Rethinking Feature Distribution for Loss Functions in Image Classification

Weitao Wan[1*]   Yuanyi Zhong[1,2*†]   Tianpeng Li[1]   Jiansheng Chen[1‡]

[1]Department of Electronic Engineering, Tsinghua University, Beijing, China
[2]Department of Computer Science, University of at Urbana-Champaign, Illinois, USA

wwt16@mails.tsinghua.edu.cn   yuanyiz2@illinois.edu
ltp16@mails.tsinghua.edu.cn   jschenthu@mail.tsinghua.edu.cn

Abstract

We propose a large-margin Gaussian Mixture (L-GM) loss for deep neural networks in classification tasks. Different from the softmax cross-entropy loss, our proposal is established on the assumption that the deep features of the training set follow a Gaussian Mixture distribution. By involving a classification margin and a likelihood regularization, the L-GM loss facilitates both a high classification performance and an accurate modeling of the training feature distribution. As such, the L-GM loss is superior to the softmax loss and its major variants in the sense that besides classification, it can be readily used to distinguish abnormal inputs, such as the adversarial examples, based on their features' likelihood to the training feature distribution. Extensive experiments on various recognition benchmarks like MNIST, CIFAR, ImageNet and LFW, as well as on adversarial examples demonstrate the effectiveness of our proposal.

1. Introduction

Recently, deep neural networks have substantially improved the state-of-the-art performances of various challenging classification tasks, including image based object recognition [17, 14, 10], face recognition [25, 36] and speech recognition [5, 6]. In these tasks, the softmax cross-entropy loss, or the *softmax loss* for short, has been widely adopted as the classification loss function for various deep neural networks [31, 10, 35, 19, 12]. For example in image classification, the affinity score of an input sample to each class is first computed by a linear transformation on the extracted deep feature. Then the posterior probability is modeled as the normalized affinity scores using the softmax function. Finally, the cross-entropy between the posterior probability and the class label is used as the loss function. The softmax loss has its probabilistic interpretation in that, for a large class of distributions, the posterior distribution complies with the softmax transformation of linear functions of the feature vectors [1]. It can also be derived from a binary Markov Random Field or a Boltzmann Machine model [3]. However, the relationship between the affinity score and the probability distribution of the training feature space is vague. In other words, for an extracted feature, its likelihood to the training feature distribution is not well formulated.

Several variants have been proposed to enhance the effectiveness of the softmax loss. The Euclidean distances between each pair [36] or among each triplet [25] of extracted features are added as an additional loss to the softmax loss. Alternatively, in [32] the Euclidean distance between each feature vector and its class centroid is used. However, under the softmax loss formulation, the cosine distance based similarity metrics is more appropriate, indicating that using the Euclidean distance based additional losses may not be the most ideal choice. Based on this understanding, an angular distance based margin is introduced in [22] to force extra intra-class compactness and inter-class separability, leading to better generalization of the trained models. Nevertheless, the softmax loss is still indispensable and mostly dominates the training process in these proposals. Therefore, the probabilistic modeling of the training feature space is still not explicitly considered.

In this paper we propose a Gaussian Mixture loss (GM loss) under the intuition that it is reasonable as well as tractable to assume the learned features of the training set to follow a Gaussian Mixture (GM) distribution, with each component representing a class. As such, the posterior probability can be computed using the Bayes' rule. The classification loss is then calculated as the cross-entropy between the posterior probability and the corresponding class labels. To force the training samples to obey the assumed GM distribution, we further add a likelihood regularization term to the classification loss. As such, for a well trained model, the probability distribution of the training features can now be
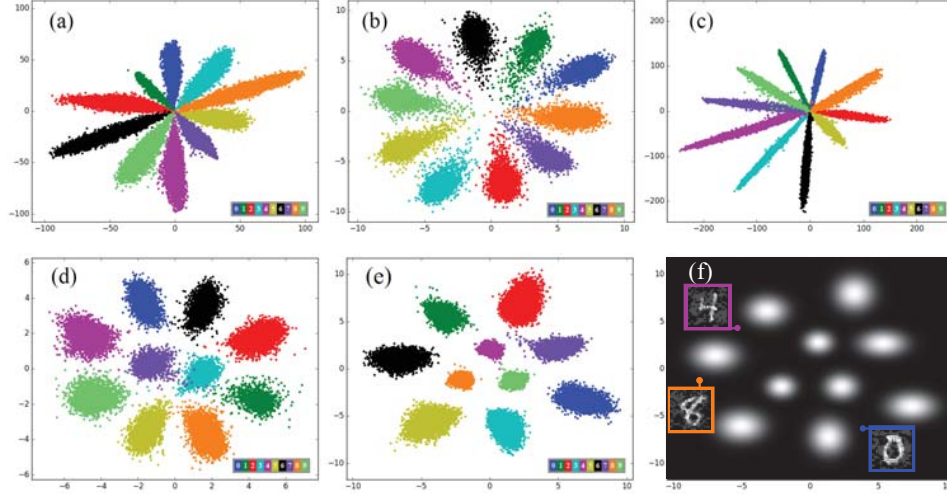

Figure 1. Two-dimensional feature embeddings on MNIST training set. (a) Softmax loss. (b) Softmax loss + center loss [32]. (c) Large-margin softmax loss [22]. (d) GM Loss without margin ($\alpha = 0$). (e) Large-margin GM loss ($\alpha = 1$). (f) Heatmap of the learned likelihood corresponding to (e). Higher values are brighter. Several adversarial examples generated by the Fast Gradient Sign Method [8] have extremely low likelihood according to the learned GM distribution and thus can be easily distinguished. This figure is best viewed in color.

explicitly formulated. It can be observed from Fig. 1 that the learned training features spaces using the proposed GM loss are intrinsically different from those learned using the softmax loss and its invariants, by approximately following a GM distribution.

