ArcGrad: Angular Gradient Margin Loss for Classification

*Jiantao Wu, Lin Wang**
Shandong Provincial Key Laboratory of Network Based Intelligent Computing
University of Jinan
Jinan 250022, China
*Corresponding Author: wangplanet@gmail.com

***Abstract*—The cosine-based softmax loss functions greatly enhance intra-class compactness and perform well on the tasks of face recognition and object classification. Outperformance, however, depends on the careful hyperparameter selection. Adaptively Scaling Cosine Logits (AdaCos) tries to propose a parameter-free version by leveraging an adaptive scaling parameter. Nevertheless, the application of AdaCos is limited in specific domains because of improper approximation.**

In this paper, to promote intra-class compactness and inter-class separability, we propose an Angular Gradient Margin Loss (ArcGrad) that generates a gradient margin by maximizing the angular gradient. Our work suggests that the margin parameter on cosine-based methods is not necessary, and the scaling parameter is inversely proportional to the margin. Furthermore, a stable and large gradient promotes better feature representation. In experiments, we test our method, as well as other methods enhancing discriminative information, on CIFAR and 15 datasets from UCI. Experimental results show ArcGrad consistently outperforms both on large and small scale problems and has the superiority in discriminative information and time-consumption.

***Index Terms*—loss function, margin, angular, gradient, adacos, softmax, ArcFace**

I. INTRODUCTION

Neural networks have achieved incredible milestones on classification tasks, including email spam filtering, document categorization, speech recognition, image recognition, and handwriting recognition [1], [2]. In supervised learning, back-propagation (BP) [3] is the crucial algorithm to optimize the neural networks. It is essential to design a proper loss function for BP to learn a workable model. The most popular choice for loss function on multi-class classification is the combination of the softmax function and the cross-entropy loss. Recent studies [4], [5], however, find this traditional method fails to emphasize both intra-class compactness and inter-class separability, and the features near the decision boundary are more likely to be misclassified. Therefore, to be discriminative features helps the ability of generalization on unseen samples.

Center loss [6] learns a center for each class, and penalizes the distance between their corresponding class center and the embedded features. Usually, Center loss evaluates combining the softmax loss with a balance parameter. Research [7] indicates center loss can slightly improve discriminative information.

Another popular study line tries to enlarge the margin between the embedded features belonging to different classes. Generally, a large margin indicates a confident classification [8]. Soft-Margin Softmax (M-Softmax) [4] penalizes the predicted probabilities by an addictive margin. This work also addresses the weakness of Softmax comes from the extracted features that are not discriminative.

Cosine-based methods project the embedded features to angular space; moreover, they incorporate a scaling parameter and an adjustable hyperparameter that controls the margin. Feature normalization [9], [10] and weight normalization [11] are necessary for cosine-based methods. Therefore, they inherit the advantages of accelerating training and good generalization ability. As a result, L-Softmax [5], SphereFace [12], CosFace [13], ArcFace [7] get state-of-the-art performance on face recognition in succession. Though they have clear geometric interpretation and represent state-of-the-art methods, the high performance depends on empirically tuning those hyperparameters and some training tricks [14]. AdaCos [14] is a parameter-free variant of cosine-based loss. It tries to maximize the change rate of the classification probability $P_{i,yi}$, so that AdaCos leverages an adaptive scale parameter automatically. However, the calculation of AdaCos is not correct when the inter-class angles are not close to $\pi/2$.

In this work, our intuition is to design a new kind of loss function which has the advantages of the cosine-based method, meanwhile getting rid of the fatiguing tuning procedure. It is still challenging to get a parameter-free version for now because the real situations are various, and we can hardly tell how wide the margin is best for all problems. However, our work reveals the relationship between the scaling parameter and the margin, that is, only one hyperparameter is needed. For comprehensive comparisons, we conduct our experiments both on the large and small scale problems.

Our contributions in this work can be summarized as:

- We explain the relationship between the scaling parameter and the margin, i.e., s and m are inversely proportional. A smaller scaling parameter leads to tighter intra-class compactness.
- Based on the analysis of s and m, angular gradient margin loss (ArcGrad) is proposed to produce a soft margin by maximizing gradient. Our method not only has a simple form but also shows superiority in regularizability, computational overhead, and performance.

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

The remainder of this paper is organized as following structure. In Section II, we introduce the relevant symbols and related works. In Section III, we interpret the relationship between the scaling parameter and margin and propose a novel loss function—angular gradient margin loss. In Section IV, we conduct a comprehensive comparison with five methods from many aspects. Finally, we conclude this paper in Section V and discuss future work.

II. RELATED WORKS

That logistic regression can only produce a decimal between 0 and 1.0 limits the application on binary classification. Softmax extends the ability of neural networks into multi-class classification by assigning decimal probabilities to each class. The softmax function calculates this probability:

$$P(c=y|z)=\frac{e^{z_y}}{\sum_{j=1}^{C}e^{z_j}} \text{ for } c=1,\ldots,C, \tag{1}$$

where C is the number of classes. And then the final score will be:

$$L=-\frac{1}{N}\sum_{i=1}^{N}\log(P(c=y_i|f(x_i))), \tag{2}$$

where y_i is the ground truth label for the i-th sample x_i, N is the total number of training samples in one batch and f denotes our model (CNN on image tasks, MLP on other tasks).

