Probabilistic Face Embeddings

Yichun Shi[1], Anil K. Jain[1], Nathan D. Kalka[2]
[1]Michigan State University, East Lansing, MI
[2]Noblis, Bridgeport, WV
shiyichu@msu.edu, jain@cse.msu.edu, nathan.kalka@noblis.org

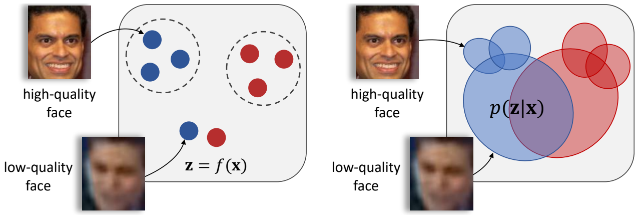
Abstract

Embedding methods have achieved success in face recognition by comparing facial features in a latent semantic space. However, in a fully unconstrained face setting, the features learned by the embedding model could be ambiguous or may not even be present in the input face, leading to noisy representations. We propose Probabilistic Face Embeddings (PFEs), which represent each face image as a Gaussian distribution in the latent space. The mean of the distribution estimates the most likely feature values while the variance shows the uncertainty in the feature values. Probabilistic solutions can then be naturally derived for matching and fusing PFEs using the uncertainty information. Empirical evaluation on different baseline models, training datasets and benchmarks show that the proposed method can improve the face recognition performance of deterministic embeddings by converting them into PFEs. The uncertainties estimated by PFEs also serve as good indicators of the potential matching accuracy, which are important for a risk-controlled recognition system.

1. Introduction

When humans are asked to describe a face image, they not only give the description of the facial attributes, but also the confidence associated with them. For example, if the eyes are blurred in the image, a person will keep the eye size as an uncertain information and focus on other features. Furthermore, if the image is completely corrupted and no attributes can be discerned, the subject may respond that he/her cannot identify this face. This kind of uncertainty (or confidence) estimation is common and important in human decision making.

On the other hand, the representations used in state-of-the-art face recognition systems are generally confidence-agnostic. These methods depend on an embedding model (*e.g*. Deep Neural Networks) to give a deterministic point representation for each face image in the latent feature space [31, 40, 23, 39, 5]. A point in the latent space represents the model's estimation of the facial features in the given image. If the error in the estimation is somehow bounded, the distance between two points can effectively measure the semantic similarity between the corresponding face images. But given a low-quality input, where the expected facial features are ambiguous or absent in the image, a large shift in the embedded points is inevitable, leading to false recognition (Figure 1a).

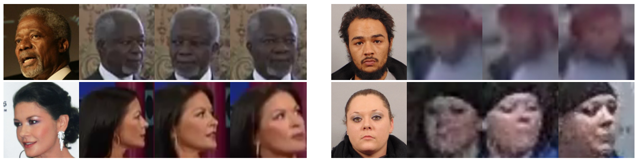


(a) deterministic embedding (b) probabilistic embedding

Figure 1: Difference between deterministic face embeddings and probabilistic face embeddings (PFEs). Deterministic embeddings represent every face as a point in the latent space without regards to its feature ambiguity. Probabilistic face embedding (PFE) gives a distributional estimation of features in the latent space instead. **Best viewed in color.**

Given that face recognition systems have already achieved high recognition accuracies on relatively constrained face recognition benchmarks, *e.g*. LFW [11] and YTF [42], where most facial attributes can be clearly observed, recent face recognition challenges have moved on to more unconstrained scenarios, including surveillance videos [21, 26, 15] (See Figure 2). In these tasks, any type and degree of variation could exist in the face image, where most of the desired facial features learned by the representation model could be absent. Given this lack of information, it is unlikely to find a feature set that could always match these faces accurately. Hence state-of-the-art face recognition systems which obtained over 99% accuracy on LFW have suffered from a large performance drop on IARPA Janus benchmarks [21, 26, 15].

To address the above problems, we propose *Probabilistic Face Embeddings (PFEs)*, which give a distributional estimation instead of a point estimation in the latent space for each input face image (Figure 1b). The mean of the distri-



(a) IJB-A [21] (b) IJB-S [15]

Figure 2: Example images from IJB-A and IJB-S. The first columns show still images, followed by video frames of the respective subjects in the next three columns. These benchmarks present a more unconstrained recognition scenario where there is a large variability in the image quality.

bution can be interpreted as the most likely latent feature values while the span of the distribution represents the uncertainty of these estimations. PFE can address the unconstrained face recognition problem in a two-fold way: (1) During matching (face comparison), PFE penalizes uncertain features (dimensions) and pays more attention to more confident features. (2) For low quality inputs, the confidence estimated by PFE can be used to reject the input or actively ask for human assistance to avoid false recognition. Besides, a natural solution can be derived to aggregate the PFE representations of a set of face images into a new distribution with lower uncertainty to increase the recognition performance. The implementation of PFE is open-sourced[1]. The contributions of the paper can be summarized as below:

1. An uncertainty-aware probabilistic face embedding (PFE) which represents face images as distributions instead of points.
2. A probabilistic framework that can be naturally derived for face matching and feature fusion using PFE.
3. A simple method that converts existing deterministic embeddings into PFEs without additional training data.
4. Comprehensive experiments showing that the proposed PFE can improve face recognition performance of deterministic embeddings and can effectively filter out low-quality inputs to enhance the robustness of face recognition systems.

2. Related Work

Uncertainty Learning in DNNs To improve the robustness and interpretability of discriminant Deep Neural Networks (DNNs), deep uncertainty learning is getting more attention [17, 6, 18]. There are two main types of uncertainty: *model uncertainty* and *data uncertainty*. Model uncertainty refers to the uncertainty of model parameters given the training data and can be reduced by collecting additional training data [25, 27, 17, 6]. Data uncertainty accounts for the uncertainty in output whose primary source is the inherent noise in input data and hence cannot be eliminated with more training data [18]. The uncertainty studied in our work can be categorized as data uncertainty. Techniques have been developed for estimating data uncertainty in different tasks, including classification and regression [18], where the target space is explicitly defined by labels. In contrast, probabilistic embeddings aim to estimate the uncertainty of the representations in latent spaces [20, 37, 1, 28]. Specific to face recognition, some studies [7, 19, 51] have leveraged the model uncertainty for analysis and learning of face representations, but to our knowledge, ours is the first work that utilizes data uncertainty[2] for face recognition.

Probabilistic Face Representation Modeling faces as probabilistic distributions is not a new idea. In the field of face template/video matching, there exists abundant literature on modeling the faces as probabilistic distributions [33, 2], subspace [4] or manifolds [2, 12] in the feature space. However, the input for such methods is a set of face images rather than a single face image, and they use a between-distribution similarity or distance measure, *e.g.* KL-divergence, for comparison, which does not penalize the uncertainty. Meanwhile, some studies [22, 10] have attempted to build a fuzzy model of a given face using the features of face parts. In comparison, the proposed PFE represents each single face image as a distribution in the latent space encoded by DNNs and we use an uncertainty-aware log likelihood score to compare the distributions.

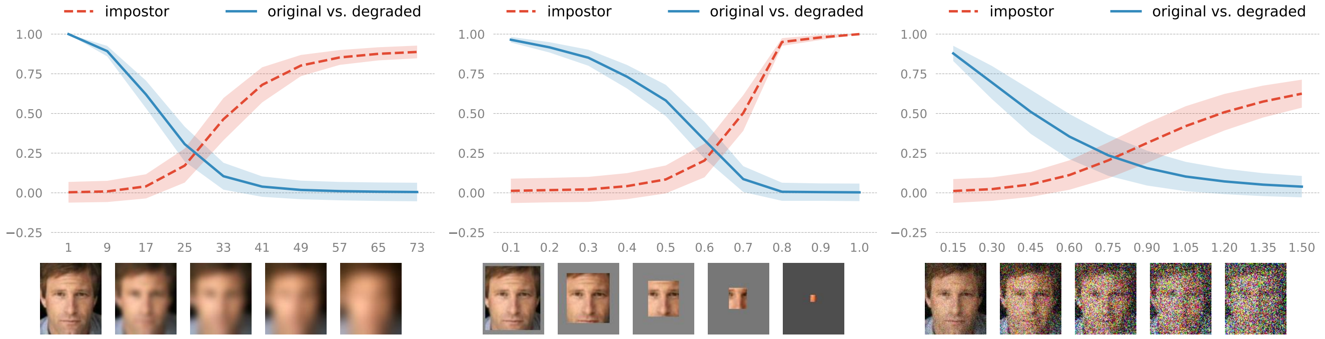
Quality-aware Pooling In contrast to the methods above, recent work on face template/video matching aims to leverage the saliency of deep CNN embeddings by aggregating the deep features of all faces into a single compact vector [47, 24, 45, 8]. In these methods, a separate module learns to predict the quality of each face in the image set, which is then normalized for a weighted pooling of feature vectors. We show that a solution can be naturally derived under our framework, which not only gives a probabilistic explanation for quality-aware pooling methods, but also leads to a more general solution where an image set can also be modeled as a PFE representation.

