Variational Prototype Learning for Deep Face Recognition

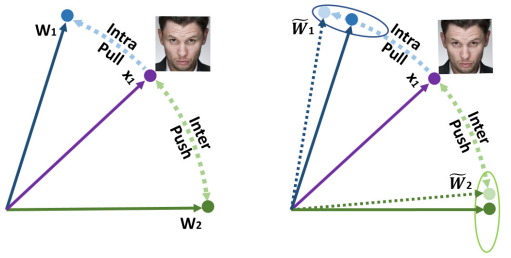
Jiankang Deng * [1,2,3] Jia Guo * [3] Jing Yang[4]
Alexandros Lattas[1,2] Stefanos Zafeiriou[1,2]
[1]Huawei [2]Imperial College [3]InsightFace [4]University of Nottingham
{j.deng16, a.lattas, s.zafeiriou}@imperial.ac.uk
guojia@gmail.com, jing.yang2@nottingham.ac.uk

Abstract

Deep face recognition has achieved remarkable improvements due to the introduction of margin-based softmax loss, in which the prototype stored in the last linear layer represents the center of each class. In these methods, training samples are enforced to be close to positive prototypes and far apart from negative prototypes by a clear margin. However, we argue that prototype learning only employs sample-to-prototype comparisons without considering sample-to-sample comparisons during training and the low loss value gives us an illusion of perfect feature embedding, impeding the further exploration of SGD. To this end, we propose Variational Prototype Learning (VPL), which represents every class as a distribution instead of a point in the latent space. By identifying the slow feature drift phenomenon, we directly inject memorized features into prototypes to approximate variational prototype sampling. The proposed VPL can simulate sample-to-sample comparisons within the classification framework, encouraging the SGD solver to be more exploratory, while boosting performance. Moreover, VPL is conceptually simple, easy to implement, computationally efficient and memory saving. We present extensive experimental results on popular benchmarks, which demonstrate the superiority of the proposed VPL method over the state-of-the-art competitors.

1. Introduction

Recent state-of-the-art face recognition methods [35, 28, 45, 8, 41, 19] mainly focus on employing margin penalty to enhance the discriminative feature embedding. The pioneering work [35] uses the Triplet loss to enforce faces from the same class to be closer than faces from different classes, by a clear margin in the Euclidean space. However, the sample-to-sample comparisons in the Triplet loss are constrained to the local mini-batch, and therefore sophisticated mining strategies are required to choose an informative mini-batch [34, 35] and select representative triplets within the mini-batch [52, 33, 40]. On large-scale datasets [31, 13], there is a combinatorial explosion in the number of triplets, which results in an immensely complicated mining step. To this end, margin-based softmax methods [28, 5, 45, 43, 8, 41, 19] introduce a margin penalty into the prototype learning and conduct global comparisons between training samples and class-wise prototypes. The sample-to-prototype comparison is more efficient and stable than the sample-to-sample comparison, as (a) the prototype number is much smaller than the sample number, and (b) both the class-wise prototype and the embedding network are optimized in every iteration during training. In that manner, the prototypes try to memorize positive features and forget negative features, while the embedding features try to get close to positive prototypes and keep far apart from negative prototypes.



(a) Prototype Learning (b) Variational Prototype Learning

Figure 1: Difference between the Prototype Learning (PL) and the proposed Variational Prototype Learning (VPL). The prototype learning (*e.g.* softmax loss) employs sample-to-prototype comparisons and represents every class as a point in the latent space. By contrast, the proposed VPL represents every class as a distribution in order to simulate sample-to-sample comparison within the classification framework.

* Equal contributions.
InsightFace is a nonprofit Github project for 2D and 3D face analysis.

Even though the margin-based softmax methods show great efficiency, stability and capability in face recognition, each class is only represented by one point in the latent space, bearing no variation information, as shown in Fig. 1(a). There exists abundant literature on modeling the faces as subspaces [42, 3, 48, 7], manifolds [17, 20] or probabilistic distributions [37, 1] in the feature space. In these works, each class is represented by a group of discrete points or a continuous distribution in the latent space, instead of one point. In fact, representing each class by one point as in the prototype learning can sometimes lead to model degeneration. In real-world applications, the face training data inherently follows an unbalanced distribution [56, 59, 61, 11], where some identities have plenty of samples, while other identities only contain very few samples. In Sec. 4.2, our derivative analysis indicates that the prototype is learned by continually absorbing positive sample features and eliminating negative sample features. When data becomes shallow, there are limited intra-class variations and the prototype vector can easily remember all samples within one class [11]. For instance, consider one training identity with only two facial images and the corresponding deep features x_1 and x_2. If the prototype can learn to remember the centroid $(x_1 + x_2)/2$, regardless of the degree of similarity between x_1 and x_2, the sample-to-prototype similarities can be very high. Therefore, the single point representation of the prototype can sometimes hinder the further exploration of the SGD solver, and thus the model may converge at suboptimal local-minima.

To deal with the aforementioned model degeneration, recent methods attempt to either improve the margin values for the tail classes [27] or recall the benefit from sample-to-sample comparisons [61, 11]. AdaptiveFace [27] proposes adaptive margins for rich and poor classes. CVC [61] employs sample features to initialize the prototypes. SST [11] employs the semi-Siamese networks and constructs a dynamic queue with gallery features to replace original prototypes. However, each method introduces significant trade-offs. The margin average loss proposed in [27] only enforces the margin values to increase for all classes, which can lead to over-fitting on the training data. The classification-verification-classification strategy proposed in [61] is not an end-to-end solution, and its stepwise fine-tuning is arduous. The Semi-Siamese networks proposed in [11] employ a probe-set network to embed the probe features and another gallery-set network to update prototypes by gallery features, which results in additional memory consumption. Finally, CVC [61] and SST [11] are only designed to tackle the bisample problem instead of handling general deep face recognition.

In this paper, we first identify the limitations of prototype learning, which represents each class as a point in the latent space and employs sample-to-prototype comparisons during training without considering class-wise variations. This single point approximation of class representation facilitates the network training but also impedes the further exploration of SGD. To this end, we propose the Variational Prototype Learning (VPL) which represents every class as a distribution instead of a point in the latent space, as illustrated in Fig. 1(b). Based on the observation of slow feature drift phenomenon after the early phase of training, we directly inject memorized features from recent mini-batches into the corresponding prototypes to approximate variational prototype sampling. The proposed VPL can easily simulate sample-to-sample comparisons within the classification framework, encouraging the SGD solver to be more exploratory and boosting performance considerably.

