of images of identities excluding c, i.e., $R_r \subset B \backslash V_c$. We repeat sampling T times to obtain negative graphs for $r = 1, 2, \cdots, T$. Note that $K_{c,r}$ is a bipartite complete graph as shown in Figure 2(c).

3) **Defining Edge Weights.** On each graph $G = (V, E)$ constructed in steps 1 and 2, edge weight $w(e)$ is defined by

$$w(e) = d(u, v) \quad (5)$$

where $e = (u, v) \in E$ and $d(u, v)$ is a metric to be learned, such as squared Euclidian distance $d(u, v) = \|u - v\|_2^2$.

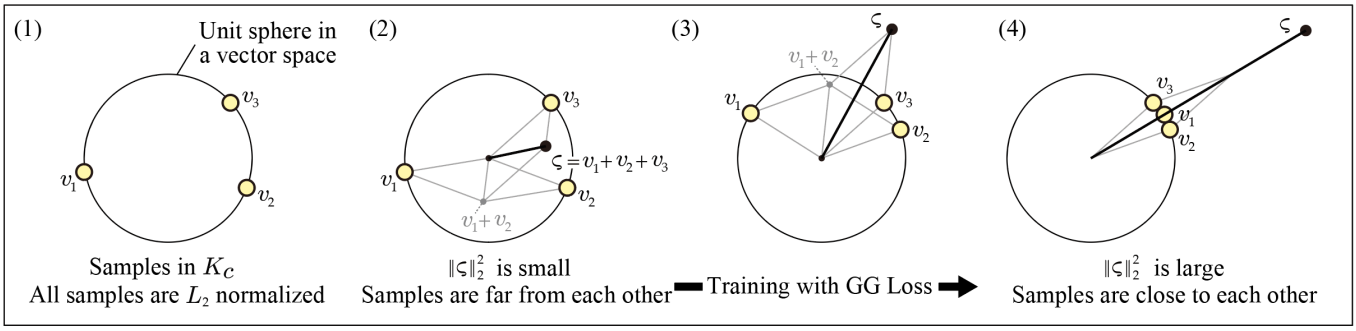

Fig. 3. Illustration of how GG loss works. (1) Samples in K_c for class c. Our two assumptions are illustrated: all samples (embeddings) are in the Euclidian space and they are L_2 normalized. (2) Before training: the sum $\varsigma = \Sigma_{v \in V_c} \phi(v)$ is computed. $\|\varsigma\|_2^2$ is small because samples are far from each other. (3) During training: GG loss predisposes $\|\varsigma\|_2^2$ to be large. (4) After training: $\|\varsigma\|_2^2$ becomes large. This means that the samples are close to each other.

4) **Computing Loss** Finally, GG loss is computed by

$$L = \sum_{c=1}^{C} \frac{\exp(-\gamma \|K_c\|_w + \beta)}{\sum_{r=0}^{R} \exp(-\gamma \|K_{c,r}\|_w + \beta)}, \tag{6}$$

where $\|G\|_w$ is the average of edge weights given by

$$\|G\|_w = \frac{1}{|E|} \sum_{e \in E} w(e), \tag{7}$$

and β, γ are learnable parameters. Intuitively, GG loss can be understood as an extension of softmax loss to the graph space. Note that we define $K_{c,0} = K_c$ to simplify the expression (summation) in the denominator in Eq. (6).

C. Efficient Computation of GG Loss

To naively compute GG loss, all edge weights need to be evaluated. However, this is computationally costly because graphs K_c and $K_{c,r}$ are dense. Here, we present an efficient way to directly compute GG loss by introducing two assumptions: 1) all embeddings $\phi(x)$ are L_2 normalized, i.e., $\|\phi(x)\|_2 = 1$; and 2) the distance $d(x, x')$ is squared Euclidean distance between embeddings, i.e., $d(x, x') = \|\phi(x) - \phi(x')\|_2^2$. The first assumption is satisfied by introducing an L_2 normalization layer at the top of the network. Adding the second assumption is reasonable in practice because it is equivalent to cosine similarity, which often performs well for both face identification [11], [12] and speaker verification [16], [15].