The GM loss is not just an alternative, it bears several essential merits comparing to the softmax loss and its invariants. First, incorporating a classification margin into the GM loss is simple and straightforward so that there is no need to introduce an additional complicated distance function as is practiced in the large-margin softmax loss [22]. Second, it can be proved that the center loss [32] is formally equivalent to a special case of the likelihood regularization in the GM loss. However, the classification loss and the regularization now share identical feature distance measurements in the GM loss since they are both induced from the same GM assumption. Last but not the least, in addition to the classification result, the GM loss can be readily used to estimate the likelihood of an input to the learned training feature distribution, leading to the possibility of improving the model's robustness, for example, towards adversarial examples.

We discuss mathematic details of the GM loss in Section 3. Extensive experimental results on object classification, face verification and adversarial examples are shown in Section 4. We conclude this work in Section 5 .

2. Related Work

The previous efforts for overcoming certain deficiencies of the softmax loss are inspiring. One of the most widely studied technical route is to explicitly encourage stronger intra-class compactness and larger inter-class separability while using the softmax loss. Y. Sun *et al.* introduced the contrastive loss in training a Siamese network for face recognition by simultaneously minimizing the distances between positive face image pairs and enlarging the distances between negative face image pairs by a predefined margin [36]. Similarly, F. Schroff *et al.* proposed to apply such inter sample distance regularizations on image triplets rather than on image pairs [25]. A major drawback of the contrastive loss and the triplet loss is the combinatoric explosion in the number of image pairs or triplets especially for large-scale data sets, leading to the significant increase in the required number of training iterations. The center loss proposed in [32] effectively circumvents the pair-wise or triplet-wise computation by minimizing the Euclidean distance between the features and the corresponding class centroids. However, such a formulation brings about inconsistency of distance measurements in the feature space. W. Liu *et al.* solved this problem by explicitly introduced an angular margin into the softmax loss through the designing of a sophisticated differentiable angular distance function [22]. Another technical route mainly aims at improving the numerical stability of the softmax loss. Along this line, the label smoothing [31] and the knowledge distilling [7] are two typical methods of which the basic idea is to replace the one-hot ground truth distribution with other distributions that are probabilistically more reasonable. An interesting recent work proposed by B. Chen *et al.* focused on mitigating the early saturation problem of the softmax loss by injecting annealed noise in the softmax function during each training iteration [2]. Generally speaking, all these works aim at improving the softmax loss rather than reformulating its fundamental assumption.

9118

It has been revealed that deep neural networks with high classification accuracies are vulnerable to adversarial examples [8]. Previous methods for solving this dilemma either directly included the adversarial samples in the training set [18] or introduced an new model for detecting the spoofing samples [24]. Intuitively, however, the features of adversarial examples should follow a probability distribution quite different from that of the learned training feature space. In other words, it is possible to better distinguish the adversarial examples if the distribution of the training feature space can be explicitly modeled.

3. Gaussian Mixture Loss

In this section, we will formulate the GM loss from a probability perspective. We will also describe how to efficiently add a classification margin to the GM loss, after which the likelihood regularization term in the GM loss is further discussed. The optimization of the GM loss is also presented.

3.1. Intuitions

Considering a K class classification task in which the softmax loss is used. For an input sample with x as its extracted deep feature vector, its posterior probability of belonging to a certain class $j \in [1, K]$ can be expressed by Eq. 1, in which the affinity score (logit) $f_k(x)$ is usually calculated by linearly transforming the feature vector x as is shown in Eq. 2. In practice, the linear functions of all the K classes are combined to form a linear transformation layer with all the w_k, b_k as the trainable parameters. A larger value of the affinity score $f_k(x)$ indicates a higher posterior probability of x belonging to the class k. However, $f_k(x)$ cannot be directly used to evaluate x's likelihood to the distribution of the training features which is not explicitly formulated at all.

$$p(j|x) = \frac{e^{f_j(x)}}{\sum_{k=1}^{K} e^{f_k(x)}} \tag{1}$$

$$f_k(x) = w_k^T x + b_k, k \in [1, K] \tag{2}$$

What is more, since $f_k(x)$ is computed through inner product, the similarity between features in the learned feature space should be measured using the cosine distance. However, in the Euclidean distance based regularization is more widely adopted in softmax variants probably due to its mathematical simplicity. For example, the Euclidean distance between the extracted feature and the corresponding class centroid was used to formulate the center loss $\mathcal{L}_C$ in Eq. 3 [32], in which N is the number of training samples; x_i and z_i are the extracted feature and the class label of the i-th sample respectively; and μ_{z_i} is the feature centroid (mean) for class z_i. Intuitively, such a regularization should be more

reasonable if the similarity measurement can be coherent to that in the classification loss.

$$\mathcal{L}_C = \frac{1}{2} \sum_{i=1}^{N} \|x_i - \mu_{z_i}\|_2^2 \tag{3}$$

3.2. GM loss formulation

Different from the softmax loss, we hereby assume that the extracted deep feature x on the training set follows a Gaussian mixture distribution expressed in Eq. 4, in which μ_k and Σ_k are the mean and covariance of class k in the feature space; and $p(k)$ is the prior probability of class k.

$$p(x) = \sum_{k=1}^{K} \mathcal{N}(x; \mu_k, \Sigma_k) p(k) \tag{4}$$

Under such an assumption, the conditional probability distribution of a feature x_i given its class label $z_i \in [1, K]$ can be expressed in Eq. 5. Consequently, the corresponding posterior probability distribution can be expressed in Eq. 6.

