This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

$\mathcal{G}$-Softmax: Improving Intraclass Compactness and Interclass Separability of Features

Yan Luo, Yongkang Wong, *Member, IEEE*, Mohan Kankanhalli, *Fellow, IEEE*, and Qi Zhao, *Member, IEEE*

***Abstract*— Intraclass compactness and interclass separability are crucial indicators to measure the effectiveness of a model to produce discriminative features, where intraclass compactness indicates how close the features with the same label are to each other and interclass separability indicates how far away the features with different labels are. In this paper, we investigate intraclass compactness and interclass separability of features learned by convolutional networks and propose a Gaussian-based softmax ($\mathcal{G}$-softmax) function that can effectively improve intraclass compactness and interclass separability. The proposed function is simple to implement and can easily replace the softmax function. We evaluate the proposed $\mathcal{G}$-softmax function on classification data sets (i.e., CIFAR-10, CIFAR-100, and Tiny ImageNet) and on multilabel classification data sets (i.e., MS COCO and NUS-WIDE). The experimental results show that the proposed $\mathcal{G}$-softmax function improves the state-of-the-art models across all evaluated data sets. In addition, the analysis of the intraclass compactness and interclass separability demonstrates the advantages of the proposed function over the softmax function, which is consistent with the performance improvement. More importantly, we observe that high intraclass compactness and interclass separability are linearly correlated with average precision on MS COCO and NUS-WIDE. This implies that the improvement of intraclass compactness and interclass separability would lead to the improvement of average precision.**

***Index Terms*— Compactness and separability, deep learning, Gaussian-based softmax, multilabel classification.**

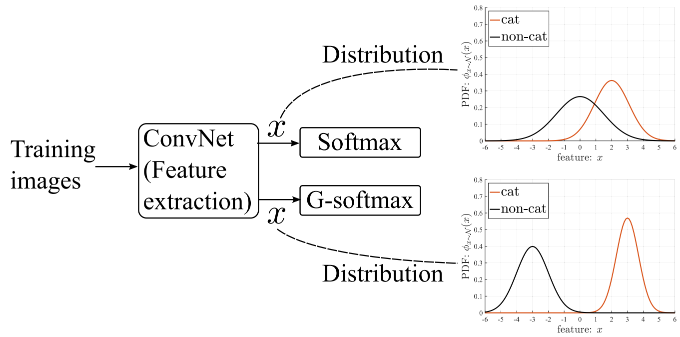


Fig. 1. Illustration to show the benefits of improving interclass separability and intraclass compactness. Given a model, an input image will be encoded to yield a discriminative feature x that is used to compute the class-dependent confidence p. As shown in the figure, interclass separability encourages to the distribution with respect to a label to be distant from the distributions with respect to the other labels, and intraclass compactness encourages the features with respect to the ground truth label to be close to the mean. PDF stands for probability density function.

I. Introduction

Machine learning is an important and fundamental component in visual understanding tasks. The core idea of supervised learning is to learn a model that explores the causal relationship between the dependent variables and the predictor variables. To quantify this relationship, the conventional approach is to make a hypothesis on the model and feed the observed pairs of dependent variables and predictor parameters to the model for predicting future cases. For most learning problems, it is infeasible to make a perfect hypothesis that matches the underlying pattern, whereas a badly designed hypothesis often leads to a model that is more complicated than necessary and violates the principle of parsimony. Therefore, when designing or evaluating a model, the core objective is to seek a balance between two conflicting goals: how complicated a model should be to achieve accurate predictions and how to design a model as simple as possible, but not simpler.

In the past decade, deep learning methods have significantly accelerated the development of machine learning research, where convolutional network (ConvNet) has achieved superior performance in numerous real-world visual understanding tasks [1], [11], [12], [16], [17], [22], [39], [41], [45], [56], [61], [68]. Although their architectures vary with each other, the softmax function is widely used along with the cross-entropy loss at the training phase [14], [22], [23], [47], [50]. The softmax function may not take the distribution pattern of the previously observed samples into account to boost classification accuracy. In this paper, we design a statistically driven extension of the softmax function that fits into the stochastic gradient descent (SGD) scheme for end-to-end learning. Furthermore, the final layer of the softmax function directly connects to the predictions and can maximally preserve generality for various ConvNets, i.e., avoid complex modification of existing network architectures.

