RBF-Softmax: Learning Deep Representative Prototypes with Radial Basis Function Softmax

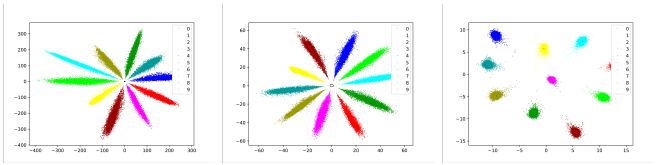
Xiao Zhang[1], Rui Zhao[2], Yu Qiao[3], and Hongsheng Li[1]

[1] CUHK-SenseTime Joint Lab, The Chinese University of Hong Kong
[2] SenseTime Research
[3] ShenZhen Key Lab of Computer Vision and Pattern Recognition, SIAT-SenseTime Joint Lab, Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences
zhangx9411@gmail.com, zhaorui@sensetime.com, yu.qiao@siat.ac.cn, hsli@ee.cuhk.edu.hk

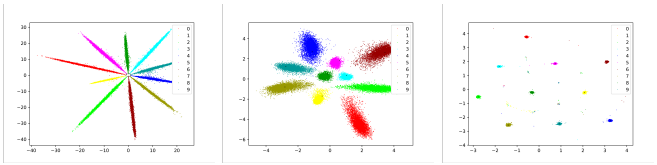
Abstract. Deep neural networks have achieved remarkable successes in learning feature representations for visual classification. However, deep features learned by the softmax cross-entropy loss generally show excessive intra-class variations. We argue that, because the traditional softmax losses aim to optimize only the relative differences between intra-class and inter-class distances (**logits**), it cannot obtain representative class prototypes (class weights/centers) to regularize intra-class distances, even when the training is converged. Previous efforts mitigate this problem by introducing auxiliary regularization losses. But these modified losses mainly focus on optimizing intra-class compactness, while ignoring keeping reasonable relations between different class prototypes. These lead to weak models and eventually limit their performance. To address this problem, this paper introduces a novel Radial Basis Function (RBF) distances to replace the commonly used inner products in the softmax loss function, such that it can adaptively assign losses to regularize the intra-class and inter-class distances by reshaping the relative differences, and thus creating more representative prototypes of classes to improve optimization. The proposed RBF-Softmax loss function not only effectively reduces intra-class distances, stabilizes the training behavior, and reserves ideal relations between prototypes, but also significantly improves the testing performance. Experiments on visual recognition benchmarks including MNIST, CIFAR-10/100, and ImageNet demonstrate that the proposed RBF-Softmax achieves better results than cross-entropy and other state-of-the-art classification losses. The code is at https://github.com/2han9x1a0release/RBF-Softmax.

1 Introduction

Recent years witnessed the breakthrough of deep Convolutional Neural Networks (CNNs) on various visual recognition tasks [15, 11, 21, 27, 13]. State-of-the-art deep learning based classification methods benefit from the following three factors: large-scale training datasets [23, 7], powerful network architectures [24, 9, 26, 11, 17], and effective training loss functions [33, 4, 29, 36, 20], which make deep neural networks the dominant model for visual classification. The cross-entropy



(a) Softmax (98.74%) (b) Prototype [1] (98.93%) (c) Center Loss [33] (99.05%)



(d) cos-softmax [32] (99.04%) (e) LGM Loss [29] (98.97%) (f) **RBF-Softmax** (99.20%)

Fig. 1: MNIST 2-D feature visualization of various losses. Intra-class feature distributions of sub-figure 1(a), 1(b) and 1(d) exhibit that conventional classification losses without additional regularization terms suffer from large intra-class sample-prototype distances (logits). Except proposed RBF-Softmax, class prototypes of all other losses and centers of their corresponding features have certain biases more or less.

softmax loss function and some of its variants [29, 4, 36] have been widely adopted for tackling the classification problem and enhancing the discriminativeness of the learned representations.

In deep classification tasks, the input image is firstly transformed into high-dimensional feature vector by Convolutional Neural Networks. To determine its class label, the similarities or distances between the feature vector and class prototypes (also called class weight vectors, class centers or class representations), namely sample-prototype distances, are calculated to get the logits. Conventionally, metrics including inner product, cosine [18] and Euclidean distance [29, 36] were exploited to produce logits.

In most existing methods, the logits of an input sample are normalized across all classes by a softmax function to generate the class probabilities. Besides the softmax function, other choices include methods like RBF Network [2], Bayesian formula and Gaussian distribution [29]. During the training process, classification probabilities of each input sample are optimized towards its ground-truth by the cross-entropy loss.

Euclidean distance is often used as the similarity metric for feature vectors because it is easy to calculate, and has clear geometric meaning. The learned image feature distributions are often expected to have small intra-class distances and sufficiently large inter-class distances. However, the existing softmax loss and its variants do not directly optimize the Euclidean distances, but rather the relative differences between the intra-class logits and inter-class logits. Specifi-

cally, when the distance between a sample to its corresponding class prototype is relatively smaller than its distances to other class prototypes, the penalty from the softmax loss will be small, but the distance to its corresponding class prototype might still be large. Therefore, contrastive loss [3] and triplet loss [10] were proposed to directly optimize the Euclidean distance and yielded better performance in practice. But such losses are subject to difficulties in mining effective sample pairs and training convergence, and thus cannot completely replace the traditional softmax losses. Subsequently, DeepID2 [25], center loss [33] and range loss [37] jointly utilized the metric loss and traditional softmax loss for supervision, and achieved great success in face recognition. Along this direction, CPL [36] and LGM [29] algorithms added intra-class distance regularization terms into the softmax function to regularize the feature distributions. However, such terms still face the challenge of extra unstable regularization losses. Fig. 1 shows MNIST 2-D feature distributions of some of these losses. Moreover, in section 4.1 we will fully exhibit that most of these variants improve the discriminativeness of class prototypes rather than their semantic representativeness, by which we mean the reasonable relations between class prototypes.