$$p(x_i|z_i) = \mathcal{N}(x_i; \mu_{z_i}, \Sigma_{z_i}) \tag{5}$$

$$p(z_i|x_i) = \frac{\mathcal{N}(x_i; \mu_{z_i}, \Sigma_{z_i}) p(z_i)}{\sum_{k=1}^{K} \mathcal{N}(x_i; \mu_k, \Sigma_k) p(k)} \tag{6}$$

As such, a *classification loss* $\mathcal{L}_{cls}$ can be computed as the cross-entropy between the posterior probability distribution and the one-hot class label as is shown in Eq. 7, in which the indicator function $\mathbb{1}()$ equals 1 if z_i equals k; or 0 otherwise.

$$\begin{aligned} \mathcal{L}_{cls} &= -\frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} \mathbb{1}(z_i = k) \log p(k|x_i) \\ &= -\frac{1}{N} \sum_{i=1}^{N} \log \frac{\mathcal{N}(x_i; \mu_{z_i}, \Sigma_{z_i}) p(z_i)}{\sum_{k=1}^{K} \mathcal{N}(x_i; \mu_k, \Sigma_k) p(k)} \end{aligned} \tag{7}$$

Optimizing the classification loss only cannot explicitly drive the extracted training features towards the GM distribution. For example, a feature x_i can be far away from the corresponding class centroid μ_{z_i} while still being correctly classified as long as it is relatively closer to μ_{z_i} than to the feature means of the other classes. To solve this problem, we further introduce a *likelihood regularization* term for measuring to what extent the training samples fit the assumed distribution. The likelihood for the complete data set $\{X, Z\}$ is expressed in Eq. 8. We define the likelihood regularization term as the negative log likelihood shown in Eq. 9. By reasonably assuming constant prior probabilities $p(z_i)$, the likelihood regularization $\mathcal{L}_{lkd}$ can be simplified as Eq. 10.

$$p(\mathrm{X}, \mathrm{Z}|\mu, \Sigma) = \prod_{i=1}^{N} \prod_{k=1}^{K} \mathbb{1}(z_i = k) \mathcal{N}(x_i; \mu_{z_i}, \Sigma_{z_i}) p(z_i) \tag{8}$$

$$\log p(X, Z | \mu, \Sigma) = -\sum_{i=1}^{N} (\log \mathcal{N}(x_i; \mu_{z_i}, \Sigma_{z_i}) + \log p(z_i))$$
$$(9)$$

$$\mathcal{L}_{lkd} = -\sum_{i=1}^{N} \log \mathcal{N}(x_i; \mu_{z_i}, \Sigma_{z_i}) \qquad (10)$$

Finally the proposed GM loss $\mathcal{L}_{GM}$ is defined in Eq. 11, in which λ is a non-negative weighting coefficient.

$$\mathcal{L}_{GM} = \mathcal{L}_{cls} + \lambda \mathcal{L}_{lkd} \qquad (11)$$

By definition, for the training feature space, the classification loss $\mathcal{L}_{cls}$ is mainly related to its discriminative capability while the likelihood regularization $\mathcal{L}_{lkd}$ is related to its probabilistic distribution. Under the GM distribution assumption, $\mathcal{L}_{cls}$ and $\mathcal{L}_{lkd}$ share all the parameters.

3.3. Large-Margin GM Loss

It has been widely recognized in statistical machine learning that large classification margin on the training set usually helps generalization, which is also believed to be applicable in deep learning [28, 22]. Denote x_i's contribution to the classification loss to be $\mathcal{L}_{cls,i}$, of which an expansion form is in Eq. 12 and Eq. 13.

$$\mathcal{L}_{cls,i} = -\log \frac{p(z_i)|\Sigma_{z_i}|^{-\frac{1}{2}} e^{-d_{z_i}}}{\sum_k p(k)|\Sigma_k|^{-\frac{1}{2}} e^{-d_k}} \qquad (12)$$

$$d_k = (x_i - \mu_k)^T \Sigma_k^{-1} (x_i - \mu_k)/2 \qquad (13)$$

Since the squared Mahalanobis distance d_k is by definition non-negative, a classification margin $m \geq 0$ can be easily introduced to achieve the large-margin GM loss as in Eq. 14. Obviously, adding the classification margin to the GM loss is more straightforward than to the softmax loss [22]. It should be emphasized that such a simple formulation cannot be directly applied to the softmax loss since an inner product can be negative, whereas a margin generally has to be non-negative to make sense.

$$\mathcal{L}_{cls,i}^m = -\log \frac{p(z_i)|\Sigma_{z_i}|^{-\frac{1}{2}} e^{-d_{z_i} - m}}{\sum_k p(k)|\Sigma_k|^{-\frac{1}{2}} e^{-d_k - \mathbb{1}(k=z_i)m}} \qquad (14)$$

To understand m's role in the large-margin GM loss, one may consider the simplest case in which $p(k)$ and Σ_k are identical for all the classes. Then x_i is classified to the class z_i if and only if Eq. 15 holds, indicating that x_i should be closer to the feature mean of class z_i than to that of the other classes by at least m.

$$e^{-d_{z_i} - m} > e^{-d_k} \Longleftrightarrow d_k - d_{z_i} > m \quad , \forall k \neq z_i \quad (15)$$

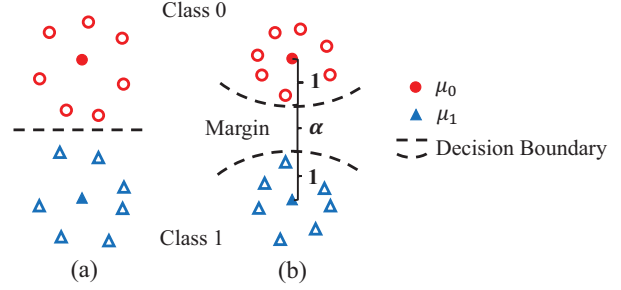To design the margin, we adopt an adaptive scheme by letting the value of m to be proportional to each sample's

Figure 2. A geometry interpretation of the relationship between α and the margin size in the training feature space using (a) GM loss without margin $\alpha = 0$; (b) large-margin GM loss with $\alpha > 0$.

distance to its corresponding class feature mean, i.e., $m = \alpha d_{z_i}$, in which α is a non-negative parameter controlling the size of the expected margin between two classes on the training set. Fig. 2 shows a schematic interpretation of α; and Fig. 1 (d) and (e) illustrate how the training feature space changes when increasing α from 0 to 1.