To summarize, our key contributions are:

- We point out the limitations of the prototype learning and propose a novel Variational Prototype Learning (VPL) method which represents each class as a distribution instead of a point in the latent space.
- Based on the observation of slow feature drift, we design a computationally efficient and memory-saving way for the variational prototype sampling, that is, injecting memorized features into the corresponding prototypes. In our VPL, both sample-to-prototype comparisons and sample-to-sample comparisons are exploited.
- The proposed VPL is a plug-and-play module, providing an orthogonal improvement to recent margin-based or mining-based softmax methods. Extensive experimental results on popular benchmarks demonstrate the superiority of our VPL over the state-of-the-art competitors in deep face recognition.

2. Related Work

Variational Face Encoding: Most recent face recognition methods [28, 45, 8, 19, 41] enforce intra-class compactness as well as inter-class separability through comparing sample features with class-wise prototypes. Both the face image and the class-wise prototype are represented as a deterministic point in the latent space. Probabilistic Face Embeddings (PFE) [38] and Data Uncertainty Learning (DUL) [4] consider data uncertainty for face recognition by mapping each sample as a Gaussian distribution [26], instead of a fixed point, in the latent space. More specifically, PFE and DUL employ parallel FC branches to simultaneously estimate the embedded feature (mean) and the uncertainty (variance), predicting small variance for high-quality samples but large variance for ambiguous or noisy ones. The uncertainty modeling in PFE and DUL is designed for a single face image. By contrast, the proposed VPL focuses on the distribution representation for the class-wise prototype. As we represent each class as a distribution in the

latent space, training samples are compared with the variational prototype representing each class, bringing sample-to-sample comparisons within the classification framework.

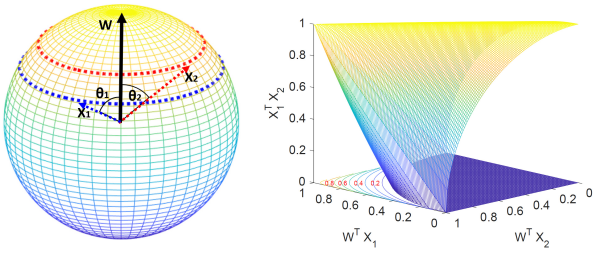
Feature Memory Bank: The non-parametric memory module paradigm has shown power in various vision tasks, including few-shot learning [53], unsupervised learning [54, 15], domain adaptation [60], metric learning [49] and face embedding [25]. MOCO [15] builds a dynamic queue from preceding mini-batches to construct a rich set of negative samples for unsupervised learning and a momentum update is designed to slowly update the encoder to ensure the consistency between different iterations. BroadFace [25] maintains a large queue to store a vast number of embedding vectors accumulated over past iterations, in order to increase the batch size. To deal with feature drift, a compensation method is proposed in BroadFace to reduce the expected error between the current and enqueued embedding vectors, by employing the difference of the identity-representative prototypes of current and past iterations. XBM [49] memorizes the embeddings of past iterations, allowing the model to collect sufficient hard negative pairs across multiple mini-batches for deep metric learning. As the feature drifts slowly after the early phase of training, XBM directly utilizes these cross-batch embeddings for training without a momentum update [15] or a compensation [25]. In MOCO, BroadFace and XBM, features from preceding iterations are directly employed for intra-class or inter-class comparisons. In contrast, we directly inject memorized features into prototypes to approximate variational prototype sampling. The proposed VPL can be directly integrated into the existing prototype learning framework (*e.g.* margin-based or mining-based softmax methods), as a plug-and-play module with a negligible consumption of GPU memory and a rounding cost of extra computation.

3. Limitations of Prototype Learning

In Circle Loss [41], learning with class-level labels and pair-wise labels are unified into one paradigm. Given class-level labels, the classification loss (*e.g.* softmax loss) optimizes the similarity between training samples and class-wise prototypes, maximizing the intra-class similarity as well as minimizing the inter-class similarity. By contrast, the pair-wise loss (*e.g.* Triplet [35]) directly optimizes the similarity between positive and negative sample pairs in the feature space without any prototype. Below, we compare similarities and differences of the softmax loss and the tuplet loss [40].

The most widely used classification loss function, softmax loss, is presented as follows:

$$\mathcal{L}_{softmax} = -\log \frac{e^{W_{y_i}^T x_i}}{e^{W_{y_i}^T x_i} + \sum_{j=1, j\neq y_i}^{N} e^{W_j^T x_i}}, \quad (1)$$



(a) x_1, x_2 and W (b) Similarity Range of $x_1^T x_2$

Figure 2: Given the sample-to-prototype similarities ($W^T x_1$ and $W^T x_2$), the sample-to-sample similarity $x_1^T x_2$ can vary greatly.

(a) Examples (b) Similarity Distribution

Figure 3: Limitations of the prototype learning. (a) High sample-to-prototype similarities can be easily achieved, but there remain low sample-to-sample similarities. (b) Sample-to-sample similarities obviously lag behind sample-to-prototype similarities.

where $W_j \in \mathbb{R}^d$ denotes the j-th column of class-wise prototype $W \in \mathbb{R}^{d\times N}$, d is the feature dimension, N is the class number, and $x_i \in \mathbb{R}^d$ denotes the feature of the i-th sample, belonging to the y_i-th class. For the pair-wise loss, we consider an (N+1)-tuplet [40] of embedding features $\{x_i, x_{y_i}, x_1, \cdots, x_{N-1}\}$: x_{y_i} is from a positive sample to x_i and $\{x_j\}_{j=1}^{N-1}$ are from negative samples. The (N+1)-tuplet is defined as follows:

$$\mathcal{L}_{tuplet} = -\log \frac{e^{x_{y_i}^T x_i}}{e^{x_{y_i}^T x_i} + \sum_{j=1}^{N-1} e^{x_j^T x_i}}. \quad (2)$$

When $N = 2$, the (2+1)-tuplet loss highly resembles the triplet loss [35] as there is only one negative sample for each positive pair. Furthermore, the tuplet loss employs the "$\sum \exp(\cdot)$" operation for "soft" mining among samples, while the triplet loss utilizes canonical hard mining [41]. Eq. 2 is similar to Eq. 1 when the positive sample $x_{y_i}^T$ is viewed as the positive prototype $W_{y_i}^T$ and the negative sample x_j^T is viewed as the negative prototype W_j^T. However, the prototype of the classification loss only represents the center point of each class [8], carrying no intra-class variation. In contrast, the comparison between samples contains real-time feature variations from the embedding network.