In this paper, we propose a Radial Basis Function softmax (RBF-Softmax) loss function for visual classification. The key idea is to reshape the Euclidean distances between the sample features and class prototypes with RBF kernel, before feeding them into the softmax function for normalization. RBF-Softmax loss function can more effectively minimize intra-class Euclidean distances, and increase expansion of multi-class distribution with keeping reasonable class relations simultaneously.

On the one hand, for optimizing the intra-class Euclidean distances, the proposed RBF kernel can provide more balanced supervisions in the early training stages and distance-sensitive supervisions in the later training stages. In the early training stage, all features of the same class are likely to be scattered sparsely in the feature space due to the random initialization, leading to large intra-class variations with large logits. With the RBF kernel, the samples in the same class would have similar penalties no matter whether they have different Euclidean distances to the class prototypes, leading to stable convergence behavior. When the training is close to convergence, existing loss functions tend to provide very few supervision due to the relatively large inter-class distances. For features belonging to the same class, they still have the potential to be closer to the class prototype. The sample-to-prototype similarities by the RBF kernel have greater change rates than their original Euclidean distances or inner products to the class prototype, which are able to provide sufficient supervisions even close to convergence.

On the other hand, our proposed RBF kernel logits can effectively reshape and bound the logits, and then results in more balanced ratios between intra-class and inter-class logits. And the resulting class prototypes are more representative and, in turn, lead to better classification performance.

Extensive experiments on validating the effectiveness of the proposed RBF-Softmax loss has been tested on multiple visual recognition benchmarks, including MNIST [16], CIFAR-10/CIFAR-100 [14] and ImageNet [23]. Experiments show that the RBF-Softmax outperforms state-of-the-art loss functions on visual classification on all the tested benchmarks.

The contributions of this paper could be summarized in to three-fold: (1) We argue that the main defect caused by biased loss allocations of conventional softmax loss can lead to weak models and imperfect class prototypes; (2) We therefore proposed an effective RBF-Softmax loss to address aforementioned defect by using RBF kernel to control loss allocations; (3) We proved that RBF-Softmax can generate ideal class prototypes as well as improve classification performance through extensive experiments.

2 Related Works

Classification losses. Visual classification is the fundamental problems in computer vision and the advances of visual classification also promote related research directions One of its major components is how to design effective classification loss functions. The designs of classification losses are usually dependent on the classification criterion during inference. In face recognition, the testing phase requires to calculate the cosine similarity between face images. Based on this demand, a series of cosine based softmax losses [6, 31] and their margin-based variants [32, 30, 4] were proposed and achieved great success. In prototype learning, samples need to be abstracted into feature vectors with one or more centers in a high-dimensional space. [1] and CPL [36] directly adopt the Euclidean distance as the classification score of prototype metrics.

Euclidean distance based losses. Metric learning has been an important research area of machine learning and deep learning. Commonly used Euclidean distance based losses include contrastive loss [3] and triplet loss [10]. Specifically, Euclidean losses take distances among samples as optimization objectives and strive to reduce distances between samples within the same classes while enlarge distances between samples across different classes. However, such a design can cause difficulties in mining efficient sample pairs or triplets. [34] showed that the different sampling methods have significants impact on networks' training behavior as well as the final performances. Therefore, Euclidean distance based losses are often used for fine-tuning rather than training from scratch.

Regularization for classification based losses. The joint supervision of classification loss and Euclidean distance based loss was adopted to train deep neural networks. Such combinations of loss terms results in more stable training behaviors. The success of Center Loss [33], Range Loss [37], Ring Loss [38], CPL [36], g-Softmax [20] and LGM [29] in face recognition and visual classification have proven that such joint supervision is a better trade-off in training deep models.

3 Radial Basis Function Softmax Loss

In this section, we will first recall the significance of class prototypes and analyze two problems in the conventional softmax cross-entorpy loss and its variants [1] (Sec. 3.1). Then we will introduce the proposed RBF-Softmax in details in Sec. 3.2. How RBF-Softmax solve defects faced by traditional softmax loss is further explained and discussed to demonstrate its effectiveness in visual classification (Sec. 3.3).