A. Softmax Loss

The softmax loss [5] obtains a fully connected layer, a softmax function, and a cross-entropy loss function. The weights of the last fully connected layer are considered the centers of the corresponding class. The embedded features, outputs of our model, are getting close to these centers while training. The last fully-connected layer calculates the similarity between the embedded feature t and the center of class c by dot product, $Z_c^1=W_c^T\ t+b_c$, where W are the weight matrix of the last layer; t denotes the embedded features $f(x)$. The original softmax loss can be rewritten as:

$$L^{\text{original\_soft}}=-\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{W_{y_i}^T t_i+b_{y_i}}}{\sum_{j=1}^{C}e^{W_j^T t_i+b_j}}, \tag{3}$$

where t_i is the embedded feature of i-th sample, N is the number of a mini-batch size and y_i is the label of i-th sample.

SphereFace [12] studies the effects of bias b. They find that omitting the bias does no harm to the performance but makes it easy to analyze. So we follow this modification, and the final version of our modified softmax loss is:

$$\begin{aligned}L^{\text{Soft}}&=-\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{W_{y_i}^T t_i}}{\sum_{j=1}^{C}e^{W_j^T t_i}}\\&=-\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{\left\|W_{y_i}^T\right\|\|t_i\|\cos(\theta_{y_i})}}{\sum_{j=1}^{C}e^{\left\|W_j^T\right\|\|t_i\|\cos(\theta_j)}},\end{aligned} \tag{4}$$

where θ is the angle between two vectors; θ_j denotes the angle between this feature and the weight vector of class j or W_j — θ_{y_i} denotes the intra-class angle and $\theta_j, j\neq y_i$ denotes the inter-class angle. For convenience, we use the abbreviation Soft for Softmax in the rest of our paper.

B. Center Loss

The intuition of center loss [6] is to minimize intra-class variations. To achieve that, it decreases the euclidean distance between the feature t_i and the class center c_{y_i} directly, as formulated :

$$L^C=\frac{1}{2N}\sum_{i=1}^{N}\|t_i-c_{y_i}\|_2^2, \tag{5}$$

where c_{y_i} denotes the y_i-th class center of the feature t_i. c will converge to the center of the features of every class as the entire training set is taken into account. Besides, it needs the softmax loss to keep different classes separated. The final formulation is balanced the softmax loss (modified) and the center loss by a scale parameter λ:

$$\begin{aligned}L^{\text{Center}}&=\lambda L^C+L^{\text{Soft}}\\&=\frac{\lambda}{2N}\sum_{i=1}^{N}\|t_i-c_{y_i}\|_2^2-\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{W_{y_i}^T t_i}}{\sum_{j=1}^{C}e^{W_j^T t_i}}.\end{aligned} \tag{6}$$

C. Cosine-based Methods

Weight normalization [11] can not only accelerate training but also solve the imbalanced data problem by re-balancing the weight of each class [15]. It applies L2-normalization to reparameterize the weight vectors. L-Softmax [5] uses this trick on its last layer and benefits from it. The other trick cosine-based methods usually use is feature normalization [9], [10], which normalizing the embedded features instead of the input features to augment softmax. In this way, the features have a smaller variance and become far more away from the mismatched class than softmax.

SphereFace [12], CosFace [13], NormFace [16] and more, use both weight and feature normalization. As a result, those losses can optimize cosine similarity instead of inner-product. First, they normalize the weight $\tilde{W}=\frac{W}{\|W\|}$ and the embedded feature $\tilde{t}=\frac{t}{\|t\|}$ by l_2 normalization. Second, a scaling hyperparameter s is multiplied to re-scale the value range, so the formulation becomes

$$\begin{aligned}L^{\text{Cosine}}&=-\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{s\cdot\tilde{W}_{y_i}^T\tilde{t}_i}}{\sum_{j=1}^{C}e^{s\cdot\tilde{W}_j^T\tilde{t}_i}}\\&=-\frac{1}{N}\sum_{i=1}^{N}log\frac{e^{s\cdot\cos(\theta_{y_i})}}{\sum_{j=1}^{C}e^{s\cdot\cos(\theta_j)}}.\end{aligned} \tag{7}$$

Main difference between cosine-based methods is the way to produce margin: SphereFace [12] produces multiplicative angular margin; CosFace [13] produces additive cosine margin; ArcFace [7] produces additive angular margin. A study [17] compares the above methods, and the results show ArcFace [7] emerges as the best performing loss function in terms of accuracy, convergence rate as well as robustness.

ArcFace [7] defines an addictive angular margin parameter m to enforce the intra-class angle smaller than the inter-class angle significantly, $\theta_{y_i} < \max(\theta_j) - m$. The formulation of the ArcFace [7] can be written as:

$$L^{\text{ArcFace}} = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{s\cdot\cos(\theta_{y_i}+m)}}{e^{s\cdot\cos(\theta_{y_i}+m)}+\sum_{j=1,j\neq y_i}^{C}e^{s\cdot\cos(\theta_j)}}. \tag{8}$$

The transformation, from Cartesian coordinate system to polar coordinate system, brings a clear geography interpretation [12], a smaller generalization error, and a lower time-consumption [5]. Besides, the limited value range generates a form of regularization[1]. However, these methods have to tune hyperparameters to improve the results, which is usually depending on empirical experiments.