As shown in Fig. 2(a), x_1 and x_2 are embedding features belonging to the same class and the corresponding prototype is W. Based on ℓ_2 normalization, both features and prototypes are distributed on a hyper-spherical embedding space. When the sample-to-prototype angle is fixed,

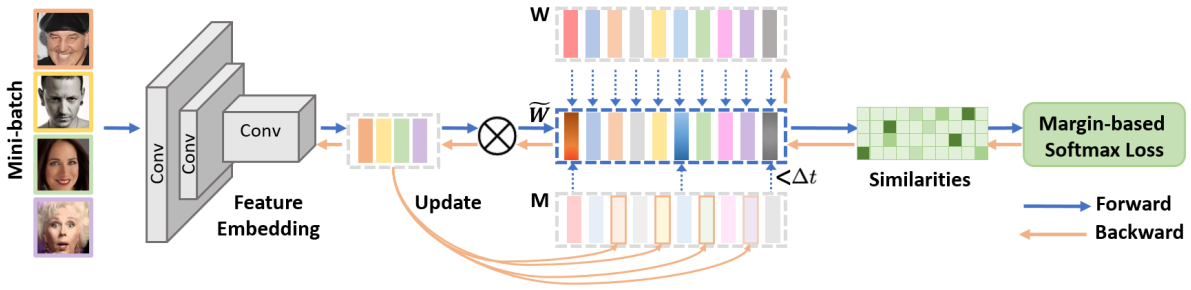


Figure 4: Variational Prototype Learning (VPL) for deep face recognition. In the forward process, embedding features $x \in \mathbb{R}^{d\times B}$ are compared with variational prototypes $\widetilde{W} \in \mathbb{R}^{d\times N}$, which are generated by combining prototypes $W \in \mathbb{R}^{d\times N}$ and memorized sample features $M \in \mathbb{R}^{d\times N}$. In the backward process, the gradient on features is affected by the variational prototypes, thus the embedding network can learn an improved fitting on the training data. After each iteration, the feature memory bank is updated by the new embedding features and the oldest features over Δt are set as invalid.

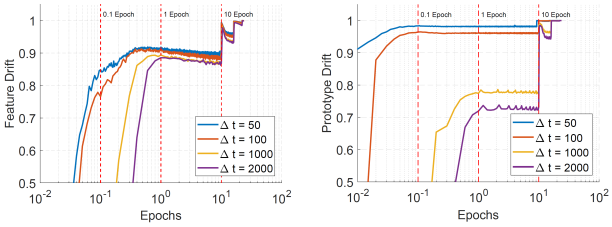
the feature is still free to move along an arc on the hypersphere. In Fig. 2(b), we draw the of the upper bound and lower bound of $x_1^T x_2$, given $W^T x_1$ and $W^T x_2$. Considering that the similarity between samples ranges from around 0 (orthogonal faces from different subjects) to 1 (same subject), we only show the lower bound above 0 and draw the contour lines at $0.2, 0.4, 0.6$ and 0.8. Intuitively, the sample-to-sample similarity can not be guaranteed if the sample-to-prototype similarities are not high enough.

To further confirm the above geometric analysis, we randomly select one class from our training data (*i.e.* MS1M [13]) and visualize the sample-to-prototype and sample-to-sample similarities in Fig. 3(a). Here, we employ the ArcFace model [8], which is comparable to state-of-the-art face recognition methods. Even though the overall sample-to-prototype similarities are high, there exist low similarities between genuine pairs. In Fig. 3(b), we show the similarity distributions of all sample-to-prototype pairs and positive sample-to-sample pairs from our training data. It is apparent that (a) sample-to-prototype pairs are much less than sample-to-sample pairs, and (b) sample-to-sample similarities are much lower than sample-to-prototype similarities. For the open-set face recognition, the prototypes of training classes are discarded during testing. Nevertheless, the prototype of each class is optimized to adapt to the imperfect feature embedding during training, resulting in a misconception of well-fitting and impeding the further optimization of the embedding network.

4. Variational Prototype Learning

To address the aforementioned limitations of the prototype learning, we propose a Variational Prototype Learning (VPL) method. VPL optimizes the similarity between training samples and a set of variational prototypes:

$$\mathcal{L}_{VPL} = -\log \frac{e^{\widetilde{W}_{y_i}^T x_i}}{e^{\widetilde{W}_{y_i}^T x_i} + \sum_{j=1, j\neq y_i}^{N} e^{\widetilde{W}_j^T x_i}}, \qquad (3)$$



(a) Feature Drift (b) Prototype Drift

Figure 5: Slow drift phenomena observed in features and prototypes. After one epoch, the features drift within a relatively small cosine variance even under a large interval (*e.g.* $\Delta t = 2000$).

where the variational prototype $\widetilde{W}_j$ is sampled from the class-wise distribution.

4.1. Variational Prototype

Feature consistency across mini-batches: The embedding features of past mini-batches are considered drifted [25] as the model parameters are continuously changing throughout the training process. Such out-of-date features are usually discarded or compensated [25], but we find that the drifting speed of the embedding features is very slow after a short time of training (*e.g.* one epoch). To observe the speed of feature drifting, we randomly sample one instance per class to construct a fixed sample set and then train ResNet50 on MS1M [10] from scratch with the ArcFace loss [8]. For a step interval Δt, the feature drift is calculated by $\frac{1}{N}\sum_{i=1}^{N} x_{i,t}^T x_{i,t+\Delta t}$. Similarly, we also compute the prototype drift over all classes through $\frac{1}{N}\sum_{j=1}^{N} W_{j,t}^T W_{j,t+\Delta t}$. As shown in Fig. 5(a), the features change drastically at the early phase but become relatively stable after one epoch. Even though the network parameters are changing in every step, the feature drift is small (~ 0.87) even under large step intervals (*e.g.* $\Delta t = 2000$). By contrast, there is an accumulative drift phenomenon on the prototypes (in Fig. 5(b)) when the step interval increases from 50 to 2000. After the learning rate decreases from 0.1

to 0.01 at ten epochs, the drift on both features and prototypes gets extremely slow.

