Learning Low-shot facial representations via 2D warping

Shen Yan
RWTH Aachen University
shen.yan@rwth-aachen.de

Abstract

Face recognition has seen a significant improvement by using the deep convolutional neural networks. In this work, we mainly study the influence of the 2D warping module for one-shot face recognition. To achieve this, we first propose a 2D-Warping Layer to generate new features for the novel classes during the training, then fine-tuning the network by adding the recent proposed fisher loss to learn more discriminative features. We evaluate the proposed method on two popular databases for unconstrained face recognition, the Labeled Faces in the Wild (LFW) and the Youtube Faces (YTF) database. In both cases, the proposed method achieves competitive results with the accuracy of 99.25% for LFW and 94.3% for YTF, separately. Moreover, the experimental results on MS-Celeb-1M one-shot faces dataset show that with the proposed method, the model achieves comparable results of 77.92% coverage rate at precision = 99% for the novel classes while still keeps top-1 accuracy of 99.80% for the normal classes.

1. Introduction

Recently, there has seen a significant improvement in the field of face recognition since the emergence of deep Convolution Neural Networks (CNNs) [13]. Especially in the area of face recognition, there is a big boost in performance with the large-scale dataset as described in [23, 17, 20]. In terms of face verification task, where two images are used to compare whether they belong to the same class or not, is often determined by transforming to the embedding space via a deep network. Different network architectures or loss functions are proposed to learn discriminative features in the embedding space as described in [5, 20, 25]. The embedding space is then used to compute the similarity and distance given a pair of face images. Specifically, in [25] the authors propose the center loss (intra class loss) that aims to minimize the feature distances within the same class in the embedding space. They train the model by using a standard backpropagation and global centers are updated after each iteration using only the features within the mini-batch during the training. One possible drawback of this method is that it only considers the intra-class compactness but completely ignores the inter-class dispersion during the optimization. Hence, in [7] the authors consider adding inter-class distances to help discriminative face representations learning. However, their method is bounded by a hard-crafted margin, which could restrict the optimization during training. Therefore, our work first did a minor change to the fisher loss by directly optimizing the inter-class part based on inter-class distance. To guarantee the overall optimization, an additional hyper-parameter is introduced to balance the intra-class and inter-class term.

Moreover, as described in [3], the well-established face recognition systems are still hard to learn strong visual representation from the low-shot face images. Specifically, given a dataset some of whose classes only have few images for training, the network can not generalize well across those novel classes. To make our network able to generalize well for the low-shot learning task, followed by [6] we reformulate the idea into an end-to-end 2D-Warping Module to handle the unbalanced data problem by generating new warped features for the low-shot ones during the training. For that we define a two-step training procedure. At first, a zero-order 2D-Warping (Section 4) is used to find a warping between two given images. During the warping, one of the images can be regarded as a source image and the other as a target image. The mapping aims to let the source image pixels match the target image pixels. The warping is optimized by the defined criterion including local feature similarity and pixel level displacement. We use the defined warping criterion to generate the new warped features for one-shot features within a mini-batch during the training. Second, we fine-tuning (training with lower learning rate) the network with the joint supervision of fisher loss and softmax cross-entropy loss so that the network is able to learn discriminative features for both normal classes and novel classes.

We evaluate the proposed method on three benchmark databases for face recognition, the Labeled Faces in the Wild (LFW) [11], the Youtube Faces (YTF) [26] and the
MS-Celeb-1M one-shot [3] databases (Section 6).

This paper is structured as follows. After an overview of related work in Section 2, we recap the fisher loss in Section 3 and give a minor changed version. Especially for the low-shot learning task, we reformulate the zero-order warping criterion into a 2D Warping Module to generate new features for the low shot features in Section 4. In Section 5, we describe the details of training the network for different tasks and the experimental results on three public available datasets are presented in Section 6. Finally, the paper is concluded with discussion and outlook in Section 7.