AdaCos [14] studies the effects of both the two hyperparameters and proposes a suitable scale $\tilde{s}$ which maximizes the change rate of predicted probability $P_{i,yi}$. s is induced by $\frac{\partial^2 P_{i,yi}}{\partial\theta_{y_i}^2}=0$:

$$s = \frac{\log(B_i)}{\cos(\theta_{y_i})}, \tag{9}$$

where $B_i = \sum_{j=1,j\neq y_i}^{C} e^{s\ \cos(\theta_j)}$. And then, combining the dynamical strategy, this formulation can be rewritten as:

$$\tilde{s}^{(l)} = \begin{cases} \sqrt{2}\cdot\log(C-1) & l=0 \\ \frac{\log B_{\text{avg}}^{(l)}}{\cos\left(\min\left(\frac{\pi}{4},\theta_{\text{med}}^{(l)}\right)\right)} & l\geq 1 \end{cases}, \tag{10}$$

where l is the iteration number of training, $\tilde{s}^0$ is the initial scale and $B_{\text{avg}}^{(l)} = \frac{1}{N}\sum_{i=1}^{N}\sum_{j\neq y_i} e^{\tilde{s}^{(l-1)}\cdot\cos\theta_{i,j}}$.

The main problem of AdaCos [14] is that this method depends on the observation that the inter-class angles $\theta_j (j\neq y_i)$ are around $\pi/2$ during training. The error of this approximation increases when the dataset is small, or the variance of θ is high. In fact, $B_{\text{avg}}^{(t)}$ is related with s, therefore, it's not appropriate to seem this item as a constant.

III. METHODOLOGY

Enlightened by the cosine-based methods, the intra-class angle has a strong indication for the classification result. Our motivation is to enlarge the minimal gap by gradient. Therefore, we design a new loss function that has only two items: a scaling parameter and the corresponding angles. Angular gradient margin loss (ArcGrad), our proposed method, though, removing the margin parameter, is still capable of generating discriminative information by maximizing the angular gradient.

[1]Later experiments in Section IV-C1 reveal cosine-based methods have a stronger ability of regularization.

The preliminary knowledge is defined in Section II-A, and we define ArcGrad:

$$\begin{aligned} L^{\text{ArcGrad}} &= -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{-s\cdot\arccos(\tilde{W}_{y_i}^T\tilde{t}_i)}}{\sum_{j=1}^{C}e^{-s\cdot\arccos(\tilde{W}_j^T\tilde{t}_i)}} \\ &= -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{-s\cdot\theta_{y_i}}}{\sum_{j=1}^{C}e^{-s\cdot\theta_j}}. \end{aligned} \tag{11}$$

In this Section, we first deduce the condition of the maximum gradient and then apply it to generate margin.

A. Maximum Gradient

First, we can rewrite Eq. 12—dividing both the molecular and denominator by $e^{-s\cdot\theta_{y_i}}$:

$$L^{\text{ArcGrad}} = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{1}{1+\sum_{j=1,j\neq y_i}^{C}e^{-s(\theta_j-\theta_{y_i})}}. \tag{12}$$

We define the minimal gap between inter-class and intra-class angles of i-th feature as:

$$m_i = \min\{\theta_j - \theta_{y_i}\}\ \text{for}\ j=1,...,C, j\neq y_i. \tag{13}$$

Combining Eq. 13 and Eq. 12, we get a larger approximation of L^{ArcGrad}:

$$L^{\text{ArcGrad}'} = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{1}{1+(C-1)e^{-s\cdot m_i}}. \tag{14}$$

We can get a large gap by optimizing $L^{\text{ArcGrad}'}$.

Our objective is to enlarge the minimal gap by maximizing the angular gradient $\frac{\partial L^{\text{ArcGrad}'}}{\partial m_i}$. Unlike the hard margin methods change the decision boundary, our method makes the embedded features in the specific region move faster by maximizing their gradient; finally, there will be little or none features in this region. Though Eq. 14 is simple, it is still unlikely to solve the maximum point of $\frac{\partial L^{\text{ArcGrad}'}}{\partial m_i}$ directly. Therefore, we use the average gap to respect all gaps,

$$\tilde{m} = \frac{1}{N}\sum_{i=1}^{N} m_i. \tag{15}$$

Now, we get an approximation of $L^{\text{ArcGrad}'}$:

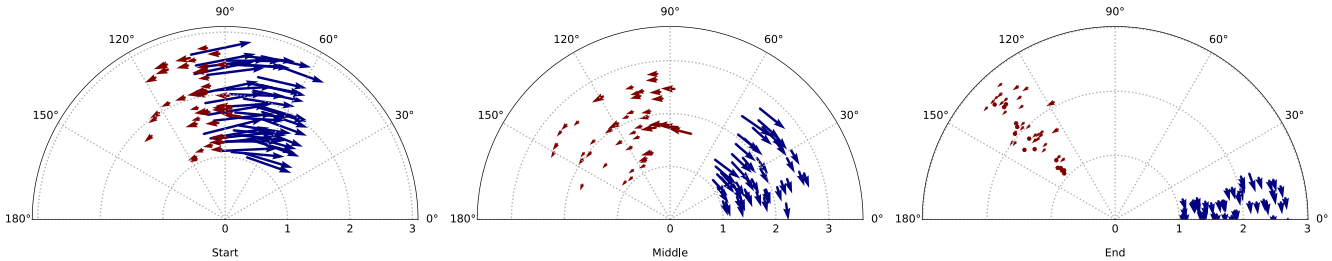
$$L^{\text{ArcGrad}''} = -\log\frac{1}{1+(C-1)e^{s\ \tilde{m}}}. \tag{16}$$