3. Limitations of Deterministic Embeddings

In this section, we explain the problems of deterministic face embeddings from both theoretical and empirical views. Let $\mathcal{X}$ denote the image space and $\mathcal{Z}$ denote the latent feature space of D dimensions. An ideal latent space $\mathcal{Z}$ should only encode *identity-salient* features and be *disentangled* from identity-irrelevant features. As such, each identity should have a unique intrinsic code $\mathbf{z} \in \mathcal{Z}$ that best represents this person and each face image $\mathbf{x} \in \mathcal{X}$ is an observation sampled from $p(\mathbf{x}|\mathbf{z})$. The process of training face embeddings can be viewed as a joint process of searching for such a latent space $\mathcal{Z}$ and learning the inverse mapping

[1] https://github.com/seasonSH/Probabilistic-Face-Embeddings

[2] Some in the literature have also used the terminology "data uncertainty" for a different purpose [46].



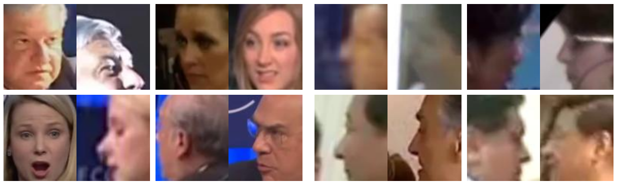
(a) Gaussian Blur (b) Occlusion (c) Random Gaussian Noise

Figure 3: Illustration of *feature ambiguity dilemma*. The plots show the cosine similarity on LFW dataset with different degrees of degradation. Blue lines show the similarity between original images and their respective degraded versions. Red lines show the similarity between impostor pairs of degraded images. The shading indicates the standard deviation. With larger degrees of degradation, the model becomes more confident (very high/low scores) in a wrong way.

$p(\mathbf{z}|\mathbf{x})$. For deterministic embeddings, the inverse mapping is a Dirac delta function $p(\mathbf{z}|\mathbf{x}) = \delta(\mathbf{z} - f(\mathbf{x}))$, where f is the embedding function. Clearly, for any space $\mathcal{Z}$, given the possibility of noises in $\mathbf{x}$, it is unrealistic to recover the exact $\mathbf{z}$ and the embedded point of a low-quality input would inevitably shift away from its intrinsic $\mathbf{z}$ (no matter how much training data we have).

The question is whether this shift could be bounded such that the we still have smaller intra-class distances compared to inter-class distances. However, this is unrealistic for fully unconstrained face recognition and we conduct an experiment to illustrate this. Let us start with a simple example: given a pair of identical images, a deterministic embedding will always map them to the same point and therefore the distance between them will always be 0, even if these images do not contain a face. This implies that "a pair of images being similar or even the same does not necessarily mean the probability of their belonging to the same person is high".

To demonstrate this, we conduct an experiment by manually degrading the high-quality images and visualizing their similarity scores. We randomly select a high-quality image of each subject from the LFW dataset [11] and manually insert Gaussian blur, occlusion, and random Gaussian noise to the faces. In particular, we linearly increase the size of Gaussian kernel, occlusion ratio and the standard deviation of the noise to control the degradation degree. At each degradation level, we extract the feature vectors with a 64-layer CNN[3], which is comparable to state-of-the-art face recognition systems. The features are normalized to a hyper-spherical embedding space. Then, two types of cosine similarities are reported: (1) similarity between pairs of original image and its respective degraded image, and (2) similarity between degraded images of different identities. As shown in Figure 3, for all the three types of degradation, the genuine similarity scores decrease to 0 while the impostor similarity scores converge to 1.0! These indicate two types of errors that can be expected in a fully unconstrained scenario even when the model is very confident (very high/low similarity scores):



(a) Low-similarity Genuine Pairs (b) High-similarity Impostor Pairs

Figure 4: Example genuine pairs from IJB-A dataset estimated with the lowest similarity scores and impostor pairs with the highest similarity scores (among all possible pairs) by a 64-layer CNN model. The genuine pairs mostly consist of one high-quality and one low-quality image while the impostor pairs are all low-quality images. Note that these pairs are not templates in the verification protocol.

(1) false accept of impostor low-quality pairs and

(2) false reject of genuine cross-quality pairs.

To confirm this, we test the model on the IJB-A dataset by finding impostor/genuine image pairs with the highest/lowest scores, respectively. The situation is exactly as we hypothesized (See Figure 4). We call this *Feature Ambiguity Dilemma* which is observed when the deterministic embeddings are forced to estimate the features of ambiguous faces. The experiment also implies that there exist a *dark space* where the ambiguous inputs are mapped to and the distance metric is distorted.

4. Probabilistic Face Embeddings

To address the aforementioned problem caused by data uncertainty, we propose to encode the uncertainty into the face representation and take it into account during matching. Specifically, instead of building a model that gives a point estimation in the latent space, we estimate a distribution $p(\mathbf{z}|\mathbf{x})$ in the latent space to represent the potential

[3] trained on Ms-Celeb-1M [9] with AM-Softmax [38]

appearance of a person's face[4]. In particular, we use a multivariate Gaussian distribution:

$$p(\mathbf{z}|\mathbf{x}_i) = \mathcal{N}(\mathbf{z}; \boldsymbol{\mu}_i, \boldsymbol{\sigma}_i^2\mathbf{I}) \tag{1}$$

where $\boldsymbol{\mu}_i$ and $\boldsymbol{\sigma}_i$ are both a D-dimensional vector predicted by the network from the i^{th} input image $\mathbf{x}_i$. Here we only consider a diagonal covariance matrix to reduce the complexity of the face representation. This representation should have the following properties:

1. The center $\boldsymbol{\mu}$ should encode the most likely facial features of the input image.
2. The uncertainty $\boldsymbol{\sigma}$ should encode the model's confidence along each feature dimension.

In addition, we wish to use a single network to predict the distribution. Considering that new approaches for training face embeddings are still being developed, we aim to develop a method that could convert existing deterministic face embedding networks to PFEs in an easy manner. In the followings, we first show how to compare and fuse the PFE representations to demonstrate their strength and then propose our method for learning PFEs.

Note 1. *Because of the space limit, we provide the proofs of all the propositions below in the Supplementary Material.*

4.1. Matching with PFEs

Given the PFE representations of a pair of images $(\mathbf{x}_i, \mathbf{x}_j)$, we can directly measure the "likelihood" of them belonging to the same person (sharing the same latent code): $p(\mathbf{z}_i = \mathbf{z}_j)$, where $\mathbf{z}_i \sim p(\mathbf{z}|\mathbf{x}_i)$ and $\mathbf{z}_j \sim p(\mathbf{z}|\mathbf{x}_j)$. Specifically,

$$p(\mathbf{z}_i = \mathbf{z}_j) = \int p(\mathbf{z}_i|\mathbf{x}_i)p(\mathbf{z}_j|\mathbf{x}_j)\delta(\mathbf{z}_i - \mathbf{z}_j)d\mathbf{z}_i d\mathbf{z}_j. \tag{2}$$

In practice, we would like to use the log likelihood instead, whose solution is given by:

$$\begin{aligned} s(\mathbf{x}_i, \mathbf{x}_j) =& \log p(\mathbf{z}_i = \mathbf{z}_j) \\ =& -\frac{1}{2}\sum_{l=1}^{D}\left(\frac{(\mu_i^{(l)} - \mu_j^{(l)})^2}{\sigma_i^{2(l)} + \sigma_j^{2(l)}} + \log(\sigma_i^{2(l)} + \sigma_j^{2(l)})\right) \\ & - const, \end{aligned} \tag{3}$$

where $const = \frac{D}{2}\log 2\pi$, $\mu_i^{(l)}$ refers to the l^{th} dimension of $\boldsymbol{\mu}_i$ and similarly for $\sigma_i^{(l)}$.