2. Related Work

Recently, there are a lot of works regarding loss functions for face recognition. Loss functions including the contrastive loss [5] and the triplet loss [20] are both aiming to minimize feature distances within the same class while maximizing feature distances of different classes. This type of loss can be used either stand-alone [20], jointly with a softmax cross-entropy loss [22], or to fine-tuning a baseline model [17]. However, both contrastive and triplet loss suffer a lot from the pairs/triplets selection. Even though there are several algorithms proposed for mining informative pairs/triplets, it is still very hard to optimize due to the largely different combination possibilities. Compared to them, center loss [25] does not need a complex combination of the training samples and enjoys the same requirements as the softmax-cross entropy loss. The goal of center loss is to minimize the intra-class variance within the mini-batch. However, it doesn’t consider the inter-class part, where in [7] the authors think is critical to discriminative feature learning. There are also related work for local feature learning but the methods are not especially for face feature representation, instead, they focus on patch-similarity learning. In [21, 27] where the siamese network is used to learning discriminative features given image patches.

For the low-shot learning task, in [24] where an approach proposed to learn similarity embedding based on the full context of the input instead of using pairwise similarity matching and in [8] where squared gradient magnitude (SGM) is used to regularize the low-shot features. In this work, we propose a 2D-Warping Layer based on the zero-order warping criterion to generate new features for the novel classes during training. Traditionally there are several methods used the 2D-Warping technique for face recognition [6, 2, 28]. However, instead of using 2D-Warping Module as an end-to-end fashion to hallucinate imposter training features, these methods mainly focus on warping complexity optimization, keypoint matching, or stereo vision.

3. Fisher Loss Recap

Similar to the idea proposed in [1], in [7] the author simultaneously optimized the intra-class variance and the inter-class variance by maximizing the between-class variance.

3.1. Center Loss

The goal of center loss is to minimize the within-class distance in a mini-batch. Here we define $m$ to be the batch size, $t$ to be the current iteration step, $x_i^t$ to be current network outputs before the classifier and $y_i^t$ to be the labels at current iteration. Note that the notations are slightly different from [25] after the iteration step $t$ be introduced. Given the following definition, the center loss can be reformulated as:

$$L_c = \frac{1}{2} \sum_{i=1}^{m} \|x_i^t - c_{y_i^t}^t\|^2$$

3.2. Modified Center Loss

In order to make it possible to update the centers during learning instead of using a specified update rule, the term (Formula 1) can be modified by simply changing $c_{y_i^t}^t$ to $c'_{y_i^t}$ in Formula (1), which leads to $L_{mc}$

$$L_{mc} = \frac{1}{2} \sum_{i=1}^{m} \|x_i^t - c'_{y_i^t}\|^2$$

3.3. Modified Inter-Class Loss

In [7], the author bounded the the inter-class distance with a margin $m$ turning the maximization into a minimization, to prevent a learning behavior where the inter-class distances increase too much compared to the intra-class distances during the optimization. This leads to the following fisher loss objective

$$L_f = \frac{1}{2} \sum_{i=1}^{m} \|x_i^t - c'_{y_i^t}\|^2 + \left( \frac{1}{2} \sum_{(y,y') \in Y^t} \max \left( m - \|c_y^t - c'_{y'}\|^2, 0 \right) \right)$$

where the set $Y^t$ contains all unique pairs $(y, y')$ of different classes within a batch at current iteration $t$. However, it is hard to select a well-defined margin in practical and the margin could also potentially restrict the optimization. For this reason, we change the inter-part by directly optimizing the inter-class distance, which leads to the following:
the intra-class variance is reduced, the data points within the same class become closer in the embedding space. Compared to them, Figure (c) is achieved by fine-tuning the model by adding the modified fisher loss, where the centers with the same class keep a relatively small variance and centers with different classes get a higher variance. We observed similar projection comparing modified fisher loss and original fisher loss. Since the focus in this work is not on the loss functions, we only use modified fisher loss for the other experiments.

4. Zero-Order 2D-Warping

Specifically for the low-shot learning, we aim to generate more training samples for novel classes during the training in order to compensate the imbalanced data problem. For this reason, followed by [6] we encapsulate the zero-order warping criterion into an end-to-end 2D warping module. The goal of 2D-Warping is to find an optimal mapping between the source and target image. In [18] and [6] the problem is defined as follows: given a source image $S$ with dimension $I \times J$ and a target image $T$ with dimension $U \times V$, there exists a warping function $w$ that can assign a pixel $w_{ij} = (u, v) \in T$ to each pixel $(i, j) \in S$. In this paper, the 2D-Warping criterion is similar with [12, 18, 6]. Instead of considering the 2D-dependencies, each pixel is optimized independently. Therefore, it is called zero-order 2D-Warping. The warping is optimized by finding each wrapped pixel location $w_{i,j}$ which minimizes the following energy function

$$E(S, T, \{(i, j), w_{ij}\}) = \sum_{i,j} \left[d(s_{ij}, t_{w_{ij}}) + T_\Delta((i, j), w_{ij})\right]$$

where

$$T_\Delta((i, j), w_{ij} = (u, v)) = \begin{cases} \alpha \cdot d_{\text{pen}}((i, j), w_{ij}) & \text{if } |v - j| \leq \Delta \land |u - i| \leq \Delta \\ \infty & \text{else} \end{cases}$$