C-1. Efficient computation for complete graphs

Under the two assumptions, $\|K_c\|_w$ for complete graphs is computed as follows:

$$\|K_c\|_w = \frac{1}{|E_c|} \sum_{(u,v) \in E_c} \|\phi(u) - \phi(v)\|_2^2 \tag{8}$$

$$= 2 - \frac{2}{|E_c|} \sum_{(u,v) \in E_c} \phi(u)^T \phi(v) \tag{9}$$

$$= 2 - \frac{1}{|E_c|} (\varsigma^T \varsigma - |V_c|), \tag{10}$$

where ς is the sum of embeddings

$$\varsigma = \sum_{v \in V_c} \phi(v). \tag{11}$$

Note that Eq. (10) is derived from

$$\varsigma^T \varsigma = \left(\sum_{u \in V_c} \phi(u)^T \right) \left(\sum_{v \in V_c} \phi(v) \right) \tag{12}$$

$$= \sum_{u,v:u \neq v} \phi(u)^T \phi(v) + \sum_{u,v:u=v} \phi(u)^T \phi(v) \tag{13}$$

$$= 2 \sum_{(u,v) \in E_c} \phi(u)^T \phi(v) + |V_c|. \tag{14}$$

Eq. (10) shows that we no longer need to compute weights for each node. This reduces the computational cost from $O(|V|^2)$ to $O(|V|)$. Figure 3 is provided to illustrate how the training works.

C-2. Efficient computation for complete bipartite graphs

To efficiently compute $\|K_{c,r}\|_w$, we reuse the values of $\|K_c\|_w$ obtained above. Specifically, $\|K_{c,r}\|_w$ is computed by

$$\|K_{c,r}\|_w = \frac{|E_{c \cup r}| \|K_{c \cup r}\|_w - |E_c| \|K_c\|_w - |E_r| \|K_r\|_w}{|E_{c,r}|}, \tag{15}$$

where K_r and $K_{c \cup r}$ are complete graphs on R_r and $V_{c \cup r} = V_c \cup R_r$, respectively. By computing $\|K_r\|_w$ and $\|K_{c \cup r}\|_w$ in the same way as 1), the computational cost is again reduced from $O(|V|^2)$ to $O(|V|)$.

IV. Experiments

A. Evaluation Settings

We use the VoxCeleb dataset [10] to conduct face verification experiments. This dataset is often used for speaker verification and is also suitable for face verification experiments because disjoint sets of identities for training and testing are provided. Note that this setting is more difficult than that of standard face identification, where the identity sets for training and testing are the same.

For training, the VoxCeleb 2 development set is used; this set consists of 1,092,009 video files for 5,994 identities. To make mini-batches for training at each iteration, video files are randomly selected and then one image frame is randomly extracted from each video file. For testing, the VoxCeleb 1 test set is used; this set consists of 37,611 verification pairs. We define two testing types, *Image* and *Video*, where the number

TABLE I. PERFORMANCE COMPARISON WITH OTHER METHODS. EQUAL ERROR RATE (%) ON THE VOXCELEB 1 TEST SET IS REPORTED. ONLY FACE IMAGES OR VIDEOS ARE USED IN THE EXPERIMENTS.

Method	Testing Type	
	Image	Video
Softmax Loss	13.25	12.04
CosFace Loss [12]	9.84	7.97
ArcFace Loss [11]	8.70	7.18
Prototypical Loss [3]	9.42	7.28
Angle-Prototypical Loss [4]	6.78	5.09
GG Loss (Proposed)	**6.10**	**4.27**
Audio only	1.90	
Multimodal (Audio+GGLoss-Video)	**0.89**	

of image frames used for testing is one and ten per video clip, respectively. Testing videos (image frames) used in our experiments are provided in [14]. The evaluation measure is equal error rate (EER). This is the standard evaluation measure for this dataset.

Implementation details are as follows. ResNet18 [6] is used to extract embeddings $\phi(x)$ from images, where $\phi(x)$ is a 256-dimensional activation vector at the final pooling layer. Network parameters are optimized by using the momentum SGD optimizer with a batch size of 256 for 30 epochs. The learning rate is initialized by 0.001, and decayed by 2 at every 2 epoch. Training took about 1.5 days with four NVIDIA P100 GPUs. The other hyperparameters are set to be the default values of the PyTorch implementation.

B. Experimental Results

Table I shows performance in terms of EER on the VoxCeleb 1 test set. For comparison, we report results using softmax loss, CosFace loss [12], ArcFace loss [11], prototypical loss [3], angle-prototypical loss [4], and the proposed GG loss. We see that the proposed method outperforms the conventional methods. This demonstrates the effectiveness of the proposed GG loss. We also see that using multiple frames from video for testing improves the performance for all types of loss.

For more detailed analysis, the detection error tradeoff (DET) curves are reported in Figure 4. As can be seen, the tendency is the same as that in Table I and the proposed method uniformly improves the performance. Through experiments, we observed that the proposed GG loss converges faster than the other methods. This is because the graphs used to define GG loss are dense. With dense graphs, many samples are simultaneously optimized, and this helps to make convergence first.