3.1 Analysis of the Softmax Cross-entropy Losses and Prototypes

Considering a classification task with C-class where the traditional softmax loss is used. $\boldsymbol{x}_i$ is the feature vector of one specific sample belonging to class $y_i \in [1, C]$, its softmax cross entropy loss is calculated as

$$\mathcal{L}_{\text{Softmax}}(\boldsymbol{x}_i) = -\log P_{i,y_i} = -\log \frac{e^{f_{i,y_i}}}{\sum_{k=1}^{C} e^{f_{i,k}}}, \tag{1}$$

where P_{i,y_i} is the probability that $\boldsymbol{x}_i$ being assigned to its ground-truth class y_i, and the logit $f_{i,j}$ represents the affinity between the sample feature $\boldsymbol{x}_i$ and class prototypes $\boldsymbol{W}_j$. Particularly, when $j = y_i$, logit f_{i,y_i} is the affinity between sample feature $\boldsymbol{x}_i$ to its corresponding class prototype $\boldsymbol{W}_{y_i}$, which is called the *intra-class sample-prototype distance* or *intra-class logit* in this paper. Conversely, when $j \neq y_i$, logit $f_{i,j}$ is named as the *inter-class sample-prototype distance* or *inter-class logit*. To measure the similarity between a sample feature and class prototypes, inner product and Euclidean distance were widely used, for example $f_{i,j} = \boldsymbol{W}_j{}^{\mathrm{T}}\boldsymbol{x}_i$ in Softmax loss and $f_{i,j} = -\alpha\|\boldsymbol{x}_i - \boldsymbol{W}_j\|_2^2$ in prototype learning [1] and CPL [36].

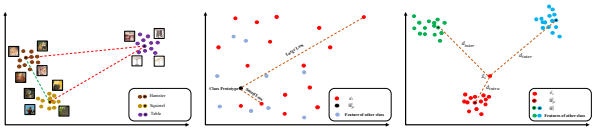
In these losses, a prototype can be seen as the representation of all sample features in a specified class. Intuitively, an ideal prototype should be the geometric center of all corresponding feature vectors. Therefore, prototype is required to have significant representativeness, which includes two aspects:

1. Prototypes should effectively discriminate and categorize samples from different classes. The inter-class distances are larger than intra-class distances;
2. Prototypes should demonstrate the relations among classes, which means similar classes is more closer than absolutely different classes.

These aspects can be demonstrate in Fig. 2(a). In this figure, there are three different classes: hamsters, squirrels, and tables. Hamsters and squirrels are similar, while both of them are very different from tables. Therefore, ideal sampels and prototypes distribution should ensure that every class is separable with other classes, but keep some similar classe prototypes closer.

During training, network parameters are gradually optimized to minimize the loss functions. The final feature distributions highly rely on prototypes as well as the losses used. The above mentioned existed logit calculations may lead to two defects in properly supervising the feature learning.

Biased loss allocation at the beginning of training. The class prototype vector $\boldsymbol{W}_j$ can be considered to be the mean representation of all samples in class j. Since the network in early training stages are not fully optimized, $\boldsymbol{W}_j$ as well as $\boldsymbol{x}_i$ tend to be somewhat random and do not have valid semantic information. Therefore, distances between features $\boldsymbol{x}_i$ and their corresponding class prototypes $\boldsymbol{W}_{y_i}$ cannot correctly represent their similarities. This fact indicates that samples should receive constrained training losses to avoid the negative impact of outliter (see illustration in Fig. 2(b)). Tabel 1 shows the intra-class sample-class distances in early-stage intra-class have large variances, which result in significant differences in loss for intra-class samples.



(a) (b) (c)

Fig. 2: Fig (a) is sampels and prototypes demonstration. Black spots represent prototypes of classes and solid color spots represent sample features. Sample features and prototypes of similar classes (hamster and squirrel) are separable, but have much closer distances than absolutely different class (table). Fig 2(b) is feature distribution at early training stage. Since features have not been well embedded at this stage, the loss value of each sample should be relatively similar. However, there might be large variances in different samples' loss values. Fig 2(c) is feature distribution diagram of late stage. The intra-class sample-prototype distance d_{intra} of the annotated $\boldsymbol{x}_i$ is relatively larger than other samples in its class y_i. Therefore its expected loss value should also be large. However, since d_{inter}s are much larger than d_{intra}, resulting in a rather small loss, so $\boldsymbol{x}_i$ can not be further optimized.

Eventually, such biased loss allocation may hinder the models training behavior and cause significant bias between class prototypes and real feature distribution centers.

Table 1: Intra-class sample-prototype distance at early training stage. For different feature dimensions, the range as well as the variances of intra-class sample-prototype distances are very large at the early training stage on the MNIST dataset with a 6-layer convolutional network.

Feat. Dim.	Early Intra-class sample-prototype distance			
	Min.	Max.	Avg.	Variance
2	0.01	59.52	6.59	146.65
32	8.08	206.17	54.27	1186.8
128	15.72	271.18	69.97	2284.84

Large intra-class sample-prototype distance at late training stage. During late training stage, softmax loss also leads to problematic phenomenons. As shown in Fig. 2(c), when a sample x_i's inter-class sample-prototype distances (For example its distances to other class prototypes, $f_{i,j}$ for $j \neq y_i$) are significantly larger than its intra-class logit f_{i,y_i}, this sample will receive small loss value and thus small gradients during optimization even when the intra-class logit f_{i,y_i} is large. Compared to other samples in class y_i, feature $\boldsymbol{x}_i$ needs a larger loss in order to get close to its corresponding class prototype $\boldsymbol{W}_{y_i}$. How-

ever, since the softmax loss focuses on optimizing relative differences between intra-class and inter-class logits and cannot generate enough penalty for this case.