The defined energy function includes two part, a feature-level displacement and a pixel-level displacement. $d(\cdot)$ denotes a distance function between two feature vectors $s_{i,j}$ in the source feature map $S$ and $t_{w_{ij}}$ in the target feature map $T$. In our work, we use Euclidean distance in feature space to measure the feature displacement. $T_\Delta(\cdot)$ is a absolute penalty function, it penalizes the pixel deviations between the source pixel location $(i, j)$ and the warped pixel location $w_{ij} = (u, v)$ by using the distance function $d_{\text{pen}}(\cdot)$ weighted by a penalty weight $\alpha$. Additionally, an absolute warping range $\Delta$ is introduced to restrict the upper bound for the warping displacement. The overall warping speed is

Figure 1: Learned embedding space on MNIST with different losses. From left to right: (a) Train a baseline with only softmax cross-entropy loss. (b) Fine-tuning the baseline with softmax cross-entropy loss + center loss. (c) Fine-tuning the baseline with softmax cross-entropy loss + modified fisher loss with $\beta=0.01$. The details of the other hyper-parameters are shown in Section 3.4.

$$L_f = \frac{1}{2} \sum_{i=1}^{m} \|x_i - c_{y_i}^t\|^2 + \frac{\beta}{2} \sum_{(y, y') \in V_1} \|c_y^t - c_{y'}^t\|^2$$

$$L = L_{CE} + \lambda L_f$$

3.4. A toy example

We show the effect of modified inter-class loss using the MNIST dataset [15]. Similar to [25] we use a simple CNN with an immediate layer of size 2 as embedding layer to be able to visualize the 2D embedding space without any additional methods. We train a baseline model with softmax cross-entropy loss and then fine-tuning this model with the center loss as well as the fisher loss with a $\beta=0.01$. We use the same hyperparameters ($\lambda = 0.003$, $\beta = 0.01$, $\alpha = 0.5$, batch size = 128) for all models apart from the margin. Since the plots are only used for illustration of different losses, we don’t concentrate on the recognition performance.

Figure 1 shows the embedding for the different losses. Figure (a) is the baseline with only softmax cross-entropy loss. In this case, the network learns to keep different classes separable. Figure (b) is achieved by fine-tuning the model using the center loss with a lower learning rate. Since

$\beta$ is a factor to balance the intra-class part and inter-class part. In all our experiments we set $\beta=0.01$. The possible pairs combinations for the set $Y^t$ can be very large depending on the different batch size. E.g., given a batch size of 128 the most case would be $(128 \times 127) / 2 = 8128$ pairs, which is a relatively large number. In order to make the computation more efficient, a random sampling of a small subset can be used instead.

Similar to the center loss in [25] we weight the fisher loss with softmax entropy loss by a hyperparameter $\lambda$ leading to the following over all loss function:

$$L = L_{CE} + \lambda L_f$$

The goal of 2D-Warping is to find an optimal mapping between the source and target image. In [18] and [6] the problem is defined as follows: given a source image $S$ with dimension $I \times J$ and a target image $T$ with dimension $U \times V$, there exists a warping function $w$ that can assign a pixel $w_{ij} = (u, v) \in T$ to each pixel $(i, j) \in S$. In this paper, the 2D-Warping criterion is similar with [12, 18, 6]. Instead of considering the 2D-dependencies, each pixel is optimized independently. Therefore, it is called zero-order 2D-Warping. The warping is optimized by finding each wrapped pixel location $w_{i,j}$ which minimizes the following energy function

$$E(S, T, \{(i, j), w_{ij}\}) = \sum_{i,j} \left[d(s_{ij}, t_{w_{ij}}) + T_\Delta((i, j), w_{ij})\right]$$

where

$$T_\Delta((i, j), w_{ij} = (u, v)) = \begin{cases} \alpha \cdot d_{\text{pen}}((i, j), w_{ij}) & \text{if } |v - j| \leq \Delta \land |u - i| \leq \Delta \\ \infty & \text{else} \end{cases}$$