Note that this symmetric measure can be viewed as the expectation of likelihood of one input's latent code conditioned on the other, that is

$$\begin{aligned} s(\mathbf{x}_i, \mathbf{x}_j) &= \log \int p(\mathbf{z}|\mathbf{x}_i)p(\mathbf{z}|\mathbf{x}_j)d\mathbf{z} \\ &= \log \mathbb{E}_{\mathbf{z}\sim p(\mathbf{z}|\mathbf{x}_i)}[p(\mathbf{z}|\mathbf{x}_j)] \\ &= \log \mathbb{E}_{\mathbf{z}\sim p(\mathbf{z}|\mathbf{x}_j)}[p(\mathbf{z}|\mathbf{x}_i)]. \end{aligned} \tag{4}$$

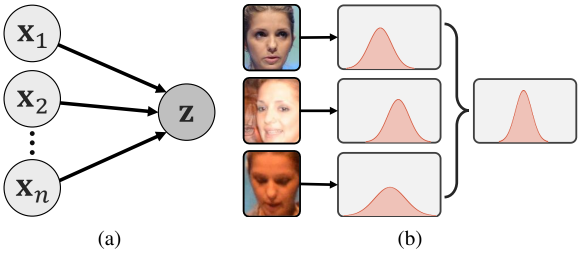


Figure 5: Fusion with PFEs. (a) Illustration of the fusion process as a directed graphical model. (b) Given the Gaussian representations of faces (from the same identity), the fusion process outputs a new Gaussian distribution in the latent space with a more precise mean and lower uncertainty.

As such, we call it *mutual likelihood score (MLS)*[5]. Different from KL-divergence, this score is unbounded and cannot be seen as a distance metric. It can be shown that the squared Euclidean distance is equivalent to a special case of MLS when all the uncertainties are assumed to be the same:

Property 1. *If $\sigma_i^{(l)}$ is a fixed number for all data $\mathbf{x}_i$ and dimensions l, MLS is equivalent to a scaled and shifted negative squared Euclidean distance.*

Further, when the uncertainties are allowed to be different, we note that MLS has some interesting properties that make it different from a distance metric:

1. *Attention* mechanism: the first term in the bracket in Equation (3) can be seen as a weighted distance which assigns larger weights to less uncertain dimensions.
2. *Penalty* mechanism: the second term in the bracket in Equation (3) can be seen as a penalty term which penalizes dimensions that have high uncertainties.
3. If either input $\mathbf{x}_i$ or $\mathbf{x}_i$ has large uncertainties, MLS will be low (because of penalty) irrespective of the distance between their mean.
4. Only if both inputs have small uncertainties and their means are close to each other, MLS could be very high.

The last two properties imply that PFE could solve the feature ambiguity dilemma if the network can effectively estimate $\boldsymbol{\sigma}_i$.

4.2. Fusion with PFEs

In many cases we have a template (set) of face images, for which we need to build a compact representation for matching. With PFEs, a conjugate formula can be derived for representation fusion (Figure 5). Let $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$ be a series of observations (face images) from the same identity and $p(\mathbf{z}|\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n)$ be the posterior distribution after the n^{th} observation. Then, assuming all the observations are conditionally independent (given the latent code $\mathbf{z}$). It can be shown that:

$$p(\mathbf{z}|\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_{n+1}) = \alpha\frac{p(\mathbf{z}|\mathbf{x}_{n+1})}{p(\mathbf{z})}p(\mathbf{z}|\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n), \tag{5}$$

[4] following the notations in Section 3.

[5] also known as expected likelihood or probability product kernel [14].

where α is a normalization factor. To simplify the notations, let us only consider a one-dimensional case below; the solution can be easily extended to the multivariate case.

If $p(\mathbf{z})$ is assumed to be a noninformative prior, *i.e.* $p(\mathbf{z})$ is a Gaussian distribution whose variance approaches ∞, the posterior distribution in Equation (5) is a new Gaussian distribution with lower uncertainty (See supplementary material). Further, given a set of face images $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$, the parameters of the fused representation can be directly given by:

$$\hat{\mu}_n = \sum_{i=1}^{n} \frac{\hat{\sigma}_n^2}{\sigma_i^2}\mu_i, \tag{6}$$

$$\frac{1}{\hat{\sigma}_n^2} = \sum_{i=1}^{n} \frac{1}{\sigma_i^2}. \tag{7}$$

In practice, because the conditional independence assumption is usually not true, *e.g.* video frames include a large amount of redundancy, Equation (7) will be biased by the number of images in the set. Therefore, we take dimension-wise minimum to obtain the new uncertainty,

Relationship to Quality-aware Pooling If we consider a case where all the dimensions share the same uncertainty σ_i for i^{th} input and let the quality value $q_i = \frac{1}{\sigma_i^2}$ be the output of the network. Then Equation (6) can be written as

$$\hat{\boldsymbol{\mu}}_n = \frac{\sum_{i=1}^{n} q_i \boldsymbol{\mu}_i}{\sum_{j}^{n} q_j}. \tag{8}$$

If we do not use the uncertainty after fusion, the algorithm will be the same as recent quality-aware aggregation methods for set-to-set face recognition [47, 24, 45].

4.3. Learning

Note that any deterministic embedding f, if properly optimized, can indeed satisfy the properties of PFEs: (1) the embedding space is a disentangled identity-salient latent space and (2) $f(\mathbf{x})$ represents the most likely features of the given input in the latent space. As such, in this work we consider a stage-wise training strategy: given a pre-trained embedding model f, we fix its parameters, take $\boldsymbol{\mu}(\mathbf{x}) = f(\mathbf{x})$, and optimize an additional uncertainty module to estimate $\boldsymbol{\sigma}(\mathbf{x})$. When the uncertainty module is trained on the same dataset of the embedding model, this stage-wise training strategy allows us to have a more fair comparison between PFE and the original embedding $f(\mathbf{x})$ than an end-to-end learning strategy.

The uncertainty module is a network with two fully-connected layers which shares the same input as of the bottleneck layer[6]. The optimization criteria is to maximize the mutual likelihood score of all genuine pairs $(\mathbf{x}_i, \mathbf{x}_j)$. Formally, the loss function to minimize is

$$\mathcal{L} = \frac{1}{|\mathcal{P}|}\sum_{(i,j)\in\mathcal{P}} -s(\mathbf{x}_i, \mathbf{x}_j) \tag{9}$$

where $\mathcal{P}$ is the set of all genuine pairs and s is defined in Equation (3). In practice, the loss function is optimized within each mini-batch. Intuitively, this loss function can be understood as an alternative to maximizing $p(\mathbf{z}|\mathbf{x})$: if the latent distributions of all possible genuine pairs have a large overlap, the latent target $\mathbf{z}$ should have a large likelihood $p(\mathbf{z}|\mathbf{x})$ for any corresponding $\mathbf{x}$. Notice that because $\boldsymbol{\mu}(\mathbf{x})$ is fixed, the optimization wouldn't lead to the collapse of all the $\boldsymbol{\mu}(\mathbf{x})$ to a single point.

5. Experiments

In this section, we first test the proposed PFE method on standard face recognition protocols to compare with deterministic embeddings. Then we conduct qualitative analysis to gain more insight into how PFE behaves. *Due to the space limit, we provide the implementation details in the supplementary material.*

To comprehensively evaluate the efficacy of PFEs, we conduct the experiments on 7 benchmarks, including the well known **LFW** [11], **YTF** [42], **MegaFace** [16] and four other more unconstrained benchmarks:
CFP [32] contains $7,000$ frontal/profile face photos of 500 subjects. We only test on the frontal-profile (FP) protocol, which includes $7,000$ pairs of frontal-profile faces.
IJB-A [21] is a template-based benchmark, containing $25,813$ faces images of 500 subjects. Each template includes a set of still photos or video frames. Compared with previous benchmarks, the faces in IJB-A have larger variations and present a more unconstrained scenario.
IJB-C [26] is an extension of IJB-A with $140,740$ faces images of $3,531$ subjects. The verification protocol of IJB-C includes more impostor pairs so we can compute True Accept Rates (TAR) at lower False Accept Rates (FAR).
IJB-S [15] is a surveillance video benchmark containing 350 surveillance videos spanning 30 hours in total, $5,656$ enrollment images, and 202 enrollment videos of 202 subjects. Many faces in this dataset are of extreme pose or low-quality, making it one of the most challenging face recognition benchmarks (See Figure 2 for example images).