Then the function of $L^{\text{ArcGrad}''}$'s gradient is

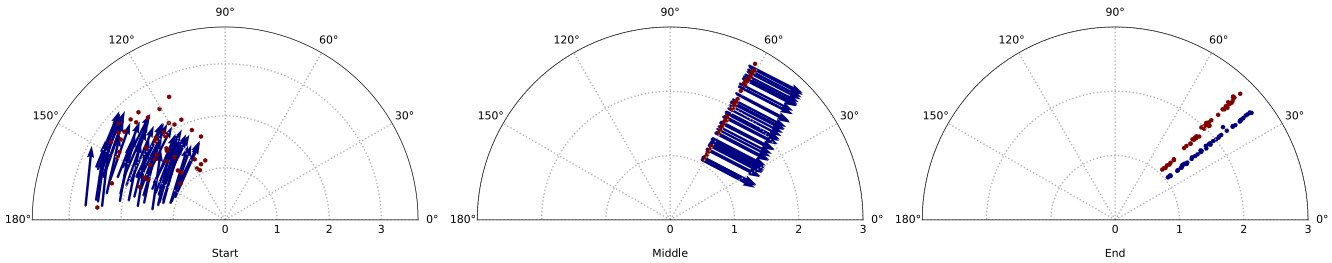
$$\frac{\partial L^{\text{ArcGrad}''}}{\partial\tilde{m}} = \frac{(C-1)e^{\tilde{m}\cdot s}s}{(C-1)e^{\tilde{m}\cdot s}+1}, \tag{17}$$

where C is the number of classes. It is a function of $\tilde{m}$ and s. Since $\tilde{m}$ can not be changed directly, the only thing we can do is to select a suitable s to make this function reach maximum. Mathematically, this function reaches its maximum when the partial derivative is equal to 0:

$$\frac{\partial^2 L^{\text{ArcGrad}''}}{\partial\tilde{m}\partial s} = 0. \tag{18}$$



(a) s=1



(b) $s = 40$

Fig. 1. Gradient visualization. The points denote the distribution of θ_{intra}(blue one) and θ_{inter}(red one) and the arrows denote the angular gradient. We randomly assign the radius of points from 1 to 3 for clear visualization.

By solving Eq. 18, the relationship between the scaling parameter s and gap $\tilde{m}$ is established:

$$s = \frac{\text{ProductLog}\left(\frac{C-1}{e}\right) + 1}{\tilde{m}}, \tag{19}$$

where ProductLog is Lambert W function.

B. Gradient Margin

Though previous studies treat the scaling parameter and the margin as isolate variables, Eq. 19 reveals they are inversely proportion. This formula indicates a small s can generate a large gap. To illustrate this phenomenon, we try two values for the scaling parameter on Iris data set. We visualize the distribution of angles and their gradient for three phases during the training procedure, as shown in Figure 1. One can see that the point within the theoretical gap has a large gradient. The result fits the theoretical gap $84\,\text{deg}$ for $s = 1$ and $2\,\text{deg}$ for $s = 40$ according to Eq. 19.

Finally, these features generate a clear margin on angular space, and we call this margin "**gradient margin**". Traditional methods change the decision boundary to generate a hard margin between embedded features belonging to different classes. In contrast, ArcGrad emphasizes the gradient of features, remaining the decision boundary unchanged. We argue that the margin parameter is not necessary on cosine-based methods, and the scaling parameter is enough to generate discriminative information. Though a small s leads to small intra-class angles, it may cause unstable situations. There are some tricks to that situation: adjustable s from an enormous value to a small one; pre-trained parameters

IV. EXPERIMENTS

In this section, we conduct our experiments for two aspects: the diversity of resolvable problems and the performance improvement on specific problems. The UCI machine learning repository [18] contains various datasets from different domains and is a common choice to test the proposed method. For the datasets from UCI repository, we examine the robustness of our method. The CIFAR10/100 [19], containing ten classes, are widely used for image classification tasks. It is separated into a training set with 50000 samples and a test set with 10000 samples. CIFAR10+/100+ denotes the CIFAR10/100 with data augment. For the data augmentation, we follow the transformations in [20]: a 32×32 random cropping with 4-pixel padding on each side, a random flipping with the probability of 0.5, and a z-score normalization. It is proved the discriminative information helps the performance on this task [4], [5]. Therefore, six methods are compared for a comprehensive study on the discriminative information and performance. They are the original softmax loss (Soft), Center loss (Center) [6], soft-margin softmax (M-Soft) [4], ArcFace [7], AdaCos [14], and the angular gradient margin loss (ArcGrad).

A. Results on UCI Repository

We select 15 data sets from UCI, and the details on these data sets are shown in Table II. On each data set, we perform 10 runs of 10-fold cross-validation with random partitions. A 3-layer feed-forward neural network with ReLU[21] activation function is trained on the training set, and then the trained model evaluates the accuracy on the test set.

TABLE I
AVERAGE ACCURACY, WINS FOR EACH METHOD AND DATA SET. ARCGRAD30 DENOTES ARCGRAD WITH $s = 30$, ARCGRAD5 DENOTES ARCGRAD WITH $s = 5$.