The defined energy function includes two part, a feature-level displacement and a pixel-level displacement. $d(\cdot)$ denotes a distance function between two feature vectors $s_{i,j}$ in the source feature map $S$ and $t_{w_{ij}}$ in the target feature map $T$. In our work, we use Euclidean distance in feature space to measure the feature displacement. $T_\Delta(\cdot)$ is a absolute penalty function, it penalizes the pixel deviations between the source pixel location $(i, j)$ and the warped pixel location $w_{ij} = (u, v)$ by using the distance function $d_{\text{pen}}(\cdot)$ weighted by a penalty weight $\alpha$. Additionally, an absolute warping range $\Delta$ is introduced to restrict the upper bound for the warping displacement. The overall warping speed is
Although very fast of $O(UV(2\Delta+1))$. One possible drawback of this method is that it discards the neighboring information so that the local image structure could not be captured during the warping. To make up for the problem, instead of warping the images directly, we warp the features with a relative small spatial size taking from one bottleneck of the network to preserve the representation inconsistency. Specifically, only one-shot features of spatial size $28 \times 28$ with depth 128 (extracted from one bottleneck of the standard 34-layers deep residual network) are used for warping. The experimental results in Section 6.3 show that with a well-defined warp range $\Delta$ and penalty weight $\alpha$, the quality of the warped features can be compensated within a limited spatial size.

This parameter-free criterion can be designed as a differentiable module added into the network, which we call it 2D-Warping Layer. As shown in the Figure 2, given a low-shot image from the novel set, the output feature after $CNN_0$ is composed of 128 feature maps with spatial size $28 \times 28$. The output will be warped in the 2D-Warping Layer, resulting in warped features $w^{(1)}, w^{(2)}, \ldots$ up to $w^{(n)}$ where each one has the same shape with the previous output. Each warped feature $w^{(n)}$ will further be forwarded through the remaining of the network to generate the feature vector $h^{(n)}$ and contribute to the optimization.

5. Training

We only use the low-shot dataset [3] associated to MS-Celeb-1M dataset [4] for training, without using any external data. It contains 21,000 persons, which is divided into two sets, a base set and a novel set. In the base set, there are 20,000 persons each with 50-100 images. In the novel set, there are 1,000 persons each with 1 or 2 or 5 images regarding different training protocols. Same with [3] we only use the base set of 20,000 classes to train a standard residual network with 34 layers [10] as our baseline. We start by training a baseline model with 20, 000 classes using only the softmax cross-entropy loss. The baseline itself has already achieved the accuracy of 98.95% on LFW dataset. For face verification on LFW [11] and YTF [26], we regard this baseline model as a constant starting point then fine-tuning the model under the joint supervision of the center loss and the modified fisher loss, respectively.

For low-shot learning task, we fine-tuning the baseline with a new 21, 000-class classifier using the training data from both the base set and the novel set. Additionally, we add a 2D-Warping Layer during the fine-tuning to generate more features for the novel set if there are low-shot images within a mini-batch. We then fine-tuning the model (which includes the 2D-Warping Layer) again by joint supervision of the modified fisher loss.

During the training, the face images were center cropped to $224 \times 224$ RGB images. As in [25] we normalize the images by subtracting the value 127.5 from all pixels followed by a division by the value 128. To add more variation to the training data each image is flipped horizontally with a probability of 50%. The training is done using the standard backpropagation with momentum set to 0.9 [14]. For regularization, the weight decay is set to $5 \times 10^{-4}$ and the dropout ratio is set to 0.3. The weight initialization for the baseline model is done as in [9] and all biases are set to zero. We use a batch-size of 128 and an initial learning rate of $1 \times 10^{-1}$, which is gradually divided by 0.1 after 3 epochs. A baseline training is finished after 9 epochs. For the fine-tuned models we keep the batch-size of 128. For the face verification task, we start with a learning rate of $1 \times 10^{-2}$ and use constant values of $\lambda$ and $\alpha$ for center loss and fisher loss throughout the experiments. The fine-tuning procedure again takes 9 epochs and the hyperparameter values are $\lambda = 0.003$ and $\alpha = 0.5$, the same as used in [25]. For the low-shot learning, we fine-tuning the baseline model with one image per novel class training protocol. For convenience, we refer it to top-1 training protocol later. We preprocess the dataset by duplicating each image 10 times, in order to somehow balance the data at the beginning. That leads to 10 images per novel class before the fine-tuning. We start fine-tuning the the baseline with an additional 2D-Warping Layer with the number of warped images $n = 4$ given a specified warp range and penalty weight (the de-
Table 1: Results on LFW.