Feature injection into the prototype: The slow drift on features indicates that the past embedding features within a certain number of steps can be approximately viewed as the output of the current network. As these historical features need no additional computation and carry the variation information of each class, we employ a memory bank to store them. At an early stage, the feature drift is relatively large, therefore, we train the networks for several epochs, until the features are stable across hundred of steps. Then, we create an empty memory bank $M \in \mathbb{R}^{d\times N}$ and a life indicator $T \in \mathbb{R}^{1\times N}$. When the face feature $\{x_i, y_i\}$ from the mini-batch is assigned to M_{y_i}, the corresponding life indicator T_{y_i} is refreshed to Δt. After each training step, we set $T_{y_i} = T_{y_i} - 1$. We define a *feature injection ratio* as $\frac{1}{N}\sum_{j=1}^{N} 1\{T_j > 0\}$, where $1(\cdot)$ is the indicator function which is 1 when the statement is true and 0 otherwise. Judging from the life indicator, we can select valid features from the memory bank. Then, these features are injected into the prototypes for the variational prototype learning. Formally, the variational prototype can be formulated as below:

$$\begin{aligned}\widetilde{W}_j &\approx (1 - 1\{T_j > 0\}\lambda)W_j + 1\{T_j > 0\}\lambda M_j \\ &\approx \lambda_1 W_j + \lambda_2 M_j,\end{aligned} \tag{4}$$

where λ is the hyper-parameter controlling the weight of injected features. We abbreviate the weights on prototypes and injected features as λ_1 and λ_2, respectively. In VPL, each training sample is compared with a linear combination of all prototypes and valid features stored in the memory bank, simultaneously enabling sample-to-prototype and sample-to-sample comparisons during training.

4.2. Derivative Analysis

For the prototype learning, the derivatives to a class-wise prototype $W_j \in \mathbb{R}^d$ and a sample feature $x_i \in \mathbb{R}^d$ are:

$$\begin{aligned}\frac{\partial \mathcal{L}_{PL}}{\partial x_i} &= \sum_{j=1}^{N}(p_{ij} - 1\{y_i == j\})W_j, \\ \frac{\partial \mathcal{L}_{PL}}{\partial W_j} &= \sum_{i=1}^{B}(p_{ij} - 1\{y_i == j\})x_i, \\ \text{with} \quad p_{ij} &= \frac{e^{W_j^T x_i}}{\sum_{k=1}^{N} e^{W_k^T x_i}},\end{aligned} \tag{5}$$

where N is the class number, B is the batch size, and p_{ij} is the similarity between the sample x_i and the prototype W_j. After the derivatives are calculated, the prototype is updated by $W_j = W_j - \eta\frac{\partial \mathcal{L}_{PL}}{\partial W_j}$, where η is the learning rate. The derivatives of x_i will be back-propagated to update parameters of the whole network. In Eq. 5, the derivative of the sample feature is a "weighted sum" over the prototypes of all classes and the derivative of prototype is a "weighted sum" over sample features within the mini-batch. From the view of features, the network will be updated towards a direction that is close to the ground-truth prototype $(+\eta(1 - p_{ij})W_{y_i})$ and far apart from inter-class prototypes $(-\eta p_{ij} W_j)$. From the view of prototypes, the prototype W_j belonging to j-th class will be updated towards a direction that is close to features of j-th class $(+\eta(1 - p_{ij})x_{y_i==j})$ and far apart from features of other classes $(-\eta p_{ij} x_{y_i\neq j})$. Therefore, both the feature embedding network and the prototype are optimized through adding intra-class signals and removing inter-class signals weighted by the similarity p_{ij}. For margin-based softmax loss [45, 43], the target logit is intentionally decreased by a constant margin. Therefore, no matter how close the sample is to the corresponding prototype, there will still be feature signals added into the corresponding prototype. Therefore, the prototype is nearly synchronized with the embedding feature center, as observed in the experiments of ArcFace [8].

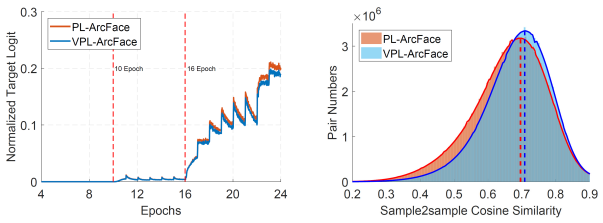
For the proposed VPL, the sample feature x_i is compared with the variational prototype $\widetilde{W}_j$, which is a linear combination of the prototype W_j and the memorized sample feature M_j. As illustrated in Fig. 4, both sample-to-prototype and sample-to-sample comparisons are included in the forward process when computing the probability p_{ij}. Furthermore, the derivatives of a sample feature $x_i \in \mathbb{R}^d$ and a class-wise prototype $W_j \in \mathbb{R}^d$ are:

$$\begin{aligned}\frac{\partial \mathcal{L}_{VPL}}{\partial x_i} &= \sum_{j=1}^{N}(p_{ij} - 1\{y_i == j\})(\underbrace{\lambda_1 W_j}_{Prototype} + \underbrace{\lambda_2 M_j}_{Sample}), \\ \frac{\partial \mathcal{L}_{VPL}}{\partial W_j} &= \lambda_1 \sum_{i=1}^{B}(p_{ij} - 1\{y_i == j\})x_i,\end{aligned} \tag{6}$$

where the derivatives of the current sample feature are affected by both the prototype and the sample feature saved in the memory bank, while the derivatives of the prototype remains as before except for an additional weight. In the proposed VPL, W_j still represents the class center and the sample-to-sample comparisons mainly enforce the embedding network to be more discriminative.

4.3. VPL-ArcFace

As Eq. 3 follows the standard softmax formulation, VPL can be easily adopted to margin-based or mining-based softmax methods (*e.g.* SphereFace [28], CosFace [45, 43], ArcFace [8], AdaptiveFace [27], Circle-Loss [41] and CurricularFace [19]) to orthogonally enhance discriminative feature embedding. Based on ℓ_2 normalization on both features and variational prototypes, $\widetilde{W}_j^T x_i = \left\|\widetilde{W}_j\right\| \|x_i\| \cos\widetilde{\theta}_j = \cos\widetilde{\theta}_j$. For instance, the VPL-ArcFace loss can be formu-



(a) Normalized Target Logit (b) Intra-class Similarities

Figure 6: Comparisons between ArcFace and VPL-ArcFace. (a) The normalized target logit is slightly lower due to sample-to-sample comparisons. (b) Intra-class sample-to-sample similarities can be significantly increased, indicating a better model fitting on the training data.

lated as below:

$$\mathcal{L}_{VPL-Arc} = -\log \frac{e^{s\cos(\widetilde{\theta}_{y_i}+m)}}{e^{s\cos(\widetilde{\theta}_{y_i}+m)} + \sum_{j=1,j\neq y_i}^{N} e^{s\cos\widetilde{\theta}_j}}, \tag{7}$$

where $\widetilde{\theta}_j$ is the angle between the feature x_i and the variational prototype $\widetilde{W}_j$, m is the additive angular margin set as 0.5, and s is the feature re-scale parameter set as 64.