3.4. A Discussion on $\mathcal{L}_{lkd}$

Although the likelihood regularization $\mathcal{L}_{lkd}$ defined in Eq. 10 is proposed from a probability perspective, it has a strong connection with the empirical center loss $\mathcal{L}_C$ defined in Eq. 3 [32] as is described in **Lemma 1**, of which the proof is quite straightfoward.

Lemma 1. *If $\Sigma_k = I$ (identity matrix), $p(k) = 1/K, \forall k \in [1, K]$, the center loss $\mathcal{L}_C$ and the likelihood regularization $\mathcal{L}_{lkd}$ satisfy Eq. 16, in which D is the feature dimension.*

$$\mathcal{L}_{lkd} = \mathcal{L}_C + \frac{N}{2} D \log(2\pi) \qquad (16)$$

Lemma 1 shows that $\mathcal{L}_C$ is identical to $\mathcal{L}_{lkd}$ except for a constant under certain conditions. In other words, the center loss [32] is basically equivalent to a special case of the proposed likelihood regularization. This indicates that it might be more appropriate to use the center loss, or the proposed likelihood $\mathcal{L}_{lkd}$ as regularization in a GM distributed feature space, as is practiced in this work.

More importantly, $\mathcal{L}_{lkd}$ can be readily used to estimate the likelihood of a sample feature to the learned GM distribution. Simply put, a model trained using our GM loss can now both generate a classification result and provide a likelihood estimation. In case that the likelihood is too low, one may refuse to make the classification decision. Such a choice may be favorable, for example, when an adversarial example [8] is generated to attack the trained classification model. In fact, the center loss $\mathcal{L}_C$ could also be used to estimate such a likelihood. However, when being combined with the softmax loss during training, the center loss may produce inaccurate likelihood estimation since the generated training feature space probably deviates from the GM distribution.

3.5. Optimization

The GM loss can be optimized using the typical stochastic gradient descent (SGD) algorithm. In practice, updating the covariance matrix with gradient descent is feasible but may suffer from singularity problems. Hence, for simplicity, we assume that the covariance matrix Σ_k is diagonal, denoted by Λ_k; and the prior probability $p(k) = 1/K$. As such, the contribution of a sample x_i to the large-margin GM loss can be rewritten in Eq. 17 and Eq. 18.

$$\mathcal{L}_{GM,i}^m = -\log \frac{|\Lambda_{z_i}|^{-\frac{1}{2}} e^{-d_{z_i}(1+\alpha)}}{\sum_k |\Lambda_k|^{-\frac{1}{2}} e^{-d_k(1+\mathbb{1}(k=z_i)\alpha)}} \quad (17)$$
$$+ \lambda(d_{z_i} + \frac{1}{2}\log|\Lambda_{z_i}|)$$

$$d_k = \frac{1}{2}(x_i - \mu_k)^T \Lambda_k^{-1}(x_i - \mu_k), \ k \in [1, K] \quad (18)$$

The gradient computations for the GM loss of the i-th sample are given in Eqs. 19 to 23. For conciseness, we denote $p(k|x_i)$ as p_k and $(x_i - \mu_k)(x_i - \mu_k)^T$ as C_k in all these equations.

$$\frac{\partial \mathcal{L}_{GM,i}^m}{\partial \mu_{z_i}} = \left[(1-p_{z_i})(1+\alpha) + \lambda\right]\Lambda_{z_i}^{-1}(\mu_{z_i} - x_i) \quad (19)$$

$$\frac{\partial \mathcal{L}_{GM,i}^m}{\partial \mu_k} = p_k \Lambda_k^{-1}(x_i - \mu_k), \ \ \forall k \neq z_i \quad (20)$$

$$\frac{\partial \mathcal{L}_{GM,i}^m}{\partial \Lambda_{z_i}} = -\frac{1}{2}\Big[\big((1-p_{z_i})(1+\alpha) + \lambda\big)\Lambda_{z_i}^{-1}C_k - \quad (21)$$
$$(1 - p_{z_i} + \lambda)I\Big]\Lambda_{z_i}^{-1}$$

$$\frac{\partial \mathcal{L}_{GM,i}^m}{\partial \Lambda_k} = -\frac{1}{2}p_k(I - \Lambda_k^{-1}C_k)\Lambda_k^{-1}, \ \ \forall k \neq z_i \quad (22)$$

$$\frac{\partial \mathcal{L}_{GM,i}^m}{\partial x_i} = \left[(1-p_{z_i})(1+\alpha) + \lambda\right]\Lambda_{z_i}^{-1}(x - \mu_{z_i}) \quad (23)$$
$$- \sum_{k \neq z_i} p_k \Lambda_k^{-1}(x_i - \mu_k)$$

4. Experiments

Two sets of experiments are presented in this section. In the first set, we conduct the image classification and face verification experiments to verify the effectiveness of the large-margin GM loss (L-GM loss for short). We report mean and standard deviation of 3 tries. In the second set, we demonstrate the feasibility of distinguishing adversarial examples using the likelihood regularization term $\mathcal{L}_{lkd}$. All experiments are carried out using the *Caffe* framework [33] on NVIDIA TitanX GPUs.