| Method          | Training images | 2D alignment | Acc. [%] |
|-----------------|-----------------|--------------|----------|
| VGG-Face [17]   | 2.6M            | yes          | 98.95    |
| FaceNet [20]    | 200M            | no           | 98.87    |
| FaceNet [20]    | 200M            | yes          | 99.63    |
| L2 softmax [19] | 3.7M            | yes          | 99.60    |
| Center Loss [25]| 0.7M            | yes          | 99.25    |
| Baseline        | 1.16M           | yes          | 98.95    |
| Center Loss     | 1.16M           | yes          | 99.22    |
| Fisher Loss     | 1.16M           | yes          | 99.27    |


details of different configurations are shown in Table 3) for 7 epochs (by using learning rate of $1 \times 10^{-2}$ at the beginning of the fine-tuning then decrease to $1 \times 10^{-3}$ after 4 epochs), then we fine-tune the model again by adding an additional supervision of center loss or fisher loss with a fixed learning rate $1 \times 10^{-2}$ for 4 epochs. For all experiments the fisher loss we subsample 128 pairs in each iteration and set $\beta = 0.01$.

6. Experimental Results

We compare the proposed method on recently reported face verification dataset LFW [11], YouTube Face [26] and the development set of MS-Celeb-1M dataset [4] for the low-shot learning task [3].

6.1. Labeled Faces in the Wild

The Labeled Faces in the Wild (LFW) [11] is a very popular benchmark for unconstrained face verification. It consists of 13233 images for 5749 identities in total. The images have been collected in unconstrained conditions as long as they were detectable by a face detector. This leads to a database with a wide range of variations, such as lighting, facial expression or pose. The evaluation is done using 10-fold cross validation with fixed splits provided by the database authors. Each split consists of 600 pairs, where half of them are positive pairs (same identity) and the other half are negative pairs (different identities). The database authors define several protocols that differ by what type of training data can be used. As most deep learning methods we use the unrestricted, labeled outside data protocol. We use the original center-cropped images as model input and receive a 512-dimensional feature representation as output. We do this for the original image and a horizontally flipped version and use the mean of the two resulting feature vectors as final representation. We use public available MTCNN tools [29] for face alignment during the evaluation. The similarity scores are determined using the Euclidean distance.

The results are given in Table 1. We report the accuracy for our baseline model trained with softmax cross-entropy alone, the model fine-tuned by adding the center loss and the fisher loss. The gain by adding center loss or fisher loss is obvious, from 98.95% to 99.22% and 99.27%, respectively. The improvement of fisher loss compared to center loss is relatively small (99.22% vs. 99.27%). We explain that it may because the baseline model is already getting saturated working as a robust feature extractor so we cannot arrive at a higher accuracy. Note that this is achieved by using different 2D alignment tools since the alignment tools for the training set is not public available now.

The comparison to the state-of-the-art methods shows that our results are competitive. We only include the results most relevant to our work, since there are too many to include them all. The gap between the best-performing methods and ours could be explained with our limited amount of data and different 2D alignment methods. Additionally, the fisher loss could also be combined with the $L2$ softmax loss presented in [19], or an additional triplet embedding layer could be learned as it is done for VGG-Face [17] to boost performance further.

6.2. YouTube Faces

Similar to the LFW [11] database the YouTube Faces (YTF) [26] database is a popular benchmark for unconstrained face verification. However, instead of single still images, full videos are compared. In total there are 3425 videos for 1595 identities. On average, the videos consist of 181.3 frames.

Similar with LFW, the evaluation is done also using a 10-fold cross validation where the splits are given with the database. Each split contains 250 pairs where the identity of both videos is the same and 250 pairs with different identities. We use the given bounding box data to crop the images, but expand the bounding box by a factor of 1.2. We use the public available aligned YouTube Faces dataset for evaluation.