As shown in Fig. 6(a), the normalized target logit of the proposed VPL is slightly lower than that of PL due to the existence of sample-to-sample comparisons. The lower normalized target logit directly contributes to the higher loss by the operation of $-log(\cdot)$. Therefore, VPL can encourage the SGD solver to be more exploratory and further bring continuous gradient propagation so as to find a better local-minima. Even though both of VPL and ArcFace decrease the normalized target logit, VPL is an orthogonal improvement to ArcFace. As we point out in Sec. 3, high sample-to-prototype similarities are not sufficient conditions for high similarities between intra-class samples. ArcFace aims at encouraging sample-to-prototype similarities with a fixed margin, while VPL considers sample-to-sample similarities within classes. In Fig. 6(b), we show the distributions of all intra-class sample-to-sample similarities on the training data. Compared to the baseline, VPL can significantly increase similarities between intra-class samples, indicating a better learning on the training data.

5. Experiments and Results

5.1. Implementation Details

Datasets. For training, we employ the refined version of MS1M [13, 10] as our training data, in order to conduct a fair comparison with other methods. For testing, we extensively evaluate the proposed VPL on popular benchmarks, including LFW [18], CFP-FP [36], CPLFW [57], AgeDB [30], CALFW [58], IJB-B [51], IJB-C [29] and MegaFace [23]. To strictly report the performance, we employ the pre-trained ArcFace model [8] to automatically remove the overlapping identities from our training data. More specifically, the feature center of each subject from our training data is compared with identity centers of LFW [18], CFP-FP [36], AgeDB [30], IJB-C [29], and FaceScrub [32], then overlaps are deleted if the cosine similarity is higher than 0.7. As presented in Tab. 1, our final training data includes 4.5M images of 82K identities. To evaluate face recognition across races [47], we also test the proposed VPL on InsightFace Recognition Test (IFRT) [21], which contains 1.6M images of 242K identities (non-celebrity) covering four demographic groups: African, Caucasian, Indian and Asian [55, 12, 47, 46]. For each demographic group, all pairs between gallery and probe sets are used for the 1:1 face verification.

Datasets	Description	#Identity	#Image
MS1M [13, 10]		93K	5.1M
MS1M (-overlap)		82K	4.5M
LFW [18]	Saturated	5,749	13,233
CFP-FP [36]	Pose	500	7,000
CPLFW [57]		5,749	11,652
AgeDB [30]	Age	568	16,488
CALFW [58]		5,749	12,174
IJB-B [51]	Image&Frame	1,845	76.8K
IJB-C [29]		3,531	148.8K
MegaFace [23]	Large	530	1M distractor
IFRT [21]	Race&Large	242K	1.6M

Table 1: Face datasets for training and testing. For strict evaluation, we remove the subjects that are also included in the test dataset from our training data.

λ	Δt	LFW	CFP-FP	AgeDB	IJB-C
$\lambda = 0$	0	99.78	98.54	98.05	96.21
$\lambda = 0.05$	100	99.80	98.69	98.26	96.42
$\lambda = 0.1$	100	99.83	98.80	98.21	96.50
$\lambda = 0.15$	100	**99.83**	**98.96**	**98.38**	**96.61**
$\lambda = 0.2$	100	99.80	98.82	98.28	96.41

Table 2: Results of VPL-ArcFace models (ResNet50) trained with different feature injection weights. TAR@FAR=1e-4 is reported on IJB-C.

Δt	M_j ratio	LFW	CFP-FP	AgeDB	IJB-C
$\Delta t = 0$	0%	99.78	98.54	98.05	96.21
$\Delta t = 50$	26.3%	99.83	98.77	98.30	96.47
$\Delta t = 100$	42.2%	**99.83**	**98.96**	**98.38**	**96.61**
$\Delta t = 200$	61.8%	99.82	98.94	98.33	96.59
$\Delta t = 1000$	94.1%	99.76	98.35	98.10	96.35
$\Delta t = 2000$	97.9%	99.70	98.21	97.97	96.22

Table 3: Results of VPL-ArcFace models (ResNet50) trained with different feature injection ratios. The feature injection weight λ is fixed as 0.15. TAR@FAR=1e-4 is reported on IJB-C.

Base Model	Diff	LFW	CFP-FP	AgeDB	IJB-C
Softmax-Norm	PL	99.48	96.99	95.70	91.32
	VPL	99.65	97.56	96.23	92.54
CosFace [45]	PL	99.80	98.51	97.96	96.18
	VPL	99.81	98.81	98.24	96.52
ArcFace [8]	PL	99.78	98.54	98.05	96.21
	VPL	99.83	98.96	98.38	96.61
AdaptiveFace [27]	PL	99.80	98.62	98.08	96.28
	VPL	99.83	98.98	98.36	96.62
CurricularFace [19]	PL	99.80	98.58	98.10	96.30
	VPL	99.83	99.01	98.38	96.65

Table 4: Orthogonal improvements to existing margin-based and mining-based softmax methods by using the proposed VPL (ResNet50).

Models	African	Caucasian	Indian	Asian	All
ArcFace	76.22	86.20	84.24	37.27	70.99
VPL-ArcFace	76.60	86.58	84.57	41.10	73.91
VPL on Asian	76.47	86.33	84.49	39.85	72.49
DBM [2]	76.45	85.94	84.01	38.82	71.71
VPL on Tail	76.58	86.30	84.35	40.48	72.78
AdaptiveFace [27]	75.91	86.22	83.69	38.97	71.46

Table 5: The 1:1 verification accuracy (%) of the proposed VPL (ResNet50) on IFRT. "Asian" and "Tail" refer to the variational prototype is only applied to the Asian group and the long-tail data. TAR@FAR=1e-6 is reported on each demographic group. ([MS1M, ResNet50, Loss*]).

Experimental settings. For data prepossessing, we follow ArcFace [8] to generate the normalized face crops (112×112) by utilizing five facial points [9]. For the embedding network, we use the most widely used CNN architectures, ResNet50 and ResNet100 [16, 14], as in [8]. All experiments in this paper are implemented using MXNet [6]. The batch size is set to 512 and models are trained on eight NVIDIA Tesla P40 (24GB) GPUs. We employ the SGD optimizer and the learning rate starts from 0.1. We decrease the learning rate by $0.1\times$ at 10, 16, and 22 epochs, and stop at 24 epochs for all models. Memory feature injection starts from the 4th epoch, as the early phase of training is finished and sample features are relatively stable afterwards. The proposed memory module only consumes $\sim$ 22MB extra memory per GPU. Compared to ArcFace, the training speed of VPL slightly decreases from 1255 samples/second to 1252 samples/second. Therefore, both the extra memory and computation cost of the proposed VPL is negligible. To be aligned with previous work, all hyper-parameters are set by referring to the original papers [45, 8, 27, 19, 2]. For the model inference, we extract the 512-D embedding feature for each normalized face crop and employ cosine distance as the metric. To get the embedding features for templates (*e.g.* IJB-B and IJB-C), we simply calculate the feature center of all face images belonging to the template.