Data Set	Soft	Center	M-Soft	ArcFace	AdaCos [14]	ArcGrad30	ArcGrad5
PimaIndiansDiabetes	77.81 ± 5.60	77.78 ± 5.84	77.87 ± 5.77	74.64 ± 11.93	50.42 ± 17.67	77.85 ± 6.19	**78.42 ± 6.08**
Hayes-Roth	57.16 ± 12.51	57.83 ± 13.68	59.11 ± 12.9	72.28 ± 19.38	72.59 ± 13.8	79.2 ± 12.98	**79.32 ± 12.34**
UserKnowledgeModeling	94.57 ± 7.29	94.12 ± 5.39	**96.00 ± 2.67**	94.66 ± 8.3	91.67 ± 5.22	95.81 ± 2.59	95.95 ± 2.39
BreastCancerWisconsin	**73.22 ± 17.23**	73.15 ± 17.43	73.05 ± 18.34	70.11 ± 17.67	53.61 ± 24.22	71.4 ± 17.13	70.75 ± 17.05
WholesaleCustomers	86.64 ± 7.35	87.91 ± 6.87	87.96 ± 6.68	80.77 ± 19.47	46.3 ± 16.52	**89.47 ± 5.55**	89.11 ± 5.22
TeachingAssistantEvaluation	53.40 ± 14.26	52.44 ± 13.63	54.06 ± 14.98	38.40 ± 10.89	**55.62 ± 14.47**	53.19 ± 14.24	53.81 ± 12.28
BanknoteAuthentication	98.64 ± 1.20	97.92 ± 4.48	99.01 ± 0.89	83.94 ± 25.19	50.85 ± 8.6	99.50 ± 0.92	**99.51 ± 0.97**
CongressionalVotingRecords	92.78 ± 6.97	**93.75 ± 6.17**	92.62 ± 6.92	92.18 ± 6.97	49.95 ± 9.18	92.37 ± 7.12	92.83 ± 6.69
WebsitePhishing	83.05 ± 1.87	83.06 ± 1.77	83.47 ± 1.97	**84.48 ± 2.61**	83.72 ± 2.2	84.18 ± 2.59	84.18 ± 2.83
Haberman	71.84 ± 1.09	71.91 ± 1.29	72.03 ± 2.09	72.64 ± 3.79	49.44 ± 21.46	72.97 ± 4.18	**73.19 ± 3.48**
Iris	96.80 ± 6.76	96.75 ± 5.62	**97.53 ± 4.78**	**97.53 ± 4.28**	95.73 ± 6.15	97.00 ± 4.26	96.93 ± 4.26
BreastTissue	62.86 ± 10.76	59.44 ± 9.14	64.14 ± 11.16	65.71 ± 9.89	65.87 ± 10.03	66.50 ± 9.73	**66.71 ± 10.21**
VertebralColumn	72.62 ± 7.18	72.62 ± 7.36	74.1 ± 7.52	76.99 ± 12.41	77.28 ± 9.76	78.45 ± 10.36	**78.81 ± 10.81**
ForestTypeMapping	82.33 ± 7.99	82.69 ± 8.18	83.61 ± 6.94	82.83 ± 16.94	**87.24 ± 5.54**	86.81 ± 6.26	85.67 ± 8.55
BalanceScale	89.65 ± 5.79	88.75 ± 4.37	90.05 ± 5.39	91.98 ± 7.02	88.08 ± 4.95	91.87 ± 5.97	**92.35 ± 6.05**
Average	79.56	79.34	80.31	78.61	67.89	82.44	**82.50**
Wins	1	1	2	1	2	1	**7**

TABLE II
DATA SETS INFORMATION.

Data Set	Classes	Feature	Instance
PimaIndiansDiabetes	2	8	392
Hayes-Roth	3	3	160
UserKnowledgeModeling	4	5	403
BreastCancerWisconsin	2	33	194
WholesaleCustomers	2	7	440
TeachingAssistantEvaluation	3	5	151
BanknoteAuthentication	2	4	1372
CongressionalVotingRecords	2	16	232
WebsitePhishing	3	9	1353
Haberman	2	3	306
Iris	3	4	150
BreastTissue	6	9	106
VertebralColumn	3	6	310
ForestTypeMapping	4	27	523
BalanceScale	3	4	625

TABLE III
THE TRAINING SETTINGS ON CIFAR10/100. "LR" DENOTES THE LEARNING RATE. IN ALL EXPERIMENTS, WEIGHT DECAY IS 0.0005 AND THE OPTIMIZER IS ADAM [22].

Item	CIFAR10/10+	CIFAR100+
Architecture	ResNet18	ResNet34
Epochs	120	160
Batch Size	256	128
LR	0.1	0.01

Table I shows the average accuracy for compared methods and the number of times each method produced the highest accuracy (wins). Though Soft only wins once for all 15 tasks, it performs better than Center [6], ArcFace [7], and AdaCos [14] on average. The discriminative information does not help the performance of general tasks. However, our method outperforms both on average and on specific tasks. Unlike the other hard margin methods, our method generates a soft margin by gradient, reducing the risk of over-fitting.

TABLE IV
CLASSIFICATION ACCURACY RATE (%) ON CIFAR10/10+. "DIS" DENOTES WHETHER THIS METHOD ENHANCES DISCRIMINATIVE INFORMATION. "MARGIN" DENOTES WHETHER THIS METHOD IS A MARGIN-BASED METHOD.

Dis	Margin	Method	CIFAR10+	CIFAR10
False	False	Soft	93.50	86.2
True		Center	93.54	86.43
	True	M-Soft	93.55	86.53
		ArcFace	94.23	87.62
		AdaCos [14]	94.10	87.30
		ArcGrad	**94.36**	**87.83**

TABLE V
THE TOP-1 AND TOP-5 CLASSIFICATION ACCURACY RATE (%) ON CIFAR100+.