To compare two face videos it is a common approach to use the scores of a face or facial landmark detector to find the best frames in a video, which are then used for evaluation. However, since we do not have such scores available, we instead use the concept of softmax operator as proposed in [16] in the context of template matching. Given two sets of images $P = \{x_1, ..., x_p\}$ and $Q = \{x_1, ..., x_q\}$, the similarity $s(P, Q)$ can be defined similar to [16]

$$s(P, Q) = \frac{1}{11} \sum_{\gamma=-10}^{0} s_\gamma(P, Q)$$ (9)
Table 2: Results on YTF.

| Method          | Training images | Acc. [%] |
|-----------------|-----------------|----------|
| VGG-Face [17]   | 2.6M            | 91.6     |
| VGG-Face + Triplet Loss[17] | 2.6M            | 97.3     |
| FaceNet [20]    | 200M            | 95.1     |
| L2 softmax [19] | 3.7M            | 95.5     |
| Center loss [25]| 0.7M            | 94.9     |
| Baseline        | 1.16M           | 92.28    |
| Center Loss     | 1.16M           | 94.1     |
| Fisher Loss     | 1.16M           | 94.3     |

where

\[ s_y(P, Q) = \sum_{p \in P} \sum_{q \in Q} e^{\gamma s(x_p, x_q)} \]

In [16] the two sets of images are given as templates, which is a mixed collection of still images and videos. Here we just have to compare two videos, so the sets are given by the frames. To save some computational complexity we randomly sample 128 frames per video. Another notable difference to [16] is that we use the Euclidean distance instead of using Cosine distance as similarity measure \( s(x_p, x_q) \) between two images (in order to make it consistent with the LFW evaluation) and therefore select \( \gamma \) to be in the range of \([-10, ..., 0]\).

The results are given in Table 2. Again, with the accuracy of 94.30% the fisher loss achieve a slightly better result than our center loss model and it is also a competitive result compared to the other state-of-the-art methods. Considering the alignment tools for the training set and testing set are different, they both are slightly worse than the result of center loss reported in [25]. That could because using different alignment tools leads to sub-optimal feature representation especially in terms of the video frames in unconstrained conditions.

6.3. MS-Celeb-1M

The MS-Celeb low-shot dataset [3] consists of two parts, a base set and a novel set. To guarantee the data accuracy, the dataset has been cleaned by running algorithms to remove the outliers. After the cleaning, there is not too much noise label in the base set and it can be regarded as a large-scale dataset for the general face representation learning. In the base set, there are 20,000 persons, each person has 50-100 images for training and about 5 images for testing. Compared to the base set, the novel set is mainly used for low-shot feature learning. It includes 1,000 persons, each person has 1/2/5 images for training depending on the different training protocols and 20 images for testing. Since we don’t have public available labels for the test set, instead, we report our evaluation results based on the development set with provided labels. The development set includes 20,000 images for the normal classes, and 5,000 images for the novel classes. We only report the coverage rate at precision 99% for the novel classes since in all our experiments the top-1 accuracy for the normal classes is around 99.8%.

The results of top-1 training protocol (one image per novel class) are listed in Table 3. Similar with [3], we use the coverage rate at precision 99% as the evaluation metrics. All different settings are listed in Table 3.

The baseline (n=0) in Row 1 is achieved by at first training a baseline model with 20,000 classes as a feature extractor, then fine-tuning the model with a 21,000 new classifier based on lower learning rate. n=0 means without using any generated warped features from the 2D-Warping Layer, only the original ones are used. It achieves the C@99% = 51.74%.

Row 2 – 5 show the results with new generated warped features in the 2D-Warping Layer during the fine-tuning. The setting in the Row 2 of Table 3 is defined with the number of warping features \( n = 4 \) and warp range \( \Delta = 0 \). According to Formula (8), warp range \( \Delta = 0 \) means the pixel location \((i,j)\) is same with the warped pixel location \(w_{ij} = (u,v)\). In other words, each one-shot feature in the mini-batch is just copied \(n = 4\) times during the training, without any warping transformation. Since the feature which belongs to the novel class gets boosted 4 times during the training, there are more duplicated features for the novel classes so that the unbalanced data problem can be further compensated. With this setting the model get bootstrapped a lot from C@99% = 51.74% to C@99% = 71.54%.