To further illustrate this issue, we analyze from the perspective of the sample gradient. According to Eq. (1), the gradient w.r.t. feature vector $\boldsymbol{x}_i$ is

$$\frac{\partial \mathcal{L}(\boldsymbol{x}_i)}{\partial \boldsymbol{x}_i} = \sum_{j=1}^{C} (P_{i,j} - \mathbb{1}(y_i = j)) \cdot \frac{\partial f_{i,j}}{\partial \boldsymbol{x}_i}, \tag{2}$$

where $\mathbb{1}$ is the indicator function and $f_{i,j}$ is the logit between $\boldsymbol{x}_i$ and $\boldsymbol{W}_j$. The classification probability $P_{i,j}$ is calculated by the softmax function. When $j = y_i$, if the relative difference between inter-class and intra-class logit of $\boldsymbol{x}_i$ is large enough, P_{i,y_i} will be very close to 1 and then the gradient of $\boldsymbol{x}_i$ will be small. At this time, the intra-class logit may still be large.

According to the above analysis, existing softmax cross-entropy loss has the problems of biased loss allocation at early training stages and large intra-class sample-prototype distance at late training stages. Therefore, we argue that solving these two defects by designing a new loss function can effectively optimize the model training.

3.2 RBF-Softmax loss function

To fix the above mentioned defects in existing softmax loss functions, we propose a distance named Radial Basis Function kernel distance (RBF-score) between $\boldsymbol{x}_i$ and $\boldsymbol{W}_j$ to measure the similarities between a sample feature $\boldsymbol{x}_i$ and different classes' weights $\boldsymbol{W}_j$,

$$K_{i,j} = K_{\mathrm{RBF}}(\boldsymbol{x}_i, \boldsymbol{W}_j) = e^{-\frac{d_{i,j}}{\gamma}} = e^{-\frac{\|\boldsymbol{x}_i - \boldsymbol{W}_j\|_2^2}{\gamma}}, \tag{3}$$

where $d_{i,j}$ is the Euclidean distance between $\boldsymbol{x}_i$ and $\boldsymbol{W}_j$, and γ is a hyperparameter. Compared to the Euclidean distance and inner product that are unbounded, RBF-score decreases as the Euclidean distance increases and its values range from 0 (when $d_{i,j} \to \infty$) to 1 (when $\boldsymbol{x}_i = \boldsymbol{W}_j$). Intuitively, RBF-score well measures the similarities between $\boldsymbol{x}_i$ and $\boldsymbol{W}_j$ and can be used as the logits in the softmax cross-entropy loss function.

Formally, we define the Radial Basis Function Softmax loss (RBF-Softmax) as

$$\begin{aligned} \mathcal{L}(\boldsymbol{x}_i)_{\text{RBF-Softmax}} &= -\log P_{i,y_i} = -\log \frac{e^{s \cdot K_{\mathrm{RBF}}(\boldsymbol{x}_i, \boldsymbol{W}_{y_i})}}{\sum_{k=1}^{C} e^{s \cdot K_{\mathrm{RBF}}(\boldsymbol{x}_i, \boldsymbol{W}_k)}} \\ &= -\log \frac{e^{s \cdot e^{-\frac{d_{i,y_i}}{\gamma}}}}{\sum_{k=1}^{C} e^{s \cdot e^{-\frac{d_{i,k}}{\gamma}}}}, \end{aligned} \tag{4}$$

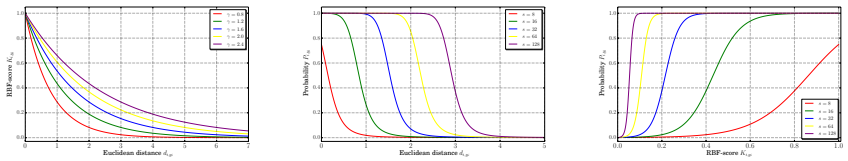
where $d_{i,j} = \|\boldsymbol{x}_i - \boldsymbol{W}_j\|_2^2, j \in \{1, \cdots, C\}$ and the hyperparameter s is a scale parameter in order to enlarge the range of RBF-scores. Similar hyperparameter has been extensively discussed in some cosine-based softmax losses [18, 19, 31, 31, 32, 4], in order to enlarge the range of RBF-scores.

3.3 Analysis of RBF-Softmax

In this subsection, we analyze two aspects of our proposed RBF-Softmax loss function: (1) the mechanism how the RBF-Softmax overcome two defects mentioned above; (2) the effects of two hyperparameters in RBF-Softmax.

Overcome the inappropriate penalties. RBF-Softmax essentially solves the above mentioned problems by adopting the original inner products or Euclidean distances as logits in a more reasonable way. On the one hand, it is important to balance each sample's intra-class logits at the early training stage. The initial values of intra-class logits are generally all very large. By adopting the RBF-score, the RBF kernel can map the very large Euclidean distances to very small RBF-scores as logits, thereby significantly reducing the intra-class variance. Then, due to the small variances of the intra-class RBF-score at early training stage, the loss allocation of samples belonging to the same classes is unbiased. On the other hand, at the late training stage, traditional softmax probabilities easily reach 1 on corresponding classes (gradients become 0), while RBF probabilities are much more difficult to reach 1 and can continually provide gradients for training.In this way, the proposed RBF-Softmax can better aggregate samples to their corresponding class centers, thereby improving the performance of model.

Effects of hyperparameters. Hyperparameters (γ and s) of the proposed RBF-Softmax affect the training of model to some extent. Let $K_{i,y_i} = e^{-\frac{d_{i,y_i}}{\gamma}}$ be the RBF-score between feature x_i and its corresponding class prototype W_{y_i}, $d_{i,y_i} = \|\boldsymbol{x}_i - \boldsymbol{W}_{y_i}\|_2^2$ is the Euclidean distance, and P_{i,y_i} is the probability of x_i being assigned to its corresponding class y_i.