Features are the key to prediction in ConvNet learning. According to the central limit theorem [20], the arithmetic mean of a sufficiently large number of iterates of independent and identically distributed random variables, each with a finite expected value and variance, can be approximately normally

Manuscript received June 5, 2018; revised December 20, 2018; accepted March 30, 2019. This research was funded in part by the NSF under Grant 1849107, in part by the University of Minnesota Department of Computer Science and Engineering Start-up Fund (QZ), and in part by the National Research Foundation, Prime Minister's Office, Singapore under its Strategic Capability Research Centres Funding Initiative. *(Corresponding author: Qi Zhao.)*

Y. Luo and Q. Zhao are with the Department of Computer Science and Engineering, University of Minnesota, Minneapolis, MN 55455 USA (e-mail: luoxx648@umn.edu; qzhao@cs.umn.edu).

Y. Wong and M. Kankanhalli are with the School of Computing, National University of Singapore, Singapore 117417 (e-mail: yongkang.wong@nus.edu.sg; mohan@comp.nus.edu.sg).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2019.2909737

distributed even if the original variables are not normally distributed. This makes the Gaussian distribution generally valid in a great variety of contexts. Following this line of thought, online learning methods [7], [8], [53] assumed that the weights follow Gaussian distribution and make use of its distribution pattern for classification. Given a large-scale training data [44], the underlying distributions of discriminative features generated by ConvNets can be modeled. This distribution pattern has not been fully explored in the existing literature.

Intraclass compactness and interclass separability of features are generally correlated with the quality of the learned features. If intraclass compactness and interclass separability are simultaneously maximized, the learned features are more discriminative [35]. We introduce a variant of the softmax function, named Gaussian-based softmax ($\mathcal{G}$-softmax) function, which aims to improve intraclass compactness and interclass separability, as shown in Fig. 1. We assume that features are distributed according to Gaussian distributions. Consequently, Gaussian cumulative distribution function (CDF) is used in prediction and normalization to generate the final confidence in a soft form.

Fig. 2 shows the role and position of the proposed $\mathcal{G}$-softmax function in a supervised learning framework. Given the training samples, the feature extractor would extract the features and then pass them to the predictor for inference. In this paper, we follow the mainstream deep learning framework where the feature extractor is modeled with a ConvNet. The proposed $\mathcal{G}$-softmax function is able to replace the softmax function. The contributions can be summarized as follows.

1) With the general assumption, i.e., features with respect to a class are subjected to a Gaussian distribution, we propose the $\mathcal{G}$-softmax function that models the distributions of features for better prediction. The experiments on CIFAR-10, CIFAR-100 [21], and Tiny ImageNet[1] show that the proposed $\mathcal{G}$-softmax function consistently outperforms the softmax and L-softmax function on various state-of-the-art models. In addition, we apply the proposed $\mathcal{G}$-softmax function to solve the multilabel classification problem, which yields a better performance than the softmax function on MS COCO [30] and NUS-WIDE [3]. The source code is available[2] and is easy for use.
2) The proposed $\mathcal{G}$-softmax function can quantify the compactness and separability. Specifically, for each learned Gaussian distribution, the corresponding mean and variance indicate the center and compactness of the predictor.
3) In our analysis of correlation between intraclass compactness (or interclass separability) and average precision, we observe that high intraclass compactness and interclass separability are linearly correlated with average precision (AP) on MS COCO and NUS-WIDE. This implies that the improvement of intraclass compactness and interclass separability would lead to the improvement of average precision.

[1]https://tiny-imagenet.herokuapp.com/
[2]https://gitlab.com/luoyan/gsoftmax

II. Related Works

A. Gaussian-Based Online Learning

We first review the Gaussian-based online learning methods. In the online learning context, the training data are provided in a sequential order to learn a predictor for unobserved data. These methods usually make some assumptions to minimize the cumulative disparity errors between the ground truth and the predictions over the entire sequence of instances [6]–[8], [43], [53]. In this sense, these works can give some guidance and inspiration for designing a flexible mapping function.