Dis	Margin	Method	Top-1	Top-5
False	False	Soft	72.55	89.56
True		Center	74.48	**91.54**
	True	M-Soft	73.02	90.63
		ArcFace	74.55	89.73
		AdaCos	72.93	85.97
		ArcGrad	**75.12**	90.98

B. Results on Image Data

The architecture is not the main point of our work, therefore we choose the-state-of-the-art one for experiments. Deep residual networks [23], widely used in image classification tasks [24], significantly improves the performance of the deep convolutional neural networks. We also use some modern training techniques introduced by Leslie Smith [25]. It provides a principle to choose maximal learning rate, batch size, and introduces one cycle policy to train model fast without losing performance. We summarize our experimental settings on Table III.

For the hyperparameter settings of the compared losses, the scaling parameter s is set as 20 and the margin m is set as 0.1 for ArcFace [7]; λ is set as 0.02 for Center loss [6]; s is set as 10 for ArcGrad; m is set as 0.5 for M-Soft [4]. For

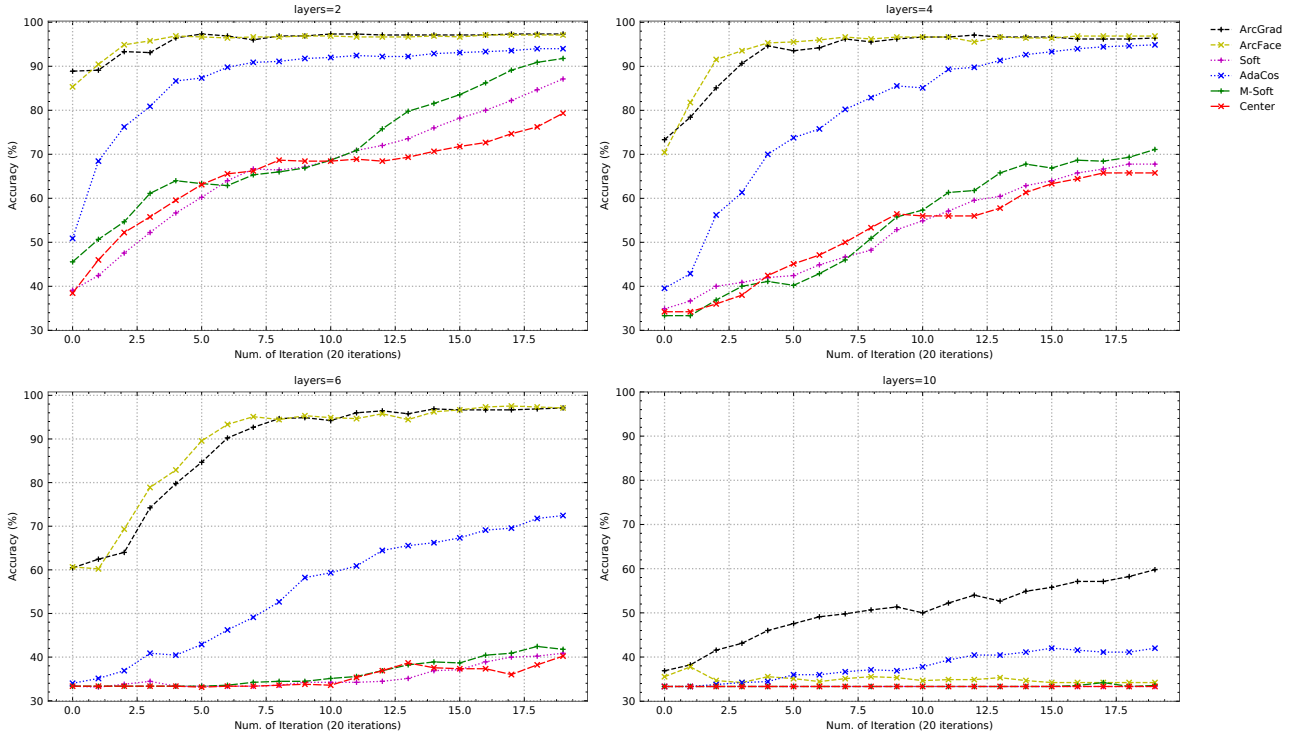


Fig. 2. The sensitivity of the layers for different losses.

each method and data set, we get the average classification accuracy of five experiments, as shown in Table IV and V. Our method gets the best results of three of four benchmarks.

C. Model Analysis

1) Regularizability: Similar to other regularization techniques, our method has a strong ability to regularize the model. We increase the complexity of our model by adding the number of layers. We compare the effects of the number of layers on Iris data set. For each experiment, we take the average curve of 10 runs of 10-fold cross-validation with random partitions. As shown in Figure 2, our method has the smallest accuracy decay comparing with the other methods when the number of layers increases. Particularly, the accuracy of Soft, ArcFace [7], and AdaCos [14] is around 33%, which is the result of random choice when the model has 10 layers. Our method reveals a strong tolerance against model changing.

Over-fitting usually means the selected model is too complex, or the training set is insufficient [26]. To investigate the performance under insufficient data, we cut out parts of the original training samples on CIFAR10+. For each experiment, we train our model 20 epochs, which are enough to produce a significant difference. Figure 3 compares the performance decay on cosine-based methods for the different remaining percentage of training data. The advantage of our method emerges when training data is cut back.