We use the CASIA-WebFace [48] and a cleaned version[7] of MS-Celeb-1M [9] as training data, from which we remove the subjects that are also included in the test datasets[8].

[6] the layer which outputs the original face embedding.

[7] https://github.com/inlmouse/MS-Celeb-1M_WashList.

[8] 84 and $4,182$ subjects were removed from CASIA-WebFace and MS-Celeb-1M, respectively.

Base Model	Representation	LFW	YTF	CFP-FP	IJB-A
Softmax + Center Loss [40]	Original	98.93	94.74	93.84	78.16
	PFE	**99.27**	**95.42**	**94.51**	**80.83**
Triplet [31]	Original	97.65	93.36	89.76	60.82
	PFE	**98.45**	**93.96**	**90.04**	**61.00**
A-Softmax [23]	Original	99.15	94.80	92.41	78.54
	PFE	**99.32**	**94.94**	**93.37**	**82.58**
AM-Softmax [38]	Original	99.28	95.64	94.77	84.69
	PFE	**99.55**	**95.92**	**95.06**	**87.58**

Table 1: Results of models trained on CASIA-WebFace. "Orignal" refers to the deterministic embeddings. The better performance among each base model are shown in bold numbers. "PFE" uses mutual likelihood score for matching. IJB-A results are verification rates at FAR=0.1%.

Method	Training Data	LFW	YTF	MF1 Rank1	MF1 Veri.
DeepFace+ [35]	4M	97.35	91.4	-	-
FaceNet [31]	200M	99.63	95.1	-	-
DeepID2+ [34]	300K	99.47	93.2	-	-
CenterFace [40]	0.7M	99.28	94.9	65.23	76.52
SphereFace [23]	0.5M	99.42	95.0	75.77	89.14
ArcFace [5]	5.8M	99.83	98.02	81.03	96.98
CosFace [39]	5M	99.73	97.6	77.11	89.88
L2-Face [29]	3.7M	99.78	96.08	-	-
Baseline	4.4M	99.70	97.18	79.43	92.93
PFE$_{\text{fuse}}$	4.4M	-	97.32	-	-
PFE$_{\text{fuse+match}}$	4.4M	99.82	97.36	78.95	92.51

Table 2: Results of our models (last three rows) trained on MS-Celeb-1M and state-of-the-art methods on LFW, YTF and MegaFace. The MegaFace verification rates are computed at FAR=0.0001%. "-" indicates that the author did report the performance on the corresponding protocol.

5.1. Experiments on Different Base Embeddings

Since our method works by converting existing deterministic embeddings, we want to evaluate how it works with different base embeddings, *i.e.* face representations trained with different loss functions. In particular, we implement the following state-of-the-art loss functions: Softmax+Center Loss [40], Triplet Loss [31], A-Softmax [23] and AM-Softmax [38][9]. To be aligned with previous work [23, 39], we train a 64-layer residual network [23] with each of these loss functions on the CASIA-WebFace dataset as base models. All the features are $\ell 2$-normalized to a hyper-spherical embedding space. Then we train the uncertainty module for each base model on the CASIA-WebFace again for $3,000$ steps. We evaluate the performance on four benchmarks: LFW [11], YTF [42], CFP-FP [32] and IJB-A [21], which present different challenges in face recognition. The results are shown in Table 1. The PFE improves over the original representation in all cases, indicating the proposed method is robust with different embeddings and testing scenarios.

5.2. Comparison with State-Of-The-Art

To compare with state-of-the-art face recognition methods, we use a different base model, which is a 64-layer network trained with AM-Softmax on the MS-Celeb-1M dataset. Then we fix the parameters and train the uncertainty module on the same dataset for $12,000$ steps. In the following experiments, we compare 3 methods:

Method	Training Data	IJB-A (TAR@FAR) 0.1%	IJB-A (TAR@FAR) 1.0%	CFP-FP
DR-GAN [36]	1M	53.9 ± 4.3	77.4 ± 2.7	93.41
Yin *et al.* [50]	0.5M	73.9 ± 4.2	77.5 ± 2.5	**94.39**
TPE [30]	0.5M	90.0 ± 1.0	93.4 ± 0.5	89.17
NAN [47]	3M	88.1 ± 1.1	94.1 ± 0.8	-
QAN [24]	5M	89.31 ± 3.92	94.20 ± 1.53	-
Cao *et al.* [3]	3.3M	90.4 ± 1.4	95.8 ± 0.6	-
Multicolumn [45]	3.3M	92.0 ± 1.3	96.2 ± 0.5	-
L2-Face [29]	3.7M	94.3 ± 0.5	97.00 ± 0.4	-
Baseline	4.4M	93.30 ± 1.29	96.15 ± 0.71	92.78
PFE$_{\text{fuse}}$	4.4M	94.59 ± 0.72	95.92 ± 0.73	-
PFE$_{\text{fuse+match}}$	4.4M	**95.25 ± 0.89**	**97.50 ± 0.43**	93.34

Table 3: Results of our models (last three rows) trained on MS-Celeb-1M and state-of-the-art methods on CFP (frontal-profile protocol) and IJB-A.

Method	Training Data	IJB-C (TAR@FAR) 0.001%	IJB-C (TAR@FAR) 0.01%	IJB-C (TAR@FAR) 0.1%	IJB-C (TAR@FAR) 1%
Yin *et al.* [49]	0.5M	-	-	69.3	83.8
Cao *et al.* [3]	3.3M	74.7	84.0	91.0	96.0
Multicolumn [45]	3.3M	77.1	86.2	92.7	96.8
DCN [44]	3.3M	-	88.5	94.7	**98.3**
Baseline	4.4M	70.10	85.37	93.61	96.91
PFE$_{\text{fuse}}$	4.4M	83.14	92.38	95.47	97.36
PFE$_{\text{fuse+match}}$	4.4M	**89.64**	**93.25**	**95.49**	97.17

Table 4: Results of our models (last three rows) trained on MS-Celeb-1M and state-of-the-art methods on IJB-C.

- **Baseline** only uses the original features of the 64-layer deterministic embedding along with cosine similarity for matching. Average pooling is used in case of template/video benchmarks.
- **PFE$_{\text{fuse}}$** uses the uncertainty estimation $\boldsymbol{\sigma}$ in PFE and Equation (6) to aggregate the features of templates but uses cosine similarity for matching. If the uncertainty module could estimate the feature uncertainty effectively, fusion with $\boldsymbol{\sigma}$ should be able to outperform average pooling by assigning larger weights to confident features.
- **PFE$_{\text{fuse+match}}$** uses $\boldsymbol{\sigma}$ both for fusion and matching (with mutual likelihood scores). Templates/videos are fused based on Equation (6) and Equation (7).

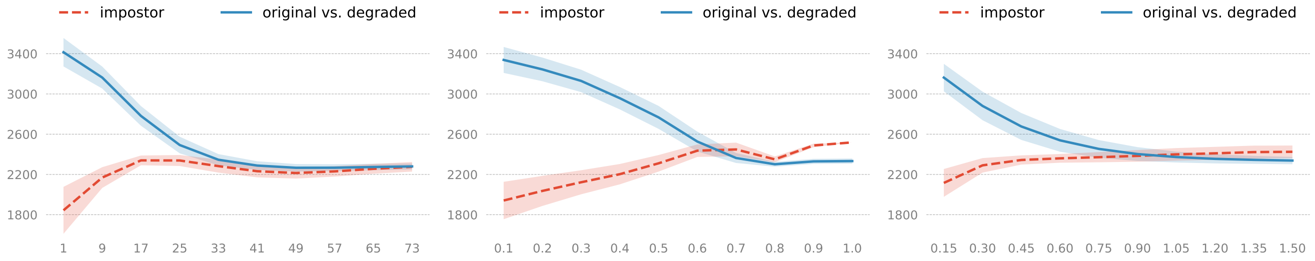
In Table 2 we show the results on three relatively easier benchmarks: LFW, YTF and MegaFace. Although the accuracy on LFW and YTF are nearly saturated, the proposed PFE still improves the performance of the original representation. Note that MegaFace is a biased dataset: because all the probes are high-quality images from FaceScrub, the positive pairs in MegaFace are both high-quality images while the negative pairs only contain at most one low-quality image[10] . Therefore, neither of the two types of error caused

[9] We also tried implementing ArcFace [5] but it does not converge well in our case. So we did not use it.