(a) K_{i,y_i} w.r.t. d_{i,y_i} (b) P_{i,y_i} w.r.t. d_{i,y_i} (c) P_{i,y_i} w.r.t. K_{i,y_i}

Fig. 3: Fig 3(a) is curves of K_{i,y_i} w.r.t. d_{i,y_i} when choosing different γ parameters. Fig 3(b) and Fig 3(c) are curves of P_{i,y_i} w.r.t. d_{i,y_i} and K_{i,y_i} when choosing different scale s parameters.

Fig. 3(a) shows the mapping of d_{i,y_i} to K_{i,y_i} under different γ hyperparameters. When γ is larger, RBF-score K_{i,y_i} obtained by the specified d_{i,y_i} will be larger, and the similarity between the sample and their corresponding class prototypes would be higher, which means that the task becomes easier. Fig 3(b) and Fig 3(c) shows the mapping of d_{i,y_i} and K_{i,y_i} to P_{i,y_i} under different s hyperparameters. Our experiments show that when $j \neq y_i$, $d_{i,j}$ is generally much larger,

causing the value of $K_{i,j}$ to be close to 0. Moreover, s controls the range of p_{i,y_i}, and the difficulty of the classification task: for the fixed d_{i,y_i} or K_{i,y_i}, smaller s leads to narrower range and smaller value of P_{i,y_i}, making the classification task harder.

The same conclusion can be drawn from the perspective of the gradient. The corresponding gradients of RBF-Softmax are as follows:

$$\frac{\partial \mathcal{L}_{\text{RBF-Softmax}}}{\partial \boldsymbol{x}_i} = \sum_{j=1}^{C} (P_{i,j} - \mathbb{1}(y_i = j)) \cdot s \cdot K_{i,j} \frac{\partial d_{i,j}}{\partial \boldsymbol{x}_i}, \tag{5}$$

$$\frac{\partial \mathcal{L}_{\text{RBF-Softmax}}}{\partial \boldsymbol{W}_j} = (P_{i,j} - \mathbb{1}(y_i = j)) \cdot s \cdot K_{i,j} \frac{\partial d_{i,j}}{\partial \boldsymbol{W}_j}, \tag{6}$$

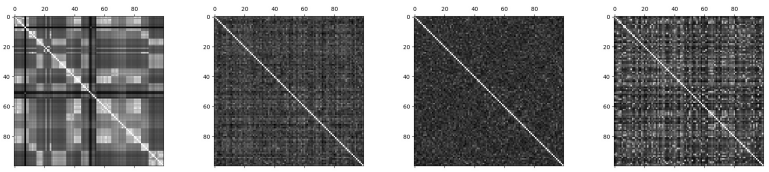
where $K_{i,j} = \mathrm{e}^{-\frac{d_{i,j}}{\gamma}}$ and $d_{i,j} = \|\boldsymbol{x}_i - \boldsymbol{W}_j\|_2^2$. In these gradients, RBF-scores are factors of gradients and determine their lengths. Therefore the change in hyperparameters can affect the norm of gradients and eventually the performance of models.

4 Experiments

In this section, we first exhibit several exploratory experiments on different prototypes, and then investigate the effectiveness and sensetiveness of different hyperparameters s and γ on the MNIST [16] dataset in Sec. 4.2. After that we evaluate the performances of proposed RBF-Softmax and compare with several state-of-the-art loss functions on CIFAR-10/100 [14] (in Sec. 4.3) and ImageNet [23] (in Sec. 4.4).

4.1 Exploratory Experiments on Prototypes

In order to analyze the prototypes of different softmax losses, here we use WordNet [5] and CIFAR-100 [14] as demonstrations. WordNet [5] is a widely used electronic lexical database, which can calculate the similarities between different English words from the perspective of computational linguistics. CIFAR-100 [14] dataset contains 100 classes which can be grouped into 20 superclasses, such as reptiles, flowers and etc. In each superclasses, there are 5 different but similar subclasses. Therefore, we can get a similarity matrix of all 100 classes in CIFAR-100 [14] by using WordNet [5] similarities. Fig. 4(a) exhibits such 100×100 WordNet [5] similarity matrix of CIFAR-100 [14], where the indexes of subclasses from the same superclass are continuous. Here we use WUP similarities [35] to measure the relations of classes. Paler block color means the two corresponding classes are more similar while darker color means two classes are more different. The WordNet [5] similarity matrix can be seen as the groundtruth. Then we trained ResNet-50 [9] models with conventional softmax loss and cosine based softmax loss [18] on CIFAR-100 [14], and computed their class prototype similarity matrices respectively. Fig. 4(b) and Fig. 4(c) imply that the relations among classes are not reserved by prototypes in these loss functions.



(a) WordNet (b) Softmax (c) Cos-Softmax [18] (d) **RBF-Softmax**

Fig. 4: Prototypes similarity matrices of WordNet [5] and different losses. The color of every block represents the degree of similarity between classes. Lighter block color means higher similarity.