5.2. Ablation Study

Impact of the hyper-parameters λ and Δt: We first fix the accumulated iteration number as 100 and investigate the feature injection weight (λ in Eq. 4). In Tab. 2, we gradually increase the feature injection weight. Even though the original prototype plays the dominant role during training, the slight feature injection can significantly improve the performance. As $\lambda = 0.15$ achieves the best balance, we fix it for the following experiments. We further investigate the effect of Δt, which determines the maximum number of embedding features accumulated over past iterations and controls the feature injection ratio. As illustrated in Tab. 3, the TAR on IJB-C increases steadily from 96.21% to 96.61%, when the accumulated iterations increase from 0 to 100. Then, the performance can be maintained around 96.6% even though Δt is doubled. Without any compensation [25], there is only a slight performance degradation when Δt increases to 1000 and 2000, indicating the feature drift is extremely slow after the early phase of training. In the following experiments, we set $\Delta t = 100$ for training on MS1M, with the feature injection ratio being around 40%.

Orthogonal improvement to existing methods: Since the proposed VPL works by introducing sample-to-sample comparisons into prototype learning, we want to evaluate how it works with different loss functions. In particular, we implement the following state-of-the-art loss functions: Softmax loss with feature and prototype normalization [44] ($s = 64$), CosFace [45] ($m = 0.35$), ArcFace [8] ($m = 0.5$), AdaptiveFace [27] ($m = 0.4$ and $\lambda = 70$), and CurricularFace [19] ($\alpha = 0.99$). As shown in Tab. 4, the proposed VPL improves the discriminative feature embedding in all cases, indicating that VPL is orthogonal to existing margin-based and mining-based softmax losses.

Analysis of the improvement: As illustrated in Tab. 5, VPL significantly improves the performance of ArcFace from 70.99% to 73.91% on IFRT. More specifically, the accuracy on the Asian group greatly increases by 3.83%. To get a better understanding of the proposed VPL, we only apply feature injection to the Asian group, whose identity number accounts for 8.7% of the whole MS1M dataset. We set Δt as 100, with around 1.25% prototypes injected with features during training. In DBM [2], it is observed that tail domains are more sparse in the feature space, requiring adaptive margins to up-weight the loss. Even though 1.25% of prototypes are variational in our method, the performance significantly improves on both the "Asian" and "All" tracks, surpassing the DBM loss [2] ($\varepsilon = 5.5$) by 1.03% and 0.78%, respectively. In addition, we also conduct experiments by applying VPL to the 16.3K long-tail classes,

Method	Verification Accuracy					IJB		MegaFace	
	LFW	CFP-FP	CPLFW	AgeDB	CALFW	IJB-B	IJB-C	Id	Ver
CosFace(0.35) [45] (CVPR18)	99.81	98.12	92.28	98.11	95.76	94.80	96.37	97.91	97.91
ArcFace(0.5) [8] (CVPR19)	99.83	98.27	92.08	98.28	95.45	94.25	96.03	98.35	98.48
AFRN [22] (ICCV19)	99.85	95.56	93.48	95.35	96.30	88.5	93.0	-	-
MV-Softmax [50] (AAAI20)	99.80	98.28	92.83	97.95	96.10	93.6	95.2	97.76	97.80
GroupFace [24] (CVPR20)	99.85	98.63	93.17	98.28	96.20	94.93	96.26	98.74	98.79
CircleLoss [41] (CVPR20)	99.73	96.02	-	-	-	-	93.95	98.50	98.73
DUL [4] (CVPR20)	99.83	98.78	-	-	-	-	94.61	98.60	-
CurricularFace [19] (CVPR20)	99.80	98.37	93.13	98.32	96.20	94.8	96.1	98.71	98.64
URFace [39] (CVPR20)	99.78	98.64	-	-	-	-	96.6	-	-
DB [2] (CVPR20)	99.78	-	92.63	97.90	96.08	-	-	96.35	96.56
Sub-center ArcFace [7](ECCV20)	99.80	98.80	-	98.31	-	94.94	96.28	98.16	98.36
BroadFace [25] (ECCV20)	99.85	98.63	93.17	98.38	96.20	94.97	96.38	98.70	98.95
SST [11](ECCV20)	99.75	95.10	88.35	97.20	94.62	-	-	96.27	96.96
MS1M, R100, VPL-ArcFace	99.83	99.11	93.45	98.60	96.12	95.56	96.76	98.80	98.97

Table 6: Performance comparisons between the proposed VPL and state-of-the-art methods on various benchmarks. 1:1 verification accuracy (%) is reported on the LFW, CFP-FP, CPLFW, AgeDB, CALFW datasets. TAR@FAR=1e-4 is reported on the IJB-B and IJB-C datasets. Identification and verification evaluation on MegaFace using FaceScrub as the probe set. "Id" refers to the rank-1 face identification accuracy with 1M distractors, and "Ver" refers to the face verification TAR@FPR=1e-6.

whose image number is less than 20. Once again, significant improvement is observed when the long-tail prototypes are variational, with a feature injection ratio of around 1.12% during training. More specifically, VPL obtains 72.78% on the "All" track, outperforming AdaptiveFace [27] by 1.32%. Both the experimental results on the minority domain and long-tail data confirm the effectiveness of the proposed VPL on dealing with real-world unbalanced data.