Compared to Row 2, Row 3 – 5 in Table 3 show results based on warp range \( \Delta = 1 \) with different penalty weight \( \alpha \). The warp range \( \Delta \) is fixed to 1 because there are only warped features with a relatively small spatial size \(28 \times 28\) be generated, in which case even a small warp range would influence the generation quality a lot. E.g., if the warp range is set too large, it can not guarantee the smoothness of the generated features. When the penalty weight \( \alpha \) is set to 0 (See Row 3), according to Formula (8) there is only feature-level displacement be used, which means it completely ignores the pixel-level deviation so that it could be mapped to anywhere. This leads to the loss of the original feature structures since the important local context could not be preserved anymore. As the result, it achieves a poorer result of C@99% = 63.98% compared to Row 2. In Row 4 we show the result with \( \alpha = 0.5 \), in which case the pixel-level displacement is considered as an additional warping restriction. Instead of ignoring the pixel-level displacement, the warped features now can preserve more local context.
Figure 3: Warped feature maps from the 2D-Warping Layer. We take the feature maps with highest mean activation from each warped features. The warped feature maps in the first row are generated without considering the pixel-level displacement penalty (corresponding to parameter settings in Row 3 of Table 3). The feature maps in the second row are generated considering the pixel-level displacement constraint (corresponding to the parameter settings in Row 5 of Table 3). Compared to the feature maps in the first row, the warped feature maps in the second row can preserve more local image structure due to the pixel-level deviation penalty.

Moreover, it gives more stable and learnable information to the network compared to just copying the features for $n$ times during the training. As the result the coverage rate at precision $= 99\%$ increase to $73.18\%$, which is better than all previous ones (Row 1-3). Row 5 shows our best result without using an additional supervision by setting $\alpha = 1$ (with a stronger pixel-level deviation restriction compared to Row 4), which achieves $C@99\% = 73.88\%$. We show the warped feature maps generated with the setting in Row 3 in the first row of Figure 3b and the setting in Row 5 in the second row of the same figure. We plot the feature maps with the highest mean activation from each warped features. It is obvious that the feature maps generated without considering the pixel-level displacement (with $\alpha = 0$) can not preserve the local image structure and it totally ignores the neighboring information. Compared to it, the warped feature maps in the second row (with $\alpha = 1$) are generated with the pixel-level displacement constraint, which leads to a better classification result due to the forced smoothness by the pixel-level deviation penalty.

Row 7 and Row 8 show the results for fine-tuning (abbr. f.t.) from the previous best model (in our case we select the model with hyperparameter listed in Row 5 of Table 3) with an additional supervision. As the result, fine-tuning with center loss increase from $C@99\% = 73.88\%$ to $76.30\%$, and fine-tuning with fisher loss increases to $77.92\%$. Although they both get a significant improvement, the model fine-tuned with fisher loss (Row 8) is again better than center loss (Row 7). The reason could be during the one-shot learning there are only a few samples (considering the data augmentation) in a novel class contribute to the center loss, if those features itself are bad, e.g. if those novel features get mixed up with the features of the normal classes, the center loss part simply will not help the optimization too much according to its definition, instead, the inter-class part of the fisher loss learns to increase the inter-class distance, making it easier to separate the one-shot features from the features of normal classes. We think this is the crucial reason why it gets a better result than model fine-tuned with center loss.

7. Conclusion

In this paper, we did minor change on established fisher loss [7]. By this manner, the optimization is not restricted by the margin. The experimental results show that jointly training with modified fisher loss the model achieves comparable results compared to the state-of-the-art methods evaluated on LFW [11] and YTF [26] databases. Furthermore, We extend our work to study one-shot face recognition problem. Specifically, followed by [6], we design a 2D-Warping Module to generate warped features of the novel classes during the training, in order to give more informative training samples to the novel classes. Experimental results on MS-Celeb-1M one-shot datasets [4, 3] show that with the proposed module, it gets a significant improvement over the baseline. Moreover, we fine-tuning the model with modified fisher loss to learn more discriminative one-shot features, which leads to a further improvement. In future work, for modified fisher loss, we would like to define constraints to select harder but more informative pairs rather than just randomly selecting them to contribute to the inter-class part. Moreover, for the 2D-Warping Module, we would like to consider more on 2D-dependencies during the warping in order to better capture the local image structures while keeping the overall smoothness of the generated features.