Loss Functions	2-D (%)	100-D (%)
Center [32]	1.45 ± 0.01	0.47 ± 0.01
L-Softmax[22]	1.30 ± 0.02	0.43 ± 0.01
Softmax	1.82 ± 0.01	0.68 ± 0.01
L-GM ($\alpha = 0$))	1.44 ± 0.01	0.49 ± 0.01
L-GM ($\alpha = 0.3$)	1.32 ± 0.01	0.42 ± 0.02
L-GM ($\alpha = 1.0$)	$\mathbf{1.17 \pm 0.01}$	$\mathbf{0.39 \pm 0.01}$

Table 1. Recognition error rates (%) on MNIST test set using a 6-layer CNN with different loss functions.

For the margin parameter α, a larger value may lead to a more difficult optimization objective. Therefore intuitively, α should be smaller when the number of classes gets larger. In our experiments, we empirically set α to 1.0, 0.3, 0.1, 0.01 and 0.01 for MNIST, CIFAR-10, CIFAR-100, ImageNet and face verification, respectively. Also, we set the likelihood regularization parameter λ to a small value, *e.g.* 0.1 in our experiments, so that the likelihood regularization starts to play a major role when the training accuracy is approaching saturation, or when p_{z_i} approaches 1.

4.1. Image Classification

MNIST We first compare the softmax loss, the center loss (with the softmax loss) [32], the large-margin softmax loss (L-Softmax loss for short) [25] and the L-GM loss by visualizing their learned 2D feature spaces for the MNIST Handwritten Digit dataset [20]. We adopt a network with 6 convolution layers and a fully connected layer with a two dimensional output. The feature embeddings on the training set with different loss functions are illustrated in Fig. 1. As we can see, different from the softmax loss and its variants, the features generated using the L-GM loss roughly follow the GM distribution, which is consistent with the assumption. The heatmap of the learned likelihood is shown in Fig. 1(f). Also, as is shown in Fig. 1 (d)-(e), with an increasing α, larger margin sizes can be observed among different classes.

For the quantitative evaluation, we also increase the output dimension of the fully connected layer from 2 to 100 and add a ReLU activation after it. For fair comparison, we train the same network with different loss functions using identical training parameters including the learning rate, weight decay, etc.. The classification accuracies on the test set are presented in Table 1.

CIFAR CIFAR-10 and CIFAR-100 [16] each consists of of 32×32 pixel colored images, with 50,000 training images and 10,000 testing images. We adopt the standard data augmentation scheme including mirroring and 32×32 random cropping after 4 pixel zero-paddings on each side [10, 22].

For CIFAR-10, We train the ResNet [10] of depth 20, 56 and 110 with different loss functions. The networks are trained with a batch size of 128 for 300 epochs; and the

Loss Functions	ResNet-20	ResNet-56	ResNet-110
Softmax [10]	8.75 ± 0.04	6.97 ± 0.05	6.43 ± 0.04
Center [32]	7.77 ± 0.05	5.94 ± 0.02	5.32 ± 0.03
L-Softmax [22]	7.73 ± 0.03	6.05 ± 0.04	5.79 ± 0.02
L-GM($\alpha = 0.3$)	$\textbf{7.21} \pm \textbf{0.04}$	$\textbf{5.61} \pm \textbf{0.02}$	$\textbf{4.96} \pm \textbf{0.03}$

Table 2. Recognition error rates (%) on CIFAR-10 using ResNet models with different loss functions.

learning rate is set to 0.1 and then divided by 10 at the 150th epoch and the 225th epoch respectively. We use a weight decay of 5×10^{-4} and the Nesterov optimization algorithm [30] with a momentum of 0.9. The network weights are initialized using the method introduced in [9]. The recognition accuracies are shown in Table 2. Results in the first row were reported in the original RestNet paper [10]. For the center loss and the large-margin softmax loss, we train the models by ourselves since the ResNet was not used on CIFAR-10 in the original papers [32] and [22]. The proposed L-GM loss outperforms the softmax loss and its two variants for different ResNet models with various depths.

For CIFAR-100, we adopt the same CNN architecture used by the large-margin softmax loss [22], which follows the design philosophy of the VGG-net [27] consisting of 13 convolutional layers and 1 fully connected layer. Bach normalization [15] is used after each convolutional layer and no dropout is used. To achieve better recognition performances, we replace the fully connected layer in this network with Global Average Pooling [21]. We report the recognition performances with or without the data augmentation in Table 3, denoted by C100+ and C100 respectively. Several points can be observed from Table 3. First, the proposed L-GM loss consistently outperforms the softmax based losses on both C100+ and C100. Second, for the augmented data set C100+, increasing the margin parameter α consistently benefits the recognition performance. However, this is not true for C100 without data augmentation. This is probably related to the fact that the number of training samples for each object class is as low as 500 on C100. The margin size on the training set and the model generalization capability is less correlated.

Loss Functions	C100	C100+
Center [32]	24.85 ± 0.06	21.05 ± 0.03
L-Softmax [22]	24.83 ± 0.05	20.98 ± 0.04
Softmax	25.61 ± 0.07	21.60 ± 0.04
LGM($\alpha = 0.1$)	23.74 ± 0.08	20.94 ± 0.03
LGM($\alpha = 0.2$)	$\textbf{23.04} \pm \textbf{0.08}$	20.85 ± 0.04
LGM($\alpha = 0.3$)	23.80 ± 0.06	$\textbf{20.76} \pm \textbf{0.03}$

Table 3. Recognition error rates (%) on CIFAR-100 using a VGG-like 13 layer CNN with different loss functions.