To further explore the representativeness of class prototypes in these losses, we introduce two indicators: comparisons between similarity matrices and Calinski-Harabaz index of all 100 subclasses. By calculating the comparisons of similarity matrices, the differences between prototype similarity matrices and WordNet [5] similarity matrices can evaluate weather trained prototypes can reserve semantic information. Calinski-Harabaz (CH) index is a widely used validation of cluster algorithm. We expect subclasses in a same superclass are compact while different superclasses are separable. Tab.2 exhibits all result of these two indicators. We first measure the similarities between matrix in Fig. 4(a) and other matrices in Fig. (b), (c), and (d). Matrix of RBF-Softmax is more similar to WordNet [5] matrix. Moreover, prototypes in RBF-Softmax have significantly higher Calinski-Harabaz index. These results preliminarily tell that models trained with RBF-Softmax are more representative.

Table 2: Representativeness experiments on CIFAR-100 [14] prototypes. Similartities between WordNet matrix and others show whether prototypes keep reasonable class relations. CH indexes indicate whether classes under the same superclass is gathered.

Losses	Similarity of Simi. Mat.	Calinski-Harabaz Index
WordNet [5]	1.00	Not Appliable
Softmax	0.17 ± 0.05	2.81 ± 0.21
Cos-Softmax [18]	0.09 ± 0.04	1.68 ± 0.10
RBF-Softmax	0.36 ± 0.11	7.33 ± 0.25

4.2 Exploratory Experiments on MNIST

We first use MNIST [16] to investigate RBF-Softmax. All experiments of MNIST [16] are trained with a simple 6-layer CNN, where all convolutinal kernels are 5×5 and the activation function is PReLU [8].

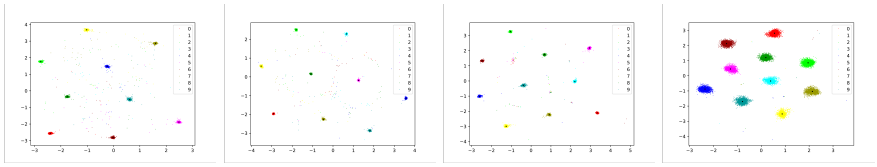
Table 3: Recognition accuracy (%) on MNIST with different s and γ hyperparameters and feature dimensions.

Hyperparameter		Feature Dimension		
		2-D	10-D	32-D
$\gamma = 1.0$	$s = 2.0$	99.20%	**99.68%**	**99.71%**
	$s = 4.0$	**99.29%**	99.65%	99.69%
	$s = 8.0$	99.25%	99.56%	99.61%
$s = 2.0$	$\gamma = 1.00$	**99.20%**	**99.68%**	**99.71%**
	$\gamma = 1.3$	99.15%	99.54%	99.65%
	$\gamma = 2.0$	99.03%	99.36%	99.42%

Tables 3 exhibit the impacts of hyperparameter s and γ on models performance respectively. Fig. 5 partly visualizes the feature distributions of different s parameters. According to these results, we find that fixing s to 2.0 and γ to 1.0 for model can outshine other configurations. Results in Table 4 compares the performances of state-of-the-art loss functions on MNIST [16]. The only difference of these models is their loss functions, where RBF-Softmax follows the mentioned setting. In the MNIST [16] dataset, our RBF-Softmax outperforms all other losses. According to Sec. 3.3, both γ and s can change the constraint of RBF-Softmax. These experiments shows that too strong or too weak constraint can lead to performance degradation and RBF-Softmax is not sensitive to hyperparameters selected within a reasonable range.

Table 4: Recognition accuracy (%) on MNIST with different compared losses. The are all trained with a 6-layer CNN and different losses for three times to obtain the average accuracies. The feature dimension is 32.

Method	1st	2nd	3rd	Avg. Acc.
Softmax	99.28%	99.27%	99.25%	99.27%
RBF Networks [2]	97.42%	97.07%	97.36%	97.28%
Center Loss [33]	99.66%	99.64%	99.64%	99.65%
Ring Loss [38]	99.56%	99.59%	99.58%	99.58%
ArcFace [4]	99.60%	99.55%	99.62%	99.59%
LGM [29]	99.41%	99.35%	99.40%	99.39%
RBF-Softmax	**99.70%**	**99.73%**	**99.75%**	**99.73%**



(a) $s = 1$ (99.18%) (b) $s = 2$ (99.20%) (c) $s = 4$ (99.29%) (d) $s = 8$ (99.25%)

Fig. 5: MNIST 2-D feature visualization when trained with different hyperparameters s.

4.3 Experiments on CIFAR-10/100

CIFAR-10 and CIFAR-100 [14] each contains $50,000$ training images and $10,000$ testing images, which are 32×32 color images. For the data augmentation scheme, horizon flipping (mirroring) and 32×32 random cropping after 4-pixel zero-padding on each side are adopted to all the training procedures [9].

Table 5: Recognition accuracy rates (%) on CIFAR-10 using ResNet-20 [9] and DenseNet-BC ($k = 12$) [12] models with different loss functions.