[10] The negative pairs of MegaFace in the verification protocol only in-

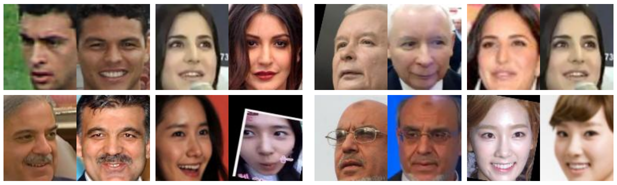
Method	Training Data	Surveillance-to-Single					Surveillance-to-Booking					Surveillance-to-Surveillance				
		Rank-1	Rank-5	Rank-10	1%	10%	Rank-1	Rank-5	Rank-10	1%	10%	Rank-1	Rank-5	Rank-10	1%	10%
C-FAN [8]	5.0M	50.82	61.16	64.95	16.44	24.19	53.04	62.67	66.35	27.40	29.70	**10.05**	17.55	21.06	0.11	0.68
Baseline	4.4M	50.00	59.07	62.70	7.22	19.05	47.54	56.14	61.08	14.75	22.99	9.40	17.52	23.04	0.06	0.71
PFE$_{\text{fuse}}$	4.4M	**53.44**	**61.40**	**65.05**	10.53	22.87	**55.45**	**63.17**	**66.38**	16.70	26.20	8.18	14.52	19.31	0.09	0.63
PFE$_{\text{fuse+match}}$	4.4M	50.16	58.33	62.28	**31.88**	**35.33**	53.60	61.75	64.97	**35.99**	**39.82**	9.20	**20.82**	**27.34**	**0.84**	**2.83**

Table 5: Performance comparison on three protocols of IJB-S. The performance is reported in terms of rank retrieval (closed-set) and TPIR@FPIR (open-set) instead of the media-normalized version [15]. The numbers "1%" and "10%" in the second row refer to the FPIR.



(a) Gaussian Blur (b) Occlusion (c) Random Noise

Figure 6: Repeated experiments on feature ambiguity dilemma with the proposed PFE. The same model in Figure 3 is used as the base model and is converted to a PFE by training an uncertainty module. No additional training data nor data augmentation is used for training.



(a) Low-score Genuine Pairs (b) High-score Impostor Pairs

Figure 7: Example genuine pairs from IJB-A dataset estimated with the lowest mutual likelihood scores and impostor pairs with the highest scores by the PFE version of the same 64-layer CNN model in Section 3. In comparison to Figure 4, most images here are high-quality ones with clear features, which can mislead the model to be confident in a wrong way. Note that these pairs are not templates in the verification protocol.

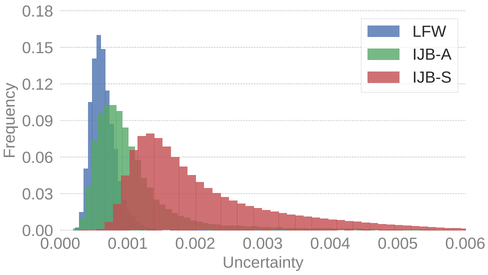


Figure 8: Distribution of estimated uncertainty on different datasets. Here, "Uncertainty" refers to the harmonic mean of σ across all feature dimensions. Note that the estimated uncertainty is proportional to the complexity of the datasets. **Best viewed in color**.

by the feature ambiguity dilemma (Section 3) will show up in MegaFace and it naturally favors deterministic embeddings. However, the PFE still maintains the performance in this case. We also note that such a bias, namely the target gallery images being of higher quality than the rest of gallery, would not exist in real world applications.

In Table 3 and Table 4 we show the results on three more challenging datasets: CFP, IJB-A and IJB-C. The images in these datasets present larger variations in pose, occlustion, etc, and facial features could be more ambiguous. As such, we can see that PFE achieves a more significant improvement on these three benchmarks. In particular on IJB-C at FAR$=0.001\%$, PFE reduces the error rate by 64%. In addition, simply fusing the original features with the learned uncertainty (PFE$_{\text{fuse}}$) also helps the performance.

In Table 5 we report the results on three protocols of the latest benchmark, IJB-S. Again, PFE is able to improve the performance in most cases. Notice that the gallery templates in the "Surveillance-to-still" and "Surveillance-to-booking" all include high-quality frontal mugshots, which present little feature ambiguity. Therefore, we see only see a slight performance gap in these two protocols. But in the most challenging "surveillance-to-surveillance" protocol, larger improvement can be achieved by using uncertainty for matching. Besides, PFE$_{\text{fuse+match}}$ improves the performance significantly on all the open-set protocols, which indicates that MLS has more impact on the absolute pairwise score than the relative ranking.

5.3. Qualitative Analysis

Why and when does PFE improve performance? We first repeat the same experiments in Section 3 using the PFE representation and MLS. The same network is used as the base model here. As one can see, although the scores of low-quality impostor pairs are still increasing, they converge to a point that is lower than the majority of genuine scores. Similarly, the scores of cross-quality genuine pairs converge to a point that is higher than the majority of impostor scores. This means the two types of errors discussed in Section 3 could be solved by PFE. This is further confirmed by the IJB-A results in Figure 7. Figure 3 shows the dis-

clude those between probes and distractors.

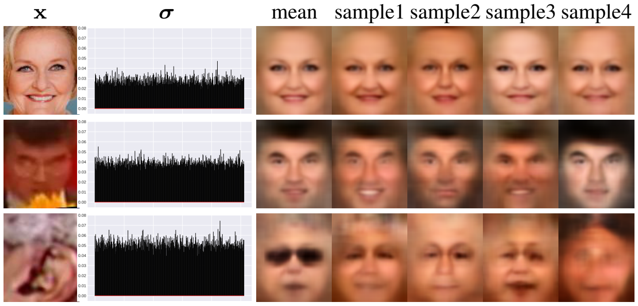


Figure 9: Visualization results on a high-quality, a low-quality and a mis-detected image from IJB-A. For each input, 5 images are reconstructed by a pre-trained decoder using the mean and 4 randomly sampled $\mathbf{z}$ vectors from the estimated distribution $p(\mathbf{z}|\mathbf{x})$.

tribution of estimated uncertainty on LFW, IJB-A and IJB-S. As one can see, the "variance" of uncertainty increases in the following order: LFW < IJB-A < IJB-S. Comparing with the performance in Section 5.2, we can see that PFE tends to achieve larger performance improvement on datasets with more diverse image quality.

What does DNN see and not see? To answer this question, we train a decoder network on the original embedding, then apply it to PFE by sampling $\mathbf{z}$ from the estimated distribution $p(\mathbf{z}|\mathbf{x})$ of given $\mathbf{x}$. For a high-quality image (Figure 9 Row 1), the reconstructed images tend to be very consistent without much variation, implying the model is very certain about the facial features in this images. In contrast, for a lower-quality input (Figure 9 Row 2), larger variation can be observed from the reconstructed images. In particular, attributes that can be clearly discerned from the image (*e.g*. thick eye-brow) are still consistent while attributes cannot (*e.g*. eye shape) be discerned have larger variation. As for a mis-detected image (Figure 9 Row 3), significant variation can be observed in the reconstructed images: the model does not see any salient feature in the given image.

6. Risk-controlled Face Recognition

In many scenarios, we may expect a higher performance than our system is able to achieve or we may want to make sure the system's performance can be controlled when facing complex application scenarios. Therefore, we would expect the model to reject input images if it is not confident. A common solution for this is to filter the images with a quality assessment tool. We show that PFE provides a natural solution for this task. We take all the images from LFW and IJB-A datasets for image-level face verification (We do not follow the original protocols here). The system is allowed to "filter out" a proportion of all images to maintain a better performance. We then report the TAR@FAR$= 0.001\%$ against the "Filter Out Rate". We consider two criteria for filtering: (1) the detection score of MTCNN [41] and (2) a confidence value predicted by our uncertainty module. Here the confidence for i^{th} sample is defined as the inverse of harmonic mean of $\boldsymbol{\sigma}_i$ across all dimensions. For fairness, both methods use the original deterministic embedding representations and cosine similarity for matching. To avoid saturated results, we use the model trained on CASIA-WebFace with AM-Softmax. The results are shown in Figure 11. As one can see, the predicted confidence value is a better indicator of the potential recognition accuracy of the input image. This is an expected result since PFE is trained under supervision for the particular model while an external quality estimator is unaware of the kind of features used for matching by the model. Example images with high/low confidence/quality scores are shown in Figure 10.