ImageNet We investigate the performance on large-scale image classification using the ImageNet dataset [4]. We

perform experiments on ImageNet (ILSVRC2012) using ResNet-101 [10] combined with different loss functions. To make fair comparison, all the models are trained for 100 epochs on 6 Titan GPUs with a mini-batch size of 16 for each GPU. The learning rate is initialized as 0.01 and divided by 10 at the 50th epochs and 75th epochs respectively. We use a weight decay of 0.0002 and a momentum of 0.9; and no dropout [11] is used. We evaluate the performances for 1-crop and 10-crop practices on the ILSVRC2012 validation set. Results in Table 4 show that our proposal is also effective on the large-scale dataset.

Loss	1-crop		10-crop	
	top-1	top-5	top-1	top-5
Softmax	23.5 ± 0.2	7.55 ± 0.08	22.6 ± 0.2	6.92 ± 0.04
L-GM	$\textbf{22.7} \pm \textbf{0.2}$	$\textbf{7.14} \pm \textbf{0.08}$	$\textbf{21.9} \pm \textbf{0.1}$	$\textbf{6.05} \pm \textbf{0.03}$

Table 4. Error rates (%) on ILSVRC2012 validation set. For L-GM, we set α=0.01 and λ=0.1.

4.2. Face Verification

We conduct the face verification experiments on the Labeled Face in the Wild (LFW) dataset [13], which contains 13,233 face images from 5749 different identities with large variations in pose, expression and illumination. The officially provided 6,000 pairs are used for face verification test. We follow the standard *unrestricted, labeled outside data* protocol of LFW and use only the CASIA-WebFace dataset [34] for training. The CASIA-WebFace dataset consists of 494,414 face images from 10,575 subjects. The training and testing images are aligned using MTCNN [37] and resized to 128×128 pixel. A simple data augmentation scheme is adopted including horizontal mirroring and 120×120 random crop from the aligned 128×128 pixel face images.

We train the ResNet [10] based face recognition model with 27 convolutional layers. The PReLU activations [9] are used after each convolutional layer and no batch normalization or Dropout is used. We train with a batch size of 256 for 20 epochs. The learning rate is initially set to 0.1 and divided by 10 at the 10th, 14th and 16th epochs. The

Method	Training Data	Accuracy
FaceNet [26]	200M	**99.65**
Deepid2+ [29]	0.3M	98.70
Softmax	0.49M	98.56 ± 0.03
L-Softmax [22]	0.49M	98.92 ± 0.03
Center [32]	0.49M	99.05 ± 0.02
LGM ($\alpha = 0.001$)	0.49M	99.03 ± 0.03
LGM ($\alpha = 0.005$)	0.49M	99.08 ± 0.02
LGM ($\alpha = 0.01$)	0.49M	$\textbf{99.20} \pm \textbf{0.03}$

Table 5. Face verification performances on LFW of a single model. The 6 models at bottom are trained on our scheme while the 2 results on top are reported from the original paper.

9122

networks are trained using stochastic gradient descent (SGD) with a momentum of 0.9 and a weight decay of 5×10^{-4}.

For the L-GM loss, we perform PCA on the 512-dimensional feature embeddings and then compute the Mahalanobis distance for verification. For fair comparison, the verification performance is evaluated on single models and model ensemble is not used. In Table 5, the accuracies for the Deepid2+ (contrastive loss) [29] and the FaceNet (triplet loss) [26] are reported in the original papers. The FaceNet achieves the highest accuracy of 99.65% by using a very large training set of 200M images. In [32], Y. Wen *et al.* reported a higher accuracy of 99.28% for the center loss by using both the CASIA-Webface and the Celebrity+ [23] dataset for training, with 0.7M training images in total. When using the CASIA-Webface training dataset only, the L-GM loss outperforms the other loss functions.

4.3. Beyond Classification

As we have discussed in Sect. 3.4, the proposed L-GM loss enables the likelihood estimation for a given input in addition to the class prediction. During training, the L-GM loss drives the deep model to generate features that follow the assumed GM distribution as well as possible, while guaranteeing the inter class separability. In other words, the training feature distribution is supposed to be well established for a trained deep model using the L-GM loss. We will validate this claim through experiments on distinguishing adversarial examples from normal inputs in this section.

Adversarial Examples For a deep neural network, adversarial examples are inputs formed by intentionally adding small but worst-case perturbations which cause the model to make incorrect classifications with high confidence [8]. We generate the adversarial examples using the fast gradient sigh method (FGSM) [8], which uses gradient backpropagation to perturb the inputs so as to maximize the classification loss. The perturbation P is generated by $P = \epsilon \cdot sign(\nabla_I \mathcal{L}(I, z))$, in which $\mathcal{L}$ is the classification loss function (*e.g.* $\mathcal{L}_{cls}$ in L-GM loss), I is the input image, z is the true class label, and $\epsilon > 0$ is called the magnitude of perturbation. Then the adversarial example is formed by adding P to the original image I. An extension of FGSM called the Targeted FGSM aims at misclassifying an input sample to a target class by minimizing the loss for the pre-set target label $\tilde{z}$. The targeted perturbation P_t is given by $P_t = \epsilon \cdot sign(-\nabla_I \mathcal{L}(I, \tilde{z}))$, in which $\mathcal{L}$ is $\mathcal{L}_{GM}$ in our experiments.