In contrast to Passive-Aggressive model [6], Dredze *et al.* [8] made an explicit assumption on the weights $w \in \mathbb{R}^m$: $w \sim \mathcal{N}(\mu, \Sigma)$, where μ is the mean of the weights w and $\Sigma \in \mathbb{R}^{m\times m}$ is a covariance matrix for the underlying Gaussian distribution. Given an input instance $x_i \in \mathbb{R}^m$ with the corresponding label y_i, the multivariate Gaussian distribution over weight vectors induces a univariate Gaussian distribution over the margin: $y_i(\langle w, x_i\rangle) \sim \mathcal{N}(y_i(\langle \mu, x_i\rangle), x_i^\top \Sigma x_i)$, where $\langle\cdot,\cdot\rangle$ is the inner product operation. Hence, the probability of a correct prediction is $\Pr[y_i(\langle w, x_i\rangle) \geq 0]$. The objective is to minimize the Kullback–Leibler divergence between the current distribution and the ideal distribution with the constraint that the probability of a correct prediction is not smaller than the confidence hyperparameter $\beta \in [0, 1]$, i.e., $\Pr[y_i(\langle w, x_i\rangle) \geq 0] \geq \beta$. With the mean of the margin $\mu_M = y_i(\langle \mu, x_i\rangle)$ and the variance $\sigma_M^2 = x_i^\top \Sigma x_i$, the constraint can lead to $y_i(\langle \mu, x_i\rangle) \geq \Phi^{-1}(\beta)(x_i^\top \Sigma x_i)^2$, where Φ is the cumulative function of the Gaussian distribution. This inequality is used as a constraint in optimization in practice. However, it is not convex with respect to Σ and Dredze *et al.* [8] linearized it by omitting the square root: $y_i(\langle \mu, x_i\rangle) \geq \Phi^{-1}(\beta)(x_i^\top \Sigma x_i)$. To solve this nonconvex problem, Crammer *et al.* [7] discovered that a change in variable helps to maintain the convexity, i.e., when $\Sigma = \Upsilon^2$, the constraint becomes $y_i(< \mu, x_i >) \geq \Phi^{-1}(\beta)\|\Upsilon x_i\|$. The confidence-weighted method [7] employs an aggressive updating strategy by changing the distribution to satisfy the constraint imposed by the current instance, which may incorrectly update the parameters of the distribution when handling a mislabeled instance. Therefore, Wang *et al.* [53] introduced a tradeoff parameter C to balance the passiveness and aggressiveness.

The aforementioned online learning methods [7], [8], [53] hypothesize that the weights are subjected to a multivariate Gaussian distribution and predefine a confidence hyperparameter β to formalize a constraint for optimization. Nevertheless, the weights are learned based on the training data, and putting hypothesis on the weights could be similar to put the cart before the horse. Moreover, such confidence hyperparameter may not be flexible or adaptive for various data sets. In this paper, we instead hypothesize that the features are subjected to Gaussian distribution and there is no confidence hyperparameter. To update the weights, [7], [8], and [53] apply the Lagrangian method to compute the optimal weights. This mechanism does not straightforwardly fit into SGD scheme. Along the same line, this paper is motivated to investigate how to incorporate the Gaussian assumption in SGD.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

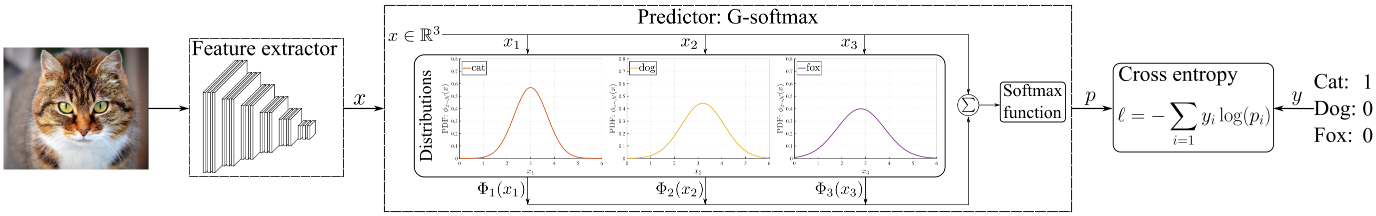


Fig. 2. Illustration of the learning framework with the proposed $\mathcal{G}$-softmax function. The proposed $\mathcal{G}$-softmax function plays as a predictor to yield prediction confidences p with respect to each class by taking input feature x and its distribution into account. $\Phi_i(\cdot)$ is the cdf. The distributions are updated by the gradient descent methods via backpropagation. In the mainstream ConvNets [14], [22], [47], [50], the predictor is often the softmax function.