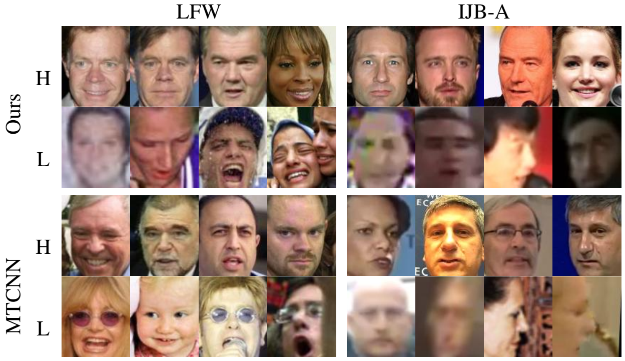
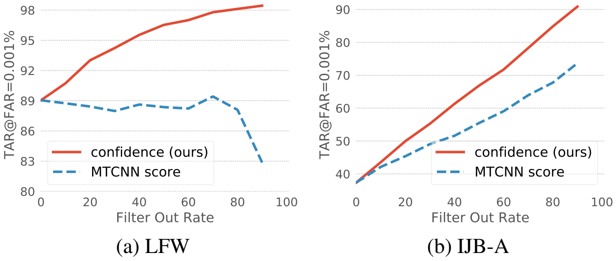


Figure 10: Example images from LFW and IJB-A that are estimated with the highest (H) confidence/quality scores and the lowest (L) scores by our method and MTCNN face detector.



(a) LFW

(b) IJB-A

Figure 11: Comparison of verification performance on LFW and IJB-A (not the original protocol) by filtering a proportion of images using different quality criteria.

7. Conclusion

We have proposed probabilistic face embeddings (PFEs), which represent face images as distributions in the latent space. Probabilistic solutions were derived to compare and aggregate the PFE of face images. Unlike deterministic embeddings, PFEs do not suffer from the feature ambiguity dilemma for unconstrained face recognition. Quantitative and qualitative analysis on different settings showed that PFEs can effectively improve the face recognition performance by converting deterministic embeddings to PFEs. We have also shown that the uncertainty in PFEs is a good indicator for the "discriminative"quality of face images. In the future work we will explore how to learn PFEs in an end-to-end manner and how to address the data dependency within face templates.

References

[1] A. A. Alemi, I. Fischer, J. V. Dillon, and K. Murphy. Deep variational information bottleneck. In *ICLR*, 2017. 2

[2] O. Arandjelovic, G. Shakhnarovich, J. Fisher, R. Cipolla, and T. Darrell. Face recognition with image sets using manifold density divergence. In *CVPR*, 2005. 2

[3] Q. Cao, L. Shen, W. Xie, O. M. Parkhi, and A. Zisserman. Vggface2: A dataset for recognising faces across pose and age. In *FG*, 2018. 6

[4] H. Cevikalp and B. Triggs. Face recognition based on image sets. In *CVPR*, 2010. 2

[5] J. Deng, J. Guo, N. Xue, and S. Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. *arXiv:1801.07698*, 2018. 1, 6

[6] Y. Gal and Z. Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *ICML*, 2016. 2

[7] S. Gong, V. N. Boddeti, and A. K. Jain. On the capacity of face representation. *arXiv:1709.10433*, 2017. 2

[8] S. Gong, Y. Shi, and A. K. Jain. Video face recognition: Component-wise feature aggregation network (c-fan). In *ICB*, 2019. 2, 7

[9] Y. Guo, L. Zhang, Y. Hu, X. He, and J. Gao. Ms-celeb-1m: A dataset and benchmark for large-scale face recognition. In *ECCV*, 2016. 3, 6, 11, 12

[10] P. Hiremath, A. Danti, and C. Prabhakar. Modelling uncertainty in representation of facial features for face recognition. In *Face recognition*. 2007. 2

[11] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical report, University of Massachusetts, Amherst, 2007. 1, 3, 5, 6

[12] Z. Huang, R. Wang, S. Shan, X. Li, and X. Chen. Log-euclidean metric learning on symmetric positive definite manifold with application to image set classification. In *ICML*, 2015. 2

[13] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015. 11

[14] T. Jebara, R. Kondor, and A. Howard. Probability product kernels. *Journal of Machine Learning Research*, 2004. 4

[15] N. D. Kalka, B. Maze, J. A. Duncan, K. J. OConnor, S. Elliott, K. Hebert, J. Bryan, and A. K. Jain. IJB-S : IARPA Janus Surveillance Video Benchmark . In *BTAS*, 2018. 1, 2, 5, 7

[16] I. Kemelmacher-Shlizerman, S. M. Seitz, D. Miller, and E. Brossard. The megaface benchmark: 1 million faces for recognition at scale. In *CVPR*, 2016. 5

[17] A. Kendall, V. Badrinarayanan, and R. Cipolla. Bayesian segnet: Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding. *arXiv:1511.02680*, 2015. 2

[18] A. Kendall and Y. Gal. What uncertainties do we need in bayesian deep learning for computer vision? In *NIPS*, 2017. 2, 11

[19] S. Khan, M. Hayat, W. Zamir, J. Shen, and L. Shao. Striking the right balance with uncertainty. *arXiv:1901.07590*, 2019. 2

[20] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv:1312.6114*, 2013. 2

[21] B. F. Klare, B. Klein, E. Taborsky, A. Blanton, J. Cheney, K. Allen, P. Grother, A. Mah, and A. K. Jain. Pushing the frontiers of unconstrained face detection and recognition: IARPA Janus Benchmark A. In *CVPR*, 2015. 1, 2, 5, 6

[22] H. Li, G. Hua, Z. Lin, J. Brandt, and J. Yang. Probabilistic elastic matching for pose variant face verification. In *CVPR*, 2013. 2

[23] W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, and L. Song. Sphereface: Deep hypersphere embedding for face recognition. In *CVPR*, 2017. 1, 6, 12

[24] Y. Liu, J. Yan, and W. Ouyang. Quality aware network for set to set recognition. In *CVPR*, 2017. 2, 5, 6

[25] D. J. MacKay. A practical bayesian framework for backpropagation networks. *Neural Computation*, 1992. 2

[26] B. Maze, J. Adams, J. A. Duncan, N. Kalka, T. Miller, O. Charles, A. K. Jain, N. Tyler, J. Anderson, J. Cheney, and P. Grother. Iarpa janus benchmark-c: Face dataset and protocol. In *ICB*, 2018. 1, 5

[27] R. M. Neal. *Bayesian learning for neural networks*. PhD thesis, University of Toronto, 1995. 2

[28] S. J. Oh, K. Murphy, J. Pan, J. Roth, F. Schroff, and A. Gallagher. Modeling uncertainty with hedged instance embedding. In *ICLR*, 2019. 2

[29] R. Ranjan, C. D. Castillo, and R. Chellappa. L2-constrained softmax loss for discriminative face verification. *arXiv:1703.09507*, 2017. 6

[30] S. Sankaranarayanan, A. Alavi, C. D. Castillo, and R. Chellappa. Triplet probabilistic embedding for face verification and clustering. In *BTAS*, 2016. 6

[31] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In *CVPR*, 2015. 1, 6, 12

[32] S. Sengupta, J.-C. Chen, C. Castillo, V. M. Patel, R. Chellappa, and D. W. Jacobs. Frontal to profile face verification in the wild. In *WACV*, 2016. 5, 6

[33] G. Shakhnarovich, J. W. Fisher, and T. Darrell. Face recognition from long-term observations. In *ECCV*, 2002. 2

[34] Y. Sun, X. Wang, and X. Tang. Deeply learned face representations are sparse, selective, and robust. In *CVPR*, 2015. 6

[35] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf. Deepface: Closing the gap to human-level performance in face verification. In *CVPR*, 2014. 6

[36] L. Tran, X. Yin, and X. Liu. Disentangled representation learning gan for pose-invariant face recognition. In *CVPR*, 2017. 6