2) Intra-class Compactness and Inter-class Separability: For better measuring both intra-class compactness and inter-class separability, we use the following three metrics [7]:

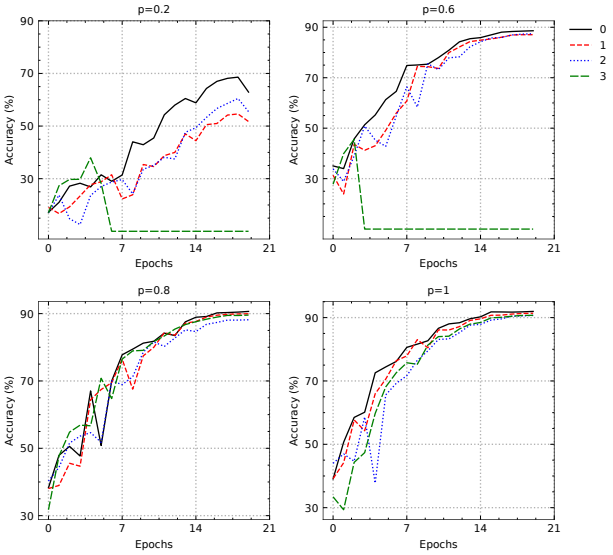


Fig. 3. Performance decay. p denotes the remaining percentage of training samples. 0, 1, 2, 3 denote ArcGrad, ArcFace, Soft, and AdaCos [14].

$$\text{WC-Intra} = \frac{1}{C}\sum_{j=1}^{C}\langle \text{Center}_j, W_j\rangle, \tag{20}$$

$$\text{W-Inter} = \frac{1}{C^2 - C}\sum_{j=1}^{C}\sum_{i=1, i\neq j}^{C}\langle W_i, W_j\rangle, \tag{21}$$

TABLE VI
THE ANGLE STATISTICS (RAD) UNDER DIFFERENT LOSSES ON CIFAR10.

Metric	Soft	Center	M-Soft	ArcFace	AdaCos	ArcGrad
WInter	1.37	1.63	1.34	1.55	1.42	**1.68**
CInter	1.52	1.63	1.67	**1.68**	1.55	1.66
Intra	0.80	0.42	0.69	0.39	0.41	**0.24**

TABLE VII
ACCURACY COMPARISON OVER DIFFERENT NUMBER OF EPOCHS.

Epochs	Soft	Center	M-Soft	ArcFace	AdaCos	ArcGrad
10	72.84	75.08	83.38	87.70	85.81	**88.34**
20	87.05	85.90	85.93	91.72	91.20	**91.77**
40	89.98	91.71	91.89	92.92	92.80	**93.38**
90	93.39	93.69	93.38	94.31	94.10	**94.59**

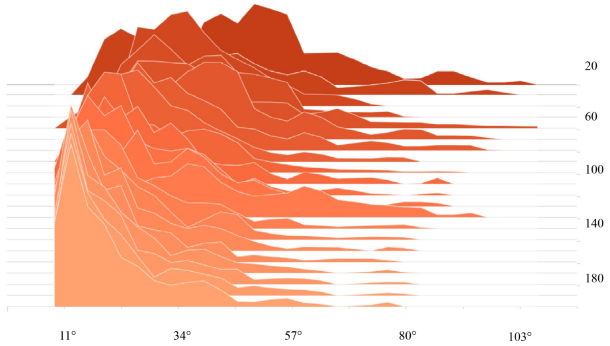
$$\text{C-Inter} = \frac{1}{C^2 - C} \sum_{j=1}^{C} \sum_{i=1, i \neq j}^{C} \langle \text{Center}_j, W_j \rangle, \qquad (22)$$

where C is the number of class, $\text{Center}_j = \bar{x}$ is the mean of the embedded features belonging to the same class j, and $\langle a, b \rangle$ denotes the angle between the vector a and b. "W-Inter" refers to the mean of angles between different target vectors W_j^T. "C-Inter" refers to the mean of angles between different classes' feature center. "WC-Intra" refers to the mean of the angles between target vectors W_j^T and feature center of the class j.

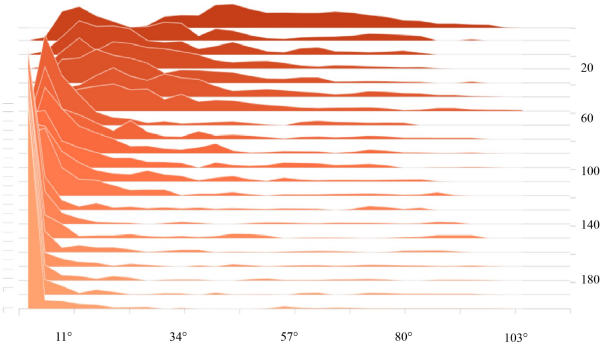
Table VI shows the final results of these angle statistics under compared losses on CIFAR10. One can see that Softmax fails to enhance intra-class compactness; Center [6] and M-Soft [4] can slightly compress intra-class variations; AdaCos [14] can decrease "WC-Intra" but also brings in smaller inter-class angles; ArcFace [7] and ArcGrad greatly enhance both inter-class separability and intra-class compactness.

We also give the angle histograms to capture the dynamic change of the intra-class angles in Figure 4. Each slice in the figure displays the intra-class angle histogram. The slices depict the change of angle during the training process. In comparison with the softmax method, our approach reduces the intra-class angle faster and makes it be at a lower level.

3) Computational Efficiency: In deep learning, the model and data set usually are huge. Therefore, time-consumption is crucial to industry application. Our method can learn not only well but also fast. For the training settings, we only change the number of epochs to train our model while keeping the other settings the same as in Table III. Results in Table VII reveal our method consistently outperforms others within the same training epochs. In particular, our method is 15.5% higher than Soft when only training the model for 10 epochs. Our method only includes an extra operation on the criterion function. So there will be a constant additional calculation time. In practice, the calculation cost of our method is 3.4% higher than Softmax, and it will be lower as the network deeper. Besides, this cost is negligible comparing to the reduction of iteration numbers.