B. Softmax Function in ConvNet Learning

The success of ConvNets is largely attributed to the layer-stacking mechanism. Despite its effectiveness in complex real-world visual classification, this mechanism will result in coadaptation and overfitting. To prevent the coadaptation problem, Hinton *et al.* [15] proposed a method which randomly omits a portion of neurons in a feedforward network. Then, Srivastava *et al.* [49] introduced the dropout unit to minimize overfitting and presented a comprehensive investigation of its effect in ConvNets. Similar regularization methods are also proposed in [13] and [51]. Instead of modifying the connection between the layers, [63] replaced the deterministic pooling with the stochastic pooling for regularizing ConvNets. The proposed $\mathcal{G}$-softmax function can be used together with these models to offer better general ability. We posit a general assumption and establish Gaussian distributions over the feature space at the final layer, i.e., the softmax module. In other words, the proposed $\mathcal{G}$-softmax function is general for most ConvNets without requiring much modification of the network structure.

ConvNets [14], [19], [22], [23], [47], [50], [60], [62] have strong representational ability in learning invariant features. Although their architectures vary with each other, the softmax function is widely used along with cross-entropy loss at the training phase. Hence, the softmax module is important and general for ConvNets. Liu *et al.* [35] introduced a large-margin softmax function to enhance the compactness and the separability from a geometric perspective. Substantially, the large-margin softmax function is fundamentally similar to the softmax function, i.e., both use the exponential function, while having different inputs for the exponential function. In contrast, we model the mappings between features and ground truth labels as Gaussian cdf. Similar to the softmax function, we utilize normalization to identify the maximal element but not its exact value.

C. Multilabel Classification

Multilabel classification is a special case of multioutput learning tasks. Read *et al.* [42] proposed the classifier chain model to model label correlations. In particular, label order is important for chain classification models. A dynamic programming-based classifier chain algorithm [31] was proposed to find the globally optimal label order for the classifier chain models. Shen *et al.* [46] introduced the coembedding and cohashing method that explores the label correlations from the perspective of cross-view learning to improve prediction accuracy and efficiency. On the other hand, the classifier chain model does not take the order of difficulty of the labels into account. Therefore, the easy-to-hard learning paradigm [34] was proposed to make good use of the predictions from simple labels to improve the predictions from hard labels. Liu and Tsang [33] presented a comprehensively theoretical analysis on the curse of dimensionality of decision tree models and introduced a sparse coding tree framework for multilabel annotation problems. In multilabel prediction, a large-margin metric learning paradigm [32] was introduced to reduce the complexity of decoding procedure in the canonical correlation analysis and maximum margin output coding methods. Liu *et al.* [36] introduced a large-margin metric learning method to efficiently learn an appropriate distance metric for multioutput problems with theoretical guarantee.

Recently, there have been attempts to apply deep networks in multilabel classification, especially ConvNets and recurrent neural networks (RNNs), for their promising performance in various vision tasks. In [52], ConvNet and RNN are utilized together to explicitly exploit the label dependences. In contrast to [52], [65] proposed a regional latent semantic dependences model to predict small-size objects and visual concepts by exploiting the label dependences at the regional level. Similarly, [10] automatically selected the relevant image regions from global image labels using weakly supervised learning. Zhao *et al.* [67] reduced irrelevant and noisy regions with the help of region gating module. These region proposal-based methods usually suffer from redundant computation and suboptimal performance. Wang *et al.* [54] addressed these problems by developing a recurrent memorized-attention module, and the module allows to locate attentional regions from the ConvNet's feature maps. Instead of utilizing the label dependences, [27] proposed a novel loss function for pairwise ranking, and the loss function is smooth everywhere so that it is easy to optimize within ConvNets. In addition, there are two works that focus on improving the architectures of the networks for multilabel classification [9], [69]. In this paper, we adopt a common baseline, i.e., ResNet-101 [14], which is widely used in the state-of-the-art models [9], [69].

III. Methodology

A. $\mathcal{G}$-Softmax Function

Logistic function, i.e., sigmoid function, and hyperbolic tangent function are widely used in deep learning, whose graphs are "S-shaped" curves. Their curves imply a graceful balance between linearity and nonlinearity [38]. The Gaussian cdf has the same monotonicity as logistic and hyperbolic tangent function and shares similar shapes. It makes the Gaussian cdf a potential substitute with the capability to model the distribution pattern with class-dependent μ and σ. Fundamentally, the softmax function in mainstream deep learning models is the normalized exponential function, which is a generalization of the logistic function. In this paper, the proposed $\mathcal{G}$-softmax function uses the Gaussian cdf to substitute the exponential function.