[37] L. Vilnis and A. McCallum. Word representations via gaussian embedding. In *ICLR*, 2015. 2

[38] F. Wang, J. Cheng, W. Liu, and H. Liu. Additive margin softmax for face verification. *IEEE Signal Processing Letters*, 2018. 3, 6, 12

[39] 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. In *CVPR*, 2018. 1, 6

[40] Y. Wen, K. Zhang, Z. Li, and Y. Qiao. A discriminative feature learning approach for deep face recognition. In *ECCV*, 2016. 1, 6, 12

[41] Y. Wen, K. Zhang, Z. Li, and Y. Qiao. A discriminative feature learning approach for deep face recognition. In *ECCV*, 2016. 8, 11

[42] L. Wolf, T. Hassner, and I. Maoz. Face recognition in unconstrained videos with matched background similarity. In *CVPR*, 2011. 1, 5, 6

[43] X. Wu, R. He, Z. Sun, and T. Tan. A light cnn for deep face representation with noisy labels. *arXiv:1511.02683*, 2015. 12

[44] W. Xie, L. Shen, and A. Zisserman. Comparator networks. In *ECCV*, 2018. 6

[45] W. Xie and A. Zisserman. Multicolumn networks for face recognition. *arXiv:1807.09192*, 2018. 2, 5, 6

[46] Y. Xu, X. Fang, X. Li, J. Yang, J. You, H. Liu, and S. Teng. Data uncertainty in face recognition. *IEEE Trans. on cybernetics*, 2014. 2

[47] J. Yang, P. Ren, D. Zhang, D. Chen, F. Wen, H. Li, and G. Hua. Neural aggregation network for video face recognition. In *CVPR*, 2017. 2, 5, 6

[48] D. Yi, Z. Lei, S. Liao, and S. Z. Li. Learning face representation from scratch. *arXiv:1411.7923*, 2014. 5, 11, 12

[49] B. Yin, L. Tran, H. Li, X. Shen, and X. Liu. Towards interpretable face recognition. *arXiv:1805.00611*, 2018. 6

[50] X. Yin and X. Liu. Multi-task convolutional neural network for pose-invariant face recognition. *IEEE Trans. on Image Processing*, 2018. 6

[51] U. Zafar, M. Ghafoor, T. Zia, G. Ahmed, A. Latif, K. R. Malik, and A. M. Sharif. Face recognition with bayesian convolutional networks for robust surveillance systems. *EURASIP Journal on Image and Video Processing*, 2019. 2

A. Proofs

A.1. Mutual Likelihood Score

Here we prove Equation (3) in the main paper. For simplicity, we do not need to directly solve the integral. Instead, let us consider an alternative vector $\Delta\mathbf{z} = \mathbf{z}_i - \mathbf{z}_j$, where $\mathbf{z}_i \sim p(\mathbf{z}|\mathbf{x}_i)$, $\mathbf{z}_j \sim p(\mathbf{z}|\mathbf{x}_j)$ and $(\mathbf{x}_i, \mathbf{x}_j)$ are the pair of images we need to compare. Then, $p(\mathbf{z}_i = \mathbf{z}_j)$, *i.e.* Equation (2) in the main paper, is equivalent to the density value of $p(\Delta\mathbf{z} = \mathbf{0})$.

The l^{th} component (dimension) of $\Delta\mathbf{z}$, $\Delta z^{(l)}$, is the subtraction of two Gaussian variables, which means:

$$\Delta z^{(l)} \sim \mathcal{N}(\mu_i^{(l)} - \mu_j^{(l)}, \sigma_i^{2(l)} + \sigma_j^{2(l)}). \tag{10}$$

Therefore, the mutual likelihood score is given by:

$$\begin{aligned}
& s(\mathbf{x}_i, \mathbf{x}_j) \\
= \; & \log p(\mathbf{z}_i = \mathbf{z}_j) \\
= \; & \log p(\Delta\mathbf{z} = \mathbf{0}) \\
= \; & \sum_{l}^{D} \log p(\Delta z^{(l)} = 0) \\
= \; & -\frac{1}{2}\sum_{l=1}^{D}\Big(\frac{(\mu_i^{(l)} - \mu_j^{(l)})^2}{\sigma_i^{2(l)} + \sigma_j^{2(l)}} + \log(\sigma_i^{2(l)} + \sigma_j^{2(l)})\Big) \\
& -\frac{D}{2}\log 2\pi.
\end{aligned} \tag{11}$$

Note that directly solving the integral will lead to the same solution.

A.2. Property 1

Let us consider the case that $\sigma_i^{2(l)}$ equals to a constant $c > 0$ for any image $\mathbf{x}_i$ and dimension l. Thus the mutual likelihood score between a pair $(\mathbf{x}_i, \mathbf{x}_j)$ becomes:

$$\begin{aligned}
& s(\mathbf{x}_i, \mathbf{x}_j) \\
= \; & -\frac{1}{2}\sum_{l=1}^{D}\Big(\frac{(\mu_i^{(l)} - \mu_j^{(l)})^2}{2c} + \log(2c)\Big) - \frac{D}{2}\log 2\pi \\
= \; & -c_1 \left\|\boldsymbol{\mu}_i - \boldsymbol{\mu}_j\right\|^2 - c_2,
\end{aligned} \tag{12}$$

where $c_1 = \frac{1}{4c}$ and $c_2 = \frac{D}{2}\log(4\pi c)$ are both constants.

A.3. Representation Fusion

We first prove Equation (5) in the main paper. Assuming all the observations $\mathbf{x}_1, \mathbf{x}_2 \ldots \mathbf{x}_{n+1}$ are conditionally independent given the latent code z. The posterior distribution is:

$$\begin{aligned}
& p(\mathbf{z}|\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{n+1}) \\
= & \frac{p(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{n+1}|\mathbf{z})p(\mathbf{z})}{p(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{n+1})} \\
= & \frac{p(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{n}|\mathbf{z})p(\mathbf{x}_{n+1}|\mathbf{z})p(\mathbf{z})}{p(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{n+1})} \\
= & \frac{p(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{n})p(\mathbf{x}_{n+1})}{p(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{n+1})} \frac{p(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{n}|\mathbf{z})p(\mathbf{x}_{n+1}|\mathbf{z})p(\mathbf{z})}{p(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{n})p(\mathbf{x}_{n+1})} \\
= & \alpha\frac{p(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{n}, \mathbf{z})p(\mathbf{x}_{n+1}, \mathbf{z})}{p(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{n})p(\mathbf{x}_{n+1})p(\mathbf{z})} \\
= & \alpha\frac{p(\mathbf{z}|\mathbf{x}_{n+1})}{p(\mathbf{z})}p(\mathbf{z}|\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{n}),
\end{aligned} \tag{13}$$

where α is a normalization constant. In this case, $\alpha = \frac{p(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{n})p(\mathbf{x}_{n+1})}{p(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{n+1})}$.

Without loss of generality, let us consider a one-dimensional case for the followings. The solution can be easily extended to a multivariate case since all feature dimensions are assumed to be independent. It can be shown that the posterior distribution in Equation (13) is a Gaussian distribution through induction. Let us assume $p(z|\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n)$ is a Gaussian distribution with $\hat{\mu}_n$ and $\hat{\sigma}_n^2$ as mean and variance, respectively. Note that the initial case $p(z|\mathbf{x}_1)$ is guaranteed to be a Gaussian distribution. Let μ_0 and σ_0^2 denote the parameters of the noninformative prior of z. Then, if we take log on both side of Equation (13), we have:

$$\begin{aligned}
&\log p(z|\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_{n+1}) \\
=& \log p(z|\mathbf{x}_{n+1}) + \log p(z|\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n) - \log p(z) + \text{const} \\
=& -\frac{(z-\mu_{n+1})^2}{2\sigma_{n+1}^2} - \frac{(z-\hat{\mu}_n)^2}{2\hat{\sigma}_n^2} + \frac{(z-\mu_0)^2}{2\sigma_0^2} + \text{const} \\
=& -\frac{(z-\hat{\mu}_{n+1})^2}{2\hat{\sigma}_{n+1}^2} + \text{const}.
\end{aligned} \tag{14}$$

where "const" refers to the terms irrelevant to z and

$$\hat{\mu}_{n+1} = \hat{\sigma}_{n+1}^2 \left(\frac{\mu_{n+1}}{\sigma_{n+1}^2} + \frac{\hat{\mu}_n}{\hat{\sigma}_n^2} - \frac{\mu_0}{\sigma_0^2}\right), \tag{15}$$

$$\frac{1}{\hat{\sigma}_{n+1}^2} = \frac{1}{\sigma_{n+1}^2} + \frac{1}{\hat{\sigma}_n^2} - \frac{1}{\sigma_0^2}. \tag{16}$$