(a) Soft



(b) ArcGrad

Fig. 4. Visualization of the angle histogram in tensorboard [27]. The top slice referring to the angle histogram at epoch 0 is darker and the lower slices referring to higher epoch are lighter.

V. CONCLUSION

In this paper, we proposed an Angular Gradient Margin Loss function (ArcGrad) for backpropagation neural networks. ArcGrad generates the inter-class angular margin by maximizing the angular gradients. We argue that the scaling parameter is capable of producing a margin, and it is not necessary to include an extra hyper-parameter. Our approach reduces the complexity of cosine-based methods and avoids over-fitting in a soft way, compared with the hard constraint.

In the future, the inner relationship between embedded features requires further investigation for outlier detection, as the experiments demonstrate that they are not independent with each other.

ACKNOWLEDGMENT

This work was supported by National Natural Science Foundation of China under Grant No. 61872419, No. 61573166, No. 61572230, No. 61873324, No. 81671785, No. 61672262, No. 61903156. Shandong Provincial Natural Science Foundation No. ZR2019MF040, No. ZR2018LF005. Shandong Provincial Key R&D Program under Grant No. 2019GGX101041, No. 2018CXGC0706, No. 2017CXZC1206. Taishan Scholar Project of Shandong Province, China, under Grant No. tsqn201812077.

References

[1] Y. Sun, Y. Chen, X. Wang, and X. Tang, "Deep learning face representation by joint identification-verification," in *Advances in neural information processing systems*, 2014, pp. 1988–1996.

[2] 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*, 2014, pp. 1701–1708.

[3] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[4] X. Wang, S. Zhang, Z. Lei, S. Liu, X. Guo, and S. Z. Li, "Ensemble soft-margin softmax loss for image classification," in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, 2018, pp. 992–998.

[5] W. Liu, Y. Wen, Z. Yu, and M. Yang, "Large-margin softmax loss for convolutional neural networks.," in *ICML*, vol. 2, 2016, p. 7.

[6] Y. Wen, K. Zhang, Z. Li, and Y. Qiao, "A comprehensive study on center loss for deep face recognition," *International Journal of Computer Vision*, vol. 127, no. 6-7, pp. 668–683, 2019.

[7] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "Arcface: Additive angular margin loss for deep face recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4690–4699.

[8] Y. Lin, "A note on margin-based loss functions in classification," *Statistics & probability letters*, vol. 68, no. 1, pp. 73–82, 2004.

[9] Y. Zheng, D. K. Pal, and M. Savvides, "Ring loss: Convex feature normalization for face recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 5089–5097.

[10] X. Zhang, Z. Fang, Y. Wen, Z. Li, and Y. Qiao, "Range loss for deep face recognition with long-tailed training data," in *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, 2017, pp. 5419–5428.

[11] T. Salimans and D. P. Kingma, "Weight normalization: A simple reparameterization to accelerate training of deep neural networks," in *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, 2016, p. 901.

[12] W. e. a. Liu, "Sphereface: Deep hypersphere embedding for face recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 212–220.

[13] H. e. a. Wang, "Cosface: Large margin cosine loss for deep face recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 5265–5274.

[14] X. Zhang, R. Zhao, Y. Qiao, X. Wang, and H. Li, "Adacos: Adaptively scaling cosine logits for effectively learning deep face representations," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, 2019, pp. 10 823–10 832.

[15] Y. Guo and L. Zhang, *One-shot face recognition by promoting underrepresented classes*, 2017. arXiv: 1707.05574 [cs.CV].

[16] F. Wang, X. Xiang, J. Cheng, and A. L. Yuille, "Normface: L_2 hypersphere embedding for face verification," in *Proceedings of the 2017 ACM on Multimedia Conference, MM 2017, Mountain View, CA, USA, October 23-27, 2017*, 2017, pp. 1041–1049.

[17] Y. Srivastava, V. Murali, and S. R. Dubey, "A performance comparison of loss functions for deep face recognition," *arXiv preprint arXiv:1901.05903*, 2019.

[18] D. Dua and C. Graff, *UCI machine learning repository*, 2017.

[19] A. Krizhevsky, V. Nair, and G. Hinton, "Cifar-10 and cifar-100 datasets," *URl: https://www. cs. toronto. edu/kriz/cifar. html*, vol. 6, 2009.

[20] C.-Y. Lee, S. Xie, P. Gallagher, Z. Zhang, and Z. Tu, "Deeply-supervised nets," in *Artificial intelligence and statistics*, 2015, pp. 562–570.

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

[22] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[23] 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*, 2016, pp. 770–778.

[24] S. e. a. Hershey, "Cnn architectures for large-scale audio classification," in *2017 ieee international conference on acoustics, speech and signal processing (icassp)*, IEEE, 2017, pp. 131–135.

[25] L. N. Smith, *A disciplined approach to neural network hyper-parameters: Part 1 – learning rate, batch size, momentum, and weight decay*, 2018. arXiv: 1803.09820 [cs.LG].

[26] G. Panchal, A. Ganatra, P. Shah, and D. Panchal, "Determination of over-learning and over-fitting problem in back propagation neural network," *International Journal on Soft Computing*, vol. 2, no. 2, pp. 40–51, 2011.

[27] M. et al., *TensorFlow: Large-scale machine learning on heterogeneous systems*, Software available from tensorflow.org, 2015.