Finally, we report results using both audio and visual streams. For the audio stream, we use ResNet18 with 32-dim filterbank features without data augmentation and vanilla softmax loss [17]. As shown in Table I, combining the two streams further improves the verification performance, and achieves a 0.89 % EER. This result confirms that face verification and speaker verification benefit from each other.

V. CONCLUSION

This paper proposed graph grouping (GG) loss for metric learning and an efficient way to compute GG loss on L_2-normalized embeddings. Our face verification experiments on the VoxCeleb dataset showed that the proposed loss outperforms conventional metric learning methods. For future work, multi-modal metric learning to simultaneously optimize audio and visual embeddings would be interesting.

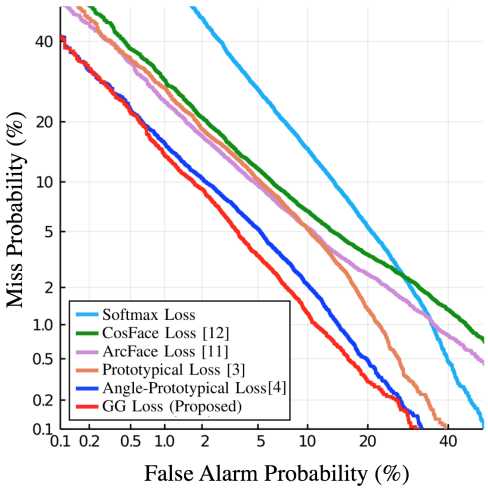

Fig. 4. Detection-error-tradeoff (DET) curves. Our method is compared with five conventional methods.

REFERENCES

[1] R. Hadsell, S. Chopra, and Y. LeCun. Dimensionality reduction by learning an invariant mapping. In *CVPR*, pp. 1735–1742, 2006.

[2] E. Hoffer and N. Ailon. Deep metric learning using triplet network. *International Workshop on Similarity-Based Pattern Recognition (SIMBAD)*, pp. 84–92, 2015.

[3] J. Snell, K. Swersky, and R. Zemel. Prototypical networks for few-shot learning. In *NeurIPS*, pp. 4077–4087, 2017.

[4] J. S. Chung, J. Huh, S. Mun, M. Lee, H.S. Heo, S. Choe, C. Ham, S. Jung, B.-J. Lee, and I. Han. In defence of metric learning for speaker recognition. *arXiv:2003.11982*, 2020.

[5] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. *CVPR*, 2001.

[6] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *CVPR*, pp. 770–778, 2016.

[7] J. Hu, L. Shen and G. Sun. Squeeze-and-excitation networks . *CVPR*, 2018.

[8] Y. Guo, L. Zhang, Y. Hu, X. He, and J. Gao. MS-Celeb-1M: A dataset and benchmark for large scale face recognition. *ECCV*, 2016.

[9] Q. Cao, L. Shen, W. Xie, O. M. Parkhi, A. Zisserman. VGGFace2: A dataset for recognising face across pose and age. *International Conference on Automatic Face and Gesture Recognition*, 2018.

[10] J.S. Chung A. Nagrani, and A. Zisserman. Voxceleb2: Deep speaker recognition. *Interspeech*, 2018.

[11] J. Deng, J. Guo, N. Xue, and S. Zafeiriou. ArcFace: Additive angular margin loss for deep face recognition. *CVPR*, pp. 4690–4699, 2019.

[12] H. Wang, Y. Wang, Z. Zhou, X. Ji, D. Gong, J. Zhou, Z. Li, and W. Liu. CosFace: Large margin cosine loss for deep face recognition. CVPR, pp. 5265–5274, 2018.

[13] C. H. Lampert. Learning to detect unseen object classes by between-class attribute transfer. *CVPR*, 2009.

[14] A. Nagrani, S. Albanie, and A. Zisserman. Seeing voices and hearing faces: cross-modal biometric matching. *CVPR*, 2018.

[15] Y. Liu, L. He, and J. Liu. Large Margin Softmax Loss for Speaker Verification. *Interspeech*, 2019.

[16] L. Wan, Q. Wang, A. Papir, and I.L. Moreno. Generalized end-to-end loss for speaker verification. *ICASSP*, pp. 4879–4883, 2018.

[17] N. Inoue and K. Goto, Semi-Supervised Contrastive Learning with Generalized Contrastive Loss and Its Application to Speaker Recognition, arXiv:2006.04326.