By using the FGSM, we first generate one adversarial example for each MNIST test image in order to evaluate the classification performance using different loss functions. As such there are altogether 10,000 adversarial examples and 10,000 original normal MNIST test images in the experiment. We use the CNN architecture as described in Sect. 4.1 with the 100-dimensional feature embedding. Three models

ϵ	Softmax	Center	L-GM($\alpha = 1$)
0	0.68	0.47	0.39
0.1	24.08	43.13	23.63
0.2	75.56	67.17	64.40
0.3	84.87	85.49	81.62

Table 6. Classification error rates (%) on adversarial examples generated from the MNIST test set using FGSM. $\epsilon = 0$ means that the inputs are normal MNIST test images.
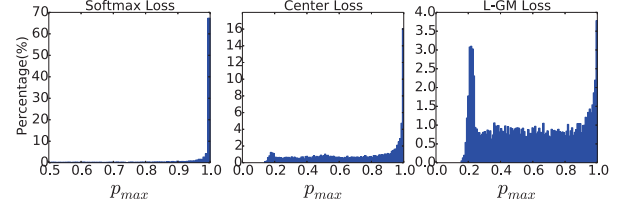

Figure 3. Histograms of the predicted posterior probability of the adversarial examples.

are trained on the standard MNIST training set by using the softmax loss, the center loss and the proposed L-GM loss respectively. The classification error rates on the adversarial examples are presented in Table 6, which shows that all three models seem to be vulnerable to adversarial attacks. We then investigate the posterior probability ($p_{max} = \max_k p(k|x)$) corresponding to the predicted class for both the normal inputs and the adversarial examples ($\epsilon = 0.3$). For adversarial examples, the histograms of p_{max} are shown in Fig. 3. For normal inputs, it is unnecessary to plot the histograms since the $p_{max} > 0.98$ for over 95% of the samples for all the three losses. Obviously, for the L-GM loss, the overlap between the histograms of p_{max} of the normal inputs and the adversarial examples is the smallest among three loss functions. This means that even by only considering the posterior probability in classification, the L-GM loss already outperforms the other two loss functions in distinguishing adversarial examples. Nevertheless, a more effective way for distinguishing adversarial examples is to directly consider the likelihood to the learned training feature distribution. We therefore design the following experiment.

Adversarial Verification Intuitively, in the feature space, the adversarial examples should follow a distribution different from that of the normal inputs. Based on this understanding, we design an experiment called the *adversarial verification* to distinguish the adversarial examples from normal inputs based on the feature likelihood. Let the predicted class be $\hat{z}_i = \arg\max_k p(k|x_i)$. For the L-GM loss, we now assume identity covariance matrix and equal priors for simplicity. Then the likelihood of x_i is $l_{GM,i} = exp(-\|x_i - \mu_{\hat{z}_i}\|^2/2)$ based on Eq. 8 by omitting the constant coefficient. And it can also be rewritten as $l_{GM,i} = exp(-\mathcal{L}_{lkd,i})$ according to Eq.10. For the center
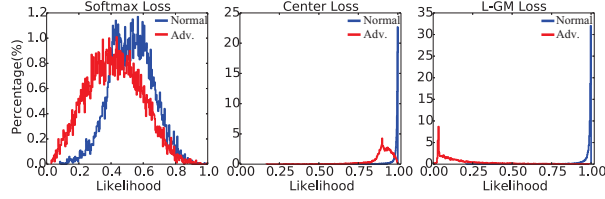
Figure 4. Histograms of the likelihood for adversarial examples (Adv.) and normal inputs (Normal).

loss, the likelihood can be computed similarly according to Lemma 1, leading to $l_{C,i} = exp(-\|x_i - \mu_{\hat{z}_i}\|^2/2)$. For the softmax loss, the likelihood is not explicitly established in its formulation. A reasonable way is to estimate the likelihood as $l_{S,i} = w_{\hat{z}_i}^T x_i + b_{\hat{z}_i}$. After all, the affinity score $w_{\hat{z}_i}^T x_i + b_{\hat{z}_i}$ represents the similarity between x_i and class $\hat{z}_i$.

In the adversarial verification experiment, the FGSM is used to generate adversarial examples for the MNIST test set, with $\epsilon = 0.3$. Then for the three models, we compute the likelihood of the normal test images and the adversarial examples. For the softmax loss, we normalize the likelihood l_S to $(0, 1]$ for comparison. The histograms of the likelihood for three loss functions are illustrated in Fig. 4. For the L-GM loss, the adversarial examples have very low likelihood in the feature space and the normal inputs can be easily distinguished from them. The softmax loss, however, clearly suffers from a serious overlap between the two likelihood histograms. The center loss lies in between by being superior to the softmax loss while inferior to the L-GM loss in terms of the capability of adversarial verification.

Quantitatively, we evaluate the adversarial verification performances by thresholding the likelihood, and resultant ROC curves are demonstrated in Fig. 5. The equal error rate (EER) for the softmax loss is 37.7%, which is practically too high in a binary classification task. The center loss performs much better with an EER of 10.2%. The proposed L-GM loss achieves the lowest EER of 3.1%. This experiment demonstrates that comparing to the other two loss functions, the L-GM loss can be effectively used for distinguishing adversarial examples. This validates our claim that the L-GM loss can well establish the training feature distribution while maintaining a satisfactory classification performance.

Discussions Theoretically speaking, it is possible to generate adversarial examples with high likelihood in the L-GM loss by jointly optimizing the classification loss and the likelihood regularization term. It can be verified that under the L-GM loss formulation, such a joint optimization can be approximately realized using the Targeted FGSM, in which the targeted perturbation P_t actually helps to reduce the distance between the feature and the center of the targeted class, or increase the likelihood. We test this approach by using the class with the second largest posterior probability as the
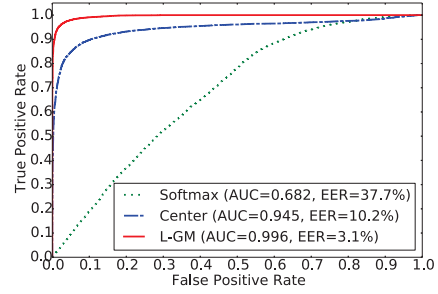

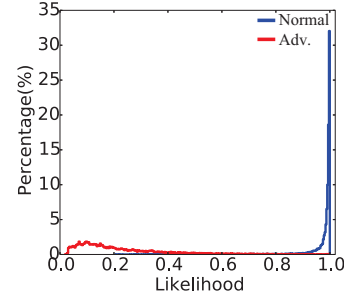Figure 5. ROC curves of the adversarial verification.