Similar to the softmax loss, we use cross entropy as the loss function, that is

$$\ell = -\sum_{i=1}^{m} y_i \log(p_i) \tag{1}$$

where ℓ is the loss, $y_i \in \{0, 1\}$ is the label with respect to the ith category, p_i is the prediction confidence with respect to the ith category, and m is the number of categories. Conventionally, given features x that with respect to various labels, p_i is given by the softmax function

$$p_i = \frac{e^{x_i}}{\sum_{j=1} e^{x_j}}. \tag{2}$$

The softmax function can be considered to represent a categorical distribution. By normalizing exponential function, the largest value is highlighted and the other values are suppressed significantly. As discussed in Section II, [7], [8], and [53] hypothesized that the classification margin is subjected to a Gaussian distribution. Slightly differently, we assume that the deep features x_i with respect to the ith category is subjected to a Gaussian distribution, i.e., $x_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$. In this paper, we define the proposed $\mathcal{G}$-softmax function as

$$p_i = \frac{\exp\left(\overbrace{x_i}^{\text{activation}} + \overbrace{\lambda\Phi(x_i; \mu_i, \sigma_i)}^{\text{distribution term}}\right)}{\sum_{j=1} \exp(x_j + \lambda\Phi(x_j; \mu_j, \sigma_j))}. \tag{3}$$

where λ is a parameter controlling the width of cdf along the y-axis. We can see that if $\lambda = 0$, (3) becomes the conventional softmax function. Φ is the cdf of a Gaussian distribution, that is

$$\Phi(x_i; \mu_i, \sigma_i) = \frac{1}{2}\text{erf}\left(-\frac{\sqrt{2}(\mu_i - x_i)}{2\,\sigma_i}\right) + \frac{1}{2}$$

where

$$\text{erf}(z) = \frac{1}{\sqrt{\pi}} \int_{-z}^{z} e^{-t^2} dt \tag{4}$$

where μ and σ are the mean and standard deviation, respectively. For simplicity, we denote $\Phi(x_i; \mu_i, \sigma_i)$ as Φ_i in the following paragraphs.

Comparing to the softmax function (2), the proposed $\mathcal{G}$-softmax function takes the feature distribution into account, i.e., the distribution term in (3). This formulation leads to two advantages. First, it enables to approximate a large variety of the distributions with respect to every class on the training samples, whereas the softmax function only learns from the current observing sample. Second, with distribution parameters μ and σ, it is straightforward to quantify intraclass compactness and interclass separability. In other words, the proposed $\mathcal{G}$-softmax function is more analytical than the softmax function.

The proposed $\mathcal{G}$-softmax function can work with any ConvNets, such as VGG [47] and ResNet [14]. In this paper, we make $f(x_l) = \Phi(x_l)$, and l is not an arbitrary layer but the fully connected layer. When $x_{l+1} = x_l + \lambda\Phi(x_l)$, x_{l+1} is prone to shift toward the positive axis direction because $\Phi(x_l) \in [0, 1]$. The curve of Φ has a similar shape as that of logistic function and hyperbolic tangent function and can accurately capture the distribution of x. As discussed in Section II, the online learning methods [7], [8], [53] considered the features as a Gaussian distribution and use Kullback–Leibler divergence (KLD) between the estimated distribution and the optimal distribution. Since their formulations involve the unknown optimal Gaussian distribution, they had to apply the Lagrangian to optimize and approximate μ and σ. This may not fit the backpropagation in modern ConvNets which commonly use SGD as a solver.

To optimize μ, we have to compute the partial derivatives of (1) using the chain rule

$$\begin{aligned}\frac{\partial \ell}{\partial \mu_i} &= \frac{\partial}{\partial \Phi_i}\left(-\sum_{i=1} y_i \log \frac{\exp(x_i + \lambda\Phi_i)}{\sum_j \exp(x_j + \lambda\Phi_j)}\right)\frac{\partial \Phi_i}{\partial \mu_i} \\ &= \lambda\left(\left((x_i + \lambda\Phi_i)\sum_j y_j\right) - y_i\right)\frac{\partial \Phi_i}{\partial \mu_i}.\end{aligned} \tag{5}$$

Usually, $\sum_j y_j$ equals to 1 due to the normalization. Similarly, we can obtain the partial derivatives with respect to σ

$$\frac{\partial \ell}{\partial \sigma_i} = \lambda\left(\left((x_i + \lambda\Phi_i)\sum_j y_j\right) - y_i\right)\frac{\partial \Phi_i}{\partial \sigma_i}. \tag{6}$$