Loss Functions	Accuracy on CIFAR-10		Settings
	ResNet-20 [9]	DenseNet-BC ($k = 12$) [12]	
Softmax	91.25%	95.49%	[9, 12]
G-CPL [36]	91.63%	–	[36]
Center Loss [33]	91.85%	95.77%	[33]
RBF-Softmax	92.26%	95.83%	γ=2, s=3
	92.42%	95.95%	γ=2, s=4
	92.61%	**96.13%**	γ=1.8, s=4
	92.77%	96.11%	γ=1.6, s=4

For CIFAR-10 [14], we train the ResNet-20 [9] and DenseNet-BC ($k = 12$) [12] with different loss functions. All ResNet-20 [9] models are trained with a batch size of 256 for 300 epochs. The initial learning rate is 0.1 and is then divided by 2 every 60 epochs. In DenseNet-BC ($k = 12$) [12] models, we use batch size 128 for 300 epochs, and the learning rate is set to 0.1 and then divided by 10 at the 150th epoch and the 225th epoch respectively. The recognition accuracy are exhibited in Table 5. For ResNet-20 [9] and DenseNet-BC ($k = 12$) [12], our RBF-Softmax achieves state-of-the-art 92.77% and 96.13% accuracy respectively.

For CIFAR-100 [14], we train VGGNet-19 [24] with different loss functions. All RBF-softmax trainings follow the same setting: models are trained with batch size 128 for 600 epochs; the initial learning rate is 0.1, and is divided by 2 at the

Table 6: Recognition accuracy rates (%) on CIFAR-100 using VGGNet-19 [24] models with different loss functions and hyperparameter settings.

Loss Functions	Accuracy on CIFAR-100 VGGNet-19 [24]	Settings
Softmax	72.23%	-
Center Loss [33]	73.02%	[33]
G-CPL [36]	72.88%	[36]
RBF-Softmax	72.72%±0.03%	γ=2.2, s=10
	73.98%±0.02%	γ=2.2, s=14
	72.62%±0.05%	γ=1.0, s=12
	71.77%±0.04%	γ=4.0, s=12

100th, 300th and 500th epoch, and by 5 at 200th, 400th and 600th epoch. The results of CIFAR-100 [14] are shown in Table 6 and RBF-Softmax again shows state-of-the-art performances on VGGNet-19 [24] architectures.

4.4 Experiments on ImageNet

We investigate the performance of proposed RBF-Softmax on large-scale visual classification task using the ImageNet [23] dataset (ILSVRC2012). In order to show that the proposed RBF-Softmax is effective on various network architectures, the performed experiments using both manually designed mdoels (like ResNet [9]) and automatically searched architectures (like EfficientNet [28]). In all ImageNet [23] experiments, models are combined with conventional softmax loss and our RBF-Softmax, respectively.

Table 7: Recognition accuracy (%) on ILSVRC2012 [23].

Networks	Methods	Single-crop Top-1 Acc.	Settings
ResNet-50 [9]	Softmax	76.8%	-
	RBF-Softmax	**77.1%**	$s = 8; \gamma = 4$
EfficientNet-B0 [28]	Softmax	75.1%	-
	RBF-Softmax	**75.3%**	$s = 35; \gamma = 16$
EfficientNet-B1 [28]	Softmax	75.9%	-
	RBF-Softmax	**76.6%**	$s = 35; \gamma = 16$
EfficientNet-B4 [28]	Softmax	78.8%	-
	RBF-Softmax	**79.0%**	$s = 35; \gamma = 16$

All models are trained on 8 NVIDIA GeForce GTX TITAN X GPUs on 1.28 million images and evaluated for both top-1 and top-5 accuracies on the 50k validation images. Most of the training processes follow settings in [22]. For

training ResNet [9], the input images are single-cropped to 224×224 pixels. For EfficientNet [28], the training image size varies following its original paper. We use simple training processes without any training enhancements, like DropOut, DropConnect, AutoAugment, and etc. We apply SGD with momentum of 0.9 as optimization method and generally train for 100 epochs. During training, we use cosine schedule with 5 epoch gradual warmup as learning rate policy. The initial learning rate of ResNet [9] is 0.1, and 0.2 for EfficientNet [28]. The batch size of most models is 256 except EfficientNet-B4 [28]. Because of the limitation of GPU memory, batch size of EfficientNet-B4 is 96. The results are expressed in Table 7, where RBF-Softmax beats conventional softmax loss for both manually designed models and automatically searched models.

5 Conclusions

In this paper, we identify biased loss allocation and large intra-class logits (scores) as two primary defects prevent some conventional softmax losses from achieving ideal class prototypes and accurate classification performances. To address this problem, we propose Radial Basis Function softmax loss (RBF-Softmax) which applies RBF-kernel logits to the softmax cross-entropy loss in order to reasonably allocate losses and optimize intra-class distributions. Our RBF-Softmax is simple but highly effective and insightful. We demonstrate its effectiveness by demonstrating prototype experiments and appling it in close-set image classification tasks (MNIST [16], CIFAR-10/100 [14], and ImageNet [23]). Results shows that RBF-Softmax achieves state-of-the-art performances on all the evaluated benchmarks.

Acknowledgements

This work is supported in part by SenseTime Group Limited, in part by the General Research Fund through the Research Grants Council of Hong Kong under Grants CUHK 14202217 / 14203118 / 14205615 / 14207814 / 14213616 / 14208417 / 14239816, in part by CUHK Direct Grant and in part by the Joint Lab of CAS-HK.