5.3. Benchmark Results

To compare with recent state-of-the-art competitors, we train VPL-ArcFace models on MS1M [13, 10] with ResNet100 and test on various benchmarks, including LFW [18] for unconstrained face verification, CFP-FP [36] and CPLFW [57] for large pose variations, AgeDB [30] and CALFW [58] for age variations, IJB-B [51] and IJB-C [29] for mixed-media (image and video) face verification, and MegaFace [23] for identification and verification under million-scale distractors. As reported in Tab. 6, the proposed method achieves comparable result (*i.e*. 99.83%) with the competitors (*e.g*. AFRN [22], GroupFace [24] and BroadFace [25]) on LFW where the performance is almost saturated. For pose-invariant and age-invariant face recognition, our method achieves 99.11% on CFP-FP and 98.60% on AgeDB, outperforming all of the other state-of-the-art methods. Even though our method is not designed for set-based face recognition, VPL-ArcFace obtains 95.56% TAR on IJB-B and 96.76% TAR on IJB-C when FAR is set as 1e-4, showing superiority over the baselines and indicating that variational prototype learning can enhance the discriminative feature embedding as a generic approach. On MegaFace, we employ the ArcFace testing protocol, which manually refines both the probe and gallery set for the correct evaluation. Compared to the recent strong competitors (*e.g*. CurricularFace [19] and BroadFace [25]), the proposed VPL exhibits better performance again under both scenarios, achieving identification accuracy of 98.80% and verification accuracy of 98.97%. Both our method and SST [11] help to improve model training on long-tail data, but our method clearly outperforms SST by 2.53% on the MegaFace identification track, demonstrating the effectiveness of feature injection into the prototype learning. Even more, our method only needs to train one network instead of a pair of Semi-Siamese networks, as in SST.

6. Conclusions

In this work, we argue that existing prototype learning methods only employ sample-to-prototype comparisons without considering sample-to-sample comparisons during training, which impedes the SGD solver to find a better local-minimum. To this end, we propose the Variational Prototype Learning (VPL) which represents every class as a distribution, instead of a point in the latent space. Based on the observation of the slow feature drift phenomenon, we directly inject memorized features into prototypes to simulate variational prototype sampling. Without bells and whistles, the proposed VPL can be directly integrated into existing margin-based or mining-base softmax methods, and orthogonally improve the performance of deep face recognition. Extensive evaluation results on popular benchmarks demonstrate the superiority of the proposed VPL over the state-of-the-art competitors.

References

[1] Ognjen Arandjelovic, Gregory Shakhnarovich, John Fisher, Roberto Cipolla, and Trevor Darrell. Face recognition with image sets using manifold density divergence. In *CVPR*, 2005. 2

[2] Dong Cao, Xiangyu Zhu, Xingyu Huang, Jianzhu Guo, and Zhen Lei. Domain balancing: Face recognition on long-tailed domains. In *CVPR*, 2020. 7, 8

[3] Hakan Cevikalp and Bill Triggs. Face recognition based on image sets. In *CVPR*, 2010. 2

[4] Jie Chang, Zhonghao Lan, Changmao Cheng, and Yichen Wei. Data uncertainty learning in face recognition. In *CVPR*, 2020. 2, 8

[5] Binghui Chen, Weihong Deng, and Junping Du. Noisy softmax: improving the generalization ability of dcnn via postponing the early softmax saturation. In *CVPR*, 2017. 1

[6] Tianqi Chen, Mu Li, Yutian Li, Min Lin, Naiyan Wang, Minjie Wang, Tianjun Xiao, Bing Xu, Chiyuan Zhang, and Zheng Zhang. Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems. *arXiv:1512.01274*, 2015. 7

[7] Jiankang Deng, Jia Guo, Tongliang Liu, Mingming Gong, and Stefanos Zafeiriou. Sub-center arcface: Boosting face recognition by large-scale noisy web faces. In *ECCV*, 2020. 2, 8

[8] Jiankang Deng, Jia Guo, Xue Niannan, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *CVPR*, 2019. 1, 2, 3, 4, 5, 6, 7, 8

[9] Jiankang Deng, Jia Guo, Evangelos Ververas, Irene Kotsia, and Stefanos Zafeiriou. Retinaface: Single-shot multi-level face localisation in the wild. In *CVPR*, 2020. 7

[10] Jiankang Deng, Jia Guo, Debing Zhang, Yafeng Deng, Xiangju Lu, and Song Shi. Lightweight face recognition challenge. In *ICCV Workshop*, 2019. 4, 6, 8

[11] Hang Du, Hailin Shi, Yuchi Liu, Jun Wang, Zhen Lei, Dan Zeng, and Tao Mei. Semi-siamese training for shallow face learning. In *ECCV*, 2020. 2, 8

[12] Sixue Gong, Xiaoming Liu, and Anil K Jain. Jointly debiasing face recognition and demographic attribute estimation. *arXiv:1911.08080*, 2019. 6

[13] Yandong Guo, Lei Zhang, Yuxiao Hu, Xiaodong He, and Jianfeng Gao. Ms-celeb-1m: A dataset and benchmark for large-scale face recognition. In *ECCV*, 2016. 1, 4, 6, 8

[14] Dongyoon Han, Jiwhan Kim, and Junmo Kim. Deep pyramidal residual networks. *arXiv:1610.02915*, 2016. 7

[15] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, 2020. 3

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 7

[17] Xiaofei He, Shuicheng Yan, Yuxiao Hu, Partha Niyogi, and Hong-Jiang Zhang. Face recognition using laplacianfaces. *TPAMI*, 2005. 2

[18] 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. Technical report, 2007. 6, 8

[19] Yuge Huang, Yuhan Wang, Ying Tai, Xiaoming Liu, Pengcheng Shen, Shaoxin Li, Jilin Li, and Feiyue Huang. Curricularface: adaptive curriculum learning loss for deep face recognition. In *CVPR*, 2020. 1, 2, 5, 7, 8

[20] Zhiwu Huang, Ruiping Wang, Shiguang Shan, Xianqiu Li, and Xilin Chen. Log-euclidean metric learning on symmetric positive definite manifold with application to image set classification. In *ICML*, 2015. 2

[21] InsightFace. https://github.com/deepinsight/insightface/tree/master/challenges/IFRT. 6

[22] Bong-Nam Kang, Yonghyun Kim, Bongjin Jun, and Daijin Kim. Attentional feature-pair relation networks for accurate face recognition. In *ICCV*, 2019. 8

[23] Ira Kemelmacher-Shlizerman, Steven M Seitz, Daniel Miller, and Evan Brossard. The megaface benchmark: 1 million faces for recognition at scale. In *CVPR*, 2016. 6, 8

[24] Yonghyun Kim, Wonpyo Park, Myung-Cheol Roh, and Jongju Shin. Groupface: Learning latent groups and constructing group-based representations for face recognition. In *CVPR*, 2020. 8

[25] Yonghyun Kim, Wonpyo Park, and Jongju Shin. Broadface: Looking at tens of thousands of people at once for face recognition. In *ECCV*, 2020. 3, 4, 7, 8