Figure 6. Histogram of the likelihood for adversarial examples generated by the Targeted FGSM against the L-GM loss.

target label $\tilde{z}$ for a given input. We still set $\epsilon = 0.3$ and only test the L-GM loss. The classification error rate is 81.37%, which is similar to that in Table. 6. The likelihood histogram is illustrated in Fig. 6. Compared to Fig. 4, the number of adversarial examples with very low likelihood (*e.g.* smaller than 0.2) is decreased, leading to a slightly higher EER of 4.3%. Nevertheless, most of the adversarial examples can still be distinguished using the likelihood.

5. Conclusions

We proposed a loss function by assuming a Gaussian Mixture (GM) distribution of the deep features on the training set. Besides the classification loss, a log likelihood regularization term is added to explicitly drive the deep model for generating GM distributed features. To further improve the generalization capability of the trained model, a classification margin is introduced. Extensive experiments demonstrate that the proposed L-GM loss outperforms the softmax loss and its variants in in both small and large-scale datasets when combined with different deep models. Besides, the L-GM loss facilitates a more effective distinguishment of abnormal inputs of which the extracted features follow a distribution different from the one learned during training. This can be practically useful, for example, to improve an deep model's robustness towards adversarial examples.

References

[1] C. M. Bishop. *Pattern recognition and machine learning*. Springer, 2006. 1

[2] B. Chen, W. Deng, and J. Du. Noisy softmax: Improving the generalization ability of dcnn via postponing the early softmax saturation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 2

[3] J. Deng, N. Ding, Y. Jia, A. Frome, K. Murphy, S. Bengio, Y. Li, H. Neven, and H. Adam. Large-scale object classification using label relation graphs. In *European Conference on Computer Vision*, pages 48–64, 2014. 1

[4] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009. 6

[5] D. G. E., Y. Dong, D. Li, and A. Alex. Context-dependent pretrained deep neural networks for large-vocabulary speech recognition. *IEEE Transactions on Audio Speech and Language Processing*, 20(1):30–42, 2011. 1

[6] H. Geoffrey, D. Li, Y. Dong, D. G. E., M. Abdelrahman, J. Navdeep, S. Andrew, V. Vincent, N. Patrick, and S. T. N. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97, 2012. 1

[7] H. Geoffrey, V. Oriol, and D. Jeff. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 2

[8] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014. 2, 3, 4, 7

[9] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *IEEE international Conference on Computer Vision*, pages 1026–1034, 2015. 6

[10] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016. 1, 5, 6

[11] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012. 6

[12] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger. Densely connected convolutional networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2261–2269. 1

[13] 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. 6

[14] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pages 448–456, 2015. 1

[15] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015. 6

[16] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009. 5

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Annual Conference on Neural Information Processing Systems*, pages 1106–1114, 2012. 1

[18] A. Kurakin, I. Goodfellow, and S. Bengio. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*, 2016. 3

[19] G. Larsson, M. Maire, and G. Shakhnarovich. Fractalnet: Ultra-deep neural networks without residuals. *arXiv preprint arXiv:1605.07648*, 2016. 1

[20] Y. LECUN, L. BOTTOU, Y. BENGIO, and P. HAFFNER. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 5

[21] M. Lin, Q. Chen, and S. Yan. Network in network. *arXiv preprint arXiv:1312.4400*, 2013. 6

[22] W. Liu, Y. Wen, Z. Yu, and M. Yang. Large-margin softmax loss for convolutional neural networks. In *International Conference on Machine Learning*, pages 507–516, 2016. 1, 2, 4, 5, 6

[23] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. pages 3730–3738, 2014. 7

[24] J. H. Metzen, T. Genewein, V. Fischer, and B. Bischoff. On detecting adversarial perturbations. *arXiv preprint arXiv:1702.04267*, 2017. 3

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

[26] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 815–823, 2015. 6, 7

[27] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 6

[28] S. Sun, W. Chen, L. Wang, X. Liu, and T. Liu. On the depth of deep neural networks: A theoretical view. In *AAAI Conference on Artificial Intelligence*, pages 2066–2072, 2016. 4

[29] Y. Sun, X. Wang, and X. Tang. Deeply learned face representations are sparse, selective, and robust. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2892–2900, 2015. 6, 7

[30] I. Sutskever, J. Martens, G. Dahl, and G. Hinton. On the importance of initialization and momentum in deep learning. In *International Conference on Machine Learning*, pages 1139–1147, 2013. 6

[31] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. *arXiv preprint arXiv:1512.00567*, 2015. 1, 2

[32] Y. Wen, K. Zhang, Z. Li, and Y. Qiao. A discriminative feature learning approach for deep face recognition. In *European Conferenc on Computer Vision*, pages 499–515, 2016. 1, 2, 3, 4, 5, 6, 7

[33] J. Yangqing, S. Evan, D. Jeff, K. Sergey, L. Jonathan, G. Ross, G. Sergio, and D. Trevor. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014. 5

[34] D. Yi, Z. Lei, S. Liao, and S. Z. Li. Learning face representation from scratch. *arXiv preprint arXiv:1411.7923*, 2014. 6

[35] S. Zagoruyko and N. Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016. 1

[36] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals. Understanding deep learning requires rethinking generalization. In *International Conference on Learning Representations*, 2016. 1, 2

[37] 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, 2016. 6