References

1. Bonilla, E., Robles-Kelly, A.: Discriminative probabilistic prototype learning. arXiv preprint arXiv:1206.4686 (2012)
2. Broomhead, D.S., Lowe, D.: Radial basis functions, multi-variable functional interpolation and adaptive networks. Tech. rep., Royal Signals and Radar Establishment Malvern (United Kingdom) (1988)
3. Chopra, S., Hadsell, R., LeCun, Y.: Learning a similarity metric discriminatively, with application to face verification. In: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05). vol. 1, pp. 539–546. IEEE (2005)
4. Deng, J., Guo, J., Zafeiriou, S.: Arcface: Additive angular margin loss for deep face recognition. arXiv preprint arXiv:1801.07698 (2018)
5. Fellbaum, C., Miller, G.: WordNet:An Electronic Lexical Database (1998)

6. Gopal, S., Yang, Y.: Von mises-fisher clustering models. In: International Conference on Machine Learning. pp. 154–162 (2014)
7. Guo, Y., Zhang, L., Hu, Y., He, X., Gao, J.: Ms-celeb-1m: A dataset and benchmark for large-scale face recognition. In: European Conference on Computer Vision. pp. 87–102. Springer (2016)
8. He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In: Proceedings of the IEEE international conference on computer vision. pp. 1026–1034 (2015)
9. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
10. Hoffer, E., Ailon, N.: Deep metric learning using triplet network. In: International Workshop on Similarity-Based Pattern Recognition. pp. 84–92. Springer (2015)
11. Hu, J., Shen, L., Sun, G.: Squeeze-and-excitation networks. arXiv preprint arXiv:1709.01507 (2017)
12. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4700–4708 (2017)
13. Huang, G.B., Ramesh, M., Berg, T., Learned-Miller, E.: Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Tech. rep., Technical Report 07-49, University of Massachusetts, Amherst (2007)
14. Krizhevsky, A., Hinton, G.: Learning multiple layers of features from tiny images. Tech. rep., Citeseer (2009)
15. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems. pp. 1097–1105 (2012)
16. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P., et al.: Gradient-based learning applied to document recognition. Proceedings of the IEEE **86**(11), 2278–2324 (1998)
17. Liu, C., Zoph, B., Neumann, M., Shlens, J., Hua, W., Li, L.J., Fei-Fei, L., Yuille, A., Huang, J., Murphy, K.: Progressive neural architecture search. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 19–34 (2018)
18. Liu, Y., Li, H., Wang, X.: Learning deep features via congenerous cosine loss for person recognition. arXiv preprint arXiv:1702.06890 (2017)
19. Liu, Y., Li, H., Wang, X.: Rethinking feature discrimination and polymerization for large-scale recognition. arXiv preprint arXiv:1710.00870 (2017)
20. Luo, Y., Wong, Y., Kankanhalli, M., Zhao, Q.: g-softmax: Improving intraclass compactness and interclass separability of features. IEEE transactions on neural networks and learning systems **31**(2), 685–699 (2019)
21. Parkhi, O.M., Vedaldi, A., Zisserman, A., et al.: Deep face recognition. In: BMVC. vol. 1, p. 6 (2015)
22. Radosavovic, I., Kosaraju, R.P., Girshick, R., He, K., Dollár, P.: Designing network design spaces. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10428–10436 (2020)
23. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L.: ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision (IJCV) **115**(3), 211–252 (2015). https://doi.org/10.1007/s11263-015-0816-y
24. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition (2015)
25. Sun, Y., Chen, Y., Wang, X., Tang, X.: Deep learning face representation by joint identification-verification. In: Advances in neural information processing systems. pp. 1988–1996 (2014)

26. Szegedy, C., Ioffe, S., Vanhoucke, V., Alemi, A.A.: Inception-v4, inception-resnet and the impact of residual connections on learning. In: AAAI. vol. 4, p. 12 (2017)
27. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1–9 (2015)
28. Tan, M., Le, Q.V.: Efficientnet: Rethinking model scaling for convolutional neural networks. arXiv preprint arXiv:1905.11946 (2019)
29. Wan, W., Zhong, Y., Li, T., Chen, J.: Rethinking feature distribution for loss functions in image classification. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 9117–9126 (2018)
30. Wang, F., Liu, W., Liu, H., Cheng, J.: Additive margin softmax for face verification. arXiv preprint arXiv:1801.05599 (2018)
31. Wang, F., Xiang, X., Cheng, J., Yuille, A.L.: Normface: l_2 hypersphere embedding for face verification. arXiv preprint arXiv:1704.06369 (2017)
32. Wang, H., Wang, Y., Zhou, Z., Ji, X., Li, Z., Gong, D., Zhou, J., Liu, W.: Cosface: Large margin cosine loss for deep face recognition. arXiv preprint arXiv:1801.09414 (2018)
33. Wen, Y., Zhang, K., Li, Z., Qiao, Y.: A discriminative feature learning approach for deep face recognition. In: European Conference on Computer Vision. pp. 499–515. Springer (2016)
34. Wu, C.Y., Manmatha, R., Smola, A.J., Krahenbuhl, P.: Sampling matters in deep embedding learning. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2840–2848 (2017)
35. Wu, Z., Palmer, M.: Verb semantics and lexical selection. arXiv preprint arXiv:cmp-lg/9406033 (1994), https://academic.microsoft.com/paper/2951798058
36. Yang, H.M., Zhang, X.Y., Yin, F., Liu, C.L.: Robust classification with convolutional prototype learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3474–3482 (2018)
37. Zhang, X., Fang, Z., Wen, Y., Li, Z., Qiao, Y.: Range loss for deep face recognition with long-tailed training data. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 5409–5418 (2017)
38. Zheng, Y., Pal, D.K., Savvides, M.: Ring loss: Convex feature normalization for face recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 5089–5097 (2018)