References

[1] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman. Eigenfaces vs. fisherfaces: Recognition using class specific linear
projection. *IEEE Transactions on pattern analysis and machine intelligence*, 19(7):711–720, 1997. 2

[2] T. Gass, L. Pishchulin, P. Dreauw, and H. Ney. Warp that smile on your face: optimal and smooth deformations for face recognition. In *FG*, pages 456–463, 2011. 2

[3] Y. Guo and L. Zhang. One-shot face recognition by promoting underrepresented classes. *arXiv preprint arXiv:1707.05574*, 2017. 1, 2, 4, 5, 6, 7

[4] Y. Guo, L. Zhang, Y. Hu, X. He, and J. Gao. MS-Celeb-1M: A dataset and benchmark for large scale face recognition. In *European Conference on Computer Vision*. Springer, 2016. 4, 5, 7

[5] R. Hadsell, S. Chopra, and Y. LeCun. Dimensionality reduction by learning an invariant mapping. In *Computer vision and pattern recognition, 2006 IEEE computer society conference on*, volume 2, pages 1735–1742. IEEE, 2006. 1, 2

[6] H. Hanselmann and H. Ney. Speeding up 2d-warping for pose-invariant face recognition. In *Automatic Face and Gesture Recognition (FG)*, 2015 11th IEEE International Conference and Workshops on, volume 1, pages 1–7. IEEE, 2015. 1, 2, 3, 7

[7] H. Hanselmann, S. Yan, and H. Ney. Deep fisher faces. In *British Machine Vision Conference (BMVC)*, 2017. 1, 2, 7

[8] B. Hariharan and R. Girshick. Low-shot visual recognition by shrinking and hallucinating features. In *ICCV*, 2017. 2

[9] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015. 4

[10] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016. 4

[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, Technical Report 07-49, University of Massachusetts, Amherst, 2007. 1, 4, 5, 7

[12] D. Keysers, T. Deselaers, C. Gollan, and H. Ney. Deformation models for image recognition. *IEEE T-PAMI*, pages 1422–1435, 2007. 3

[13] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 1

[14] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989. 4

[15] Y. LeCun, C. Cortes, and C. J. Burges. The mnist database of handwritten digits, 1998. 3

[16] I. Masi, A. T. Trm, T. Hassner, J. T. Leksut, and G. Medioni. Do we really need to collect millions of faces for effective face recognition? In *European Conference on Computer Vision*, pages 579–596. Springer, 2016. 5, 6

[17] O. M. Parkhi, A. Vedaldi, and A. Zisserman. Deep face recognition. In *Proceedings of the British Machine Vision Conference 2015, BMVC 2015, Swansea, UK, September 7-10, 2015*, pages 41.1–41.12, 2015. 1, 2, 5, 6

[18] L. Pishchulin, T. Gass, P. Dreuw, and H. Ney. Image warping for face recognition: From local optimality towards global optimization. *Pattern Recognition*, 45(9):3131–3140, 2012. 3

[19] R. Ranjan, C. D. Castillo, and R. Chellappa. L2-constrained softmax loss for discriminative face verification. *arXiv preprint arXiv:1703.09507*, 2017. 5, 6

[20] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 815–823, 2015. 1, 2, 5, 6

[21] E. Simo-Serra, E. Trulls, L. Ferraz, I. Kokkinos, P. Fua, and F. Moreno-Noguer. Discriminative learning of deep convolutional feature point descriptors. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2015. 2

[22] Y. Sun, D. Liang, X. Wang, and X. Tang. Deepid3: Face recognition with very deep neural networks. *arXiv preprint arXiv:1502.00873*, 2015. 2

[23] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf. Deepface: Closing the gap to human-level performance in face verification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1701–1708, 2014. 1

[24] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, and D. Wierstra. Matching networks for one shot learning. In *NIPS*, 2016. 2

[25] Y. Wen, K. Zhang, Z. Li, and Y. Qiao. A discriminative feature learning approach for deep face recognition. In *European Conference on Computer Vision*, pages 499–515. Springer, 2016. 1, 2, 3, 4, 5, 6

[26] L. Wolf, T. Hassner, and I. Maoz. Face recognition in unconstrained videos with matched background similarity. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 529–534. IEEE, 2011. 1, 4, 5, 7

[27] S. Zagoruyko and N. Komodakis. Learning to compare image patches via convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4353–4361, 2015. 2

[28] J. Zbontar and Y. LeCun. Stereo matching by training a convolutional neural network to compare image patches. *Journal of Machine Learning Research*, 17:1–32, 2016. 2

[29] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao. Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE Signal Processing Letters*, 23(10):1499–1503, Oct 2016. 5