Considering $\sigma_0 \to \infty$, we have

$$\hat{\mu}_{n+1} = \frac{\hat{\sigma}_n^2 \mu_{n+1} + \sigma_{n+1}^2 \hat{\mu}_n}{\sigma_{n+1}^2 + \hat{\sigma}_n^2}, \tag{17}$$

$$\hat{\sigma}_{n+1}^2 = \frac{\sigma_{n+1}^2 \hat{\sigma}_n^2}{\sigma_{n+1}^2 + \hat{\sigma}_n^2}. \tag{18}$$

The result means the posterior distribution is a new Gaussian distribution with a smaller variance. Further, we can directly give the solution of fusing n samples:

$$\begin{aligned}
&\log p(z|\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n) \\
=& \log[\alpha p(z|\mathbf{x}_1) \prod_{i=2}^{n} \frac{p(z|\mathbf{x}_i)}{p(z)}] \\
=& (n-1)\log p(z) - \sum_{i=1}^{n} \log p(z|\mathbf{x}_i) + \text{const} \\
=& (n-1)\frac{(z-\mu_0)^2}{2\sigma_0^2} - \sum_{i=1}^{n} \frac{(z-\mu_i)^2}{2\sigma_i^2} + \text{const} \\
=& -\frac{(z-\hat{\mu}_n)^2}{2\hat{\sigma}_n^2} + \text{const}.
\end{aligned} \tag{19}$$

where $\alpha = \frac{\prod_{i=1}^{n} p(\mathbf{x}_i)}{p(\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n)}$ and

$$\hat{\mu}_n = \sum_{i=1}^{n} \frac{\hat{\sigma}_n^2}{\sigma_i^2} \mu_i - (n-1)\frac{\hat{\sigma}_n^2}{\sigma_0^2} \mu_0, \tag{20}$$

$$\frac{1}{\hat{\sigma}_n^2} = \sum_{i=1}^{n} \frac{1}{\sigma_i^2} - (n-1)\frac{1}{\sigma_0^2}. \tag{21}$$

Considering $\sigma_0 \to \infty$, we have

$$\hat{\mu}_n = \sum_{i=1}^{n} \frac{\hat{\sigma}_n^2}{\sigma_i^2} \mu_i, \tag{22}$$

$$\frac{1}{\hat{\sigma}_n^2} = \sum_{i=1}^{n} \frac{1}{\sigma_i^2}. \tag{23}$$

B. Implementation Details

All the models in the paper are implemented using Tensorflow r1.9. Two and Four GeForce GTX 1080 Ti GPUs are used for training base models on CASIA-Webface [48] and MS-Celeb-1M [9], respectively. The uncertainty modules are trained using one GPU.

B.1. Data Preprocessing

All the face images are first passed through MTCNN face detector [41] to detect 5 facial landmarks (two eyes, nose and two mouth corners). Then, similarity transformation is used to normalize the face images based on the five landmarks. After transformation, the images are resized to 112×96. Before passing into networks, each pixel in the RGB image is normalized by subtracting 127.5 and dividing by 128.

B.2. Base Models

The base models for CASIA-Webface [48] are trained for $28,000$ steps using a SGD optimizer with a momentum of 0.9. The learning rate starts at 0.1, and is decreased to 0.01 and 0.001 after $16,000$ and $24,000$ steps, respectively. For the base model trained on Ms-Celeb-1M [9], we train the network for $140,000$ steps using the same optimizer settings. The learning rate starts at 0.1, and is decreased to 0.01 and 0.001 after $80,000$ and $120,000$ steps, respectively. The batch size, feature dimension and weight decay are set to 256, 512 and 0.0005, respectively, for both cases.

B.3. Uncertainty Module

Architecture The uncertainty module for all models are two-layer perceptrons with the same architecture: `FC-BN-ReLU-FC-BN-exp`, where `FC` refers to fully connected layers, `BN` refers to batch normalization layers [13] and `exp` function ensures the outputs σ^2 are all positive values [18]. The first `FC` shares the same input with the bottleneck layer, *i.e.* the output feature map of the last convolutional layer. The output of both `FC` layers are D-dimensional vectors, where D is the dimensionality of the latent space. In addition, we constrain the last `BN` layer to share the same γ and β across all dimensions, which we found to help stabilizing the training.

Training For the models trained on CASIA-WebFace [48], we train the uncertainty module for

Base Model	Representation	LFW	YTF	CFP-FP	IJB-A
Softmax + Center Loss [40]	Original	97.70	92.56	91.13	63.93
	PFE	**97.89**	**93.10**	**91.36**	**64.33**
Triplet [31]	Original	96.98	90.72	**85.69**	**54.47**
	PFE	**97.10**	**91.22**	85.10	51.35
A-Softmax [23]	Original	97.12	**92.38**	89.31	54.48
	PFE	**97.92**	91.78	**89.96**	**58.09**
AM-Softmax [38]	Original	98.32	93.50	90.24	71.28
	PFE	**98.63**	**94.00**	**90.50**	**75.92**

Table 6: Results of CASIA-Net models trained on CASIA-WebFace. "Orignal" refers to the deterministic embeddings. The better performance among each base model are shown in bold numbers. "PFE" uses mutual likelihood score for matching. IJB-A results are verification rates at FAR=0.1%.

Base Model	Representation	LFW	YTF	CFP-FP	IJB-A
Softmax + Center Loss [40]	Original	97.77	92.34	90.96	60.42
	PFE	**98.28**	**93.24**	**92.29**	**62.41**
Triplet [31]	Original	97.48	92.46	90.01	52.34
	PFE	**98.15**	**93.62**	**90.54**	**56.81**
A-Softmax [23]	Original	98.07	92.72	89.34	63.21
	PFE	**98.47**	**93.44**	**90.54**	**71.96**
AM-Softmax [38]	Original	98.68	93.78	90.59	76.50
	PFE	**98.95**	**94.34**	**91.26**	**80.00**

Table 7: Results of Light-CNN models trained on CASIA-WebFace. "Orignal" refers to the deterministic embeddings. The better performance among each base model are shown in bold numbers. "PFE" uses mutual likelihood score for matching. IJB-A results are verification rates at FAR=0.1%.

3,000 steps using a SGD optimizer with a momentum of 0.9. The learning rate starts at 0.001, and is decreased to 0.0001 after 2,000 steps. For the model trained on MS-Celeb-1M[9], we train the uncertainty module for 12,000 steps. The learning rate starts at 0.001, and is decreased to 0.0001 after 8,000 steps. The batch size for both cases are 256. For each mini-batch, we randomly select 4 images from 64 different subjects to compose the positive pairs (384 pairs in all). The weight decay is set to 0.0005 in all cases. A Subset of the training data was separated as the validation set for choosing these hyper-parameters during development phase.

Inference Speed Feature extraction (passing through the whole network) using one GPU takes 1.5ms per image. Note that given the small size of the uncertainty module, it has little impact on the feature extraction time. Matching images using cosine similarity and mutual likelihood score takes 4ns and 15ns , respectively. Both are neglectable in comparison with feature extraction time.

C. Results on Different Architectures

Throughout the main paper, we conducted the experiments using a 64-layer CNN network [23]. Here, we evaluate the proposed method on two different network architectures for face recognition: CASIA-Net [48] and 29-layer Light-CNN [43]. Notice that both networks require different image shapes from our preprocessed ones. Thus we pad our images with zero values and resize them into the target size. Since the main purpose of the experiment is to evaluate the efficacy of the uncertainty module rather than comparing with the original results of these networks, the difference in the preprocessing should not affect a fair comparison. Besides, the original CASIA-Net does not converge for A-Softmax and AM-Softmax, so we add an bottleneck layer to output the embedding representation after the average pooling layer. Then we conduct the experiments by comparing probabilistic embeddings with base deterministic embeddings, similar to Section 5.1 in the main paper. The results are shown in Table 6 and Table 7. Without tuning the architecture of the uncertainty module nor the hyper-parameters, PFE still improve the performance in most cases.