[26] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv:1312.6114*, 2013. 2

[27] Hao Liu, Xiangyu Zhu, Zhen Lei, and Stan Z Li. Adaptiveface: Adaptive margin and sampling for face recognition. In *CVPR*, 2019. 2, 5, 7, 8

[28] Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song. Sphereface: Deep hypersphere embedding for face recognition. In *CVPR*, 2017. 1, 2, 5

[29] Brianna Maze, Jocelyn Adams, James A Duncan, Nathan Kalka, Tim Miller, Charles Otto, Anil K Jain, W Tyler Niggel, Janet Anderson, and Jordan Cheney. Iarpa janus benchmark–c: Face dataset and protocol. In *ICB*, 2018. 6, 8

[30] Stylianos Moschoglou, Athanasios Papaioannou, Christos Sagonas, Jiankang Deng, Irene Kotsia, and Stefanos Zafeiriou. Agedb: The first manually collected in-the-wild age database. In *CVPR Workshop*, 2017. 6, 8

[31] Aaron Nech and Ira Kemelmacher-Shlizerman. Level playing field for million scale face recognition. In *CVPR*, 2017. 1

[32] Hong-Wei Ng and Stefan Winkler. A data-driven approach to cleaning large face datasets. In *ICIP*, 2014. 6

[33] Hyun Oh Song, Yu Xiang, Stefanie Jegelka, and Silvio Savarese. Deep metric learning via lifted structured feature embedding. In *CVPR*, 2016. 1

[34] Oren Rippel, Manohar Paluri, Piotr Dollar, and Lubomir Bourdev. Metric learning with adaptive density discrimination. In *ICLR*, 2016. 1

[35] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *CVPR*, 2015. 1, 3

[36] Soumyadip Sengupta, Jun-Cheng Chen, Carlos Castillo, Vishal M Patel, Rama Chellappa, and David W Jacobs. Frontal to profile face verification in the wild. In *WACV*, 2016. 6, 8

[37] Gregory Shakhnarovich, John W Fisher, and Trevor Darrell. Face recognition from long-term observations. In *ECCV*, 2002. 2

[38] Yichun Shi and Anil K Jain. Probabilistic face embeddings. In *ICCV*, 2019. 2

[39] Yichun Shi, Xiang Yu, Kihyuk Sohn, Manmohan Chandraker, and Anil K Jain. Towards universal representation learning for deep face recognition. In *CVPR*, 2020. 8

[40] Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. In *NeurIPS*, 2016. 1, 3

[41] Yifan Sun, Changmao Cheng, Yuhan Zhang, Chi Zhang, Liang Zheng, Zhongdao Wang, and Yichen Wei. Circle loss: A unified perspective of pair similarity optimization. In *CVPR*, 2020. 1, 2, 3, 5, 8

[42] Matthew A Turk and Alex P Pentland. Face recognition using eigenfaces. In *CVPR*, 1991. 2

[43] Feng Wang, Weiyang Liu, Haijun Liu, and Jian Cheng. Additive margin softmax for face verification. *SPL*, 2018. 1, 5

[44] Feng Wang, Xiang Xiang, Jian Cheng, and Alan L Yuille. Normface: l_2 hypersphere embedding for face verification. *arXiv:1704.06369*, 2017. 7

[45] Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Zhifeng Li, Dihong Gong, Jingchao Zhou, and Wei Liu. Cosface: Large margin cosine loss for deep face recognition. In *CVPR*, 2018. 1, 2, 5, 7, 8

[46] Mei Wang and Weihong Deng. Mitigating bias in face recognition using skewness-aware reinforcement learning. In *CVPR*, 2020. 6

[47] Mei Wang, Weihong Deng, Jiani Hu, Xunqiang Tao, and Yaohai Huang. Racial faces in the wild: Reducing racial bias by information maximization adaptation network. In *ICCV*, 2019. 6

[48] Xinshao Wang, Yang Hua, Elyor Kodirov, Guosheng Hu, Romain Garnier, and Neil M Robertson. Ranked list loss for deep metric learning. In *CVPR*, 2019. 2

[49] Xun Wang, Haozhi Zhang, Weilin Huang, and Matthew R Scott. Cross-batch memory for embedding learning. In *CVPR*, 2020. 3

[50] Xiaobo Wang, Shifeng Zhang, Shuo Wang, Tianyu Fu, Hailin Shi, and Tao Mei. Mis-classified vector guided softmax loss for face recognition. In *AAAI*, 2020. 8

[51] Cameron Whitelam, Emma Taborsky, Austin Blanton, Brianna Maze, Jocelyn C Adams, Tim Miller, Nathan D Kalka, Anil K Jain, James A Duncan, and Kristen Allen. Iarpa janus benchmark-b face dataset. In *CVPR Workshop*, 2017. 6, 8

[52] Chao-Yuan Wu, R Manmatha, Alexander J Smola, and Philipp Krahenbuhl. Sampling matters in deep embedding learning. In *ICCV*, 2017. 1

[53] Zhirong Wu, Alexei A Efros, and Stella X Yu. Improving generalization via scalable neighborhood component analysis. In *ECCV*, 2018. 3

[54] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *CVPR*, 2018. 3

[55] Tian Xu, Jennifer White, Sinan Kalkan, and Hatice Gunes. Investigating bias and fairness in facial expression recognition. *arXiv:2007.10075*, 2020. 6

[56] Xiao Zhang, Zhiyuan Fang, Yandong Wen, Zhifeng Li, and Yu Qiao. Range loss for deep face recognition with long-tail. In *ICCV*, 2017. 2

[57] Tianyue Zheng and Weihong Deng. Cross-pose lfw: A database for studying cross-pose face recognition in unconstrained environments. Technical report, 2018. 6, 8

[58] Tianyue Zheng, Weihong Deng, and Jiani Hu. Cross-age lfw: A database for studying cross-age face recognition in unconstrained environments. *arXiv:1708.08197*, 2017. 6, 8

[59] Yaoyao Zhong, Weihong Deng, Mei Wang, Jiani Hu, Jianteng Peng, Xunqiang Tao, and Yaohai Huang. Unequal-training for deep face recognition with long-tailed noisy data. In *CVPR*, 2019. 2

[60] Zhun Zhong, Liang Zheng, Zhiming Luo, Shaozi Li, and Yi Yang. Invariance matters: Exemplar memory for domain adaptive person re-identification. In *CVPR*, 2019. 3

[61] Xiangyu Zhu, Hao Liu, Zhen Lei, Hailin Shi, Fan Yang, Dong Yi, Guojun Qi, and Stan Z Li. Large-scale bisample learning on id versus spot face recognition. *IJCV*, 2019. 2