According to the cdf, i.e., (3), the derivatives with respect to μ and σ are

$$\frac{\partial \Phi_i}{\partial \mu_i} = -\frac{\sqrt{2}e^{\left(-\frac{(\mu_i - x_i)^2}{2\sigma_i^2}\right)}}{2\sqrt{\pi}\sigma_i} \tag{7}$$

$$\frac{\partial \Phi_i}{\partial \sigma_i} = \frac{\sqrt{2}(\mu_i - x_i)e^{\left(-\frac{(\mu_i - x_i)^2}{2\sigma_i^2}\right)}}{2\sqrt{\pi}\sigma_i^2}. \tag{8}$$

Plugging (7) and (8) into (5) and (6), partial derivatives of μ and σ are

$$\frac{\partial \ell}{\partial \mu_i} = \lambda\left(y_i - (x_i + \lambda\Phi_i)\sum_j y_j\right)\frac{\sqrt{2}e^{\left(-\frac{(\mu_i - x_i)^2}{2\sigma_i^2}\right)}}{2\sqrt{\pi}\sigma_i} \tag{9}$$

$$\begin{aligned}\frac{\partial \ell}{\partial \sigma_i} &= \lambda\left(\left((x_i + \lambda\Phi_i)\sum_j y_j\right) - y_i\right) \\ &\quad\times \frac{\sqrt{2}(\mu_i - x_i)e^{\left(-\frac{(\mu_i - x_i)^2}{2\sigma_i^2}\right)}}{2\sqrt{\pi}\sigma_i^2}.\end{aligned} \tag{10}$$

In the backpropagation of ConvNets, the chain rule requires the derivatives of upper layers to compute the weight derivatives of lower layers. Therefore, $(\partial \ell / \partial x_i)$ is needed to pass backward the lower layers. Because $(\partial \ell / \partial x_i)$ has the same form as $(\partial \ell / \partial \mu_i)$ in (5), we know

$$\frac{\partial \Phi_i}{\partial x_i} = \frac{\sqrt{2}\exp\left(-\frac{(\mu_i - x_i)^2}{2\sigma_i^2}\right)}{2\sqrt{\pi}\sigma_i}. \tag{11}$$

Then, $(\partial \ell / \partial x_i)$ is obtained

$$\frac{\partial \ell}{\partial x_i} = \left(\left((x_i + \lambda\Phi_i)\sum_j y_j\right) - y_i\right) \times \left(1 + \lambda \frac{\sqrt{2}\exp\left(-\frac{(\mu_i - x_i)^2}{2\sigma_i^2}\right)}{2\sqrt{\pi}\sigma_i}\right). \tag{12}$$

B. $\mathcal{G}$-Softmax in Multilabel Classification

Section III-A is based on the single-label classification problems. Here, we apply the proposed $\mathcal{G}$-softmax function to the multilabel classification problem. In the single-label classification problems, the softmax loss and the $\mathcal{G}$-softmax variant are defined as

Softmax:

$$\ell = -\sum_{i=1}^{m} y_i \log\left(\frac{\exp(x_i)}{\sum_{j=1}\exp(x_j)}\right)$$

$\mathcal{G}$-Softmax:

$$\ell = -\sum_{i=1}^{m} y_i \log\left(\frac{\exp(x_i + \lambda\Phi(x_i;\mu_i,\sigma_i))}{\sum_{j=1}\exp(x_j + \lambda\Phi(x_j;\mu_j,\sigma_j))}\right). \tag{13}$$

For multilabel classification, multilabel soft margin loss (MSML) is widely used to solve the multilabel classification problems [9], [69], as defined in the following equation:

$$\ell = -\sum_{i=1}^{m} y_i \log\left(\frac{1}{1+\exp(-x_i)}\right) + (1-y_i)\log\left(1 - \frac{1}{1+\exp(-x_i)}\right). \tag{14}$$

In contrast with MSML, there is a variant that takes x_i^+ and x_i^- as inputs, instead of only taking x_i as inputs in MSML. x_i^+ is the positive feature that is used to compute the probability that the input image is classified to the ith category, while x_i^- is the negative feature that is used to compute the probability that the input image is classified to the non-ith category. The variant is used in the multilabel classification problems [28]. It is defined in the following equation:

$$\ell = -\sum_{i=1}^{m} y_i \log\left(\frac{1}{1+\exp\left(-x_i^+\right)}\right) + (1-y_i)\log\left(\frac{1}{1+\exp\left(-x_i^-\right)}\right). \tag{15}$$

The terms $1/(1+\exp(-x_i))$ and $1 - 1/(1+\exp(-x_i))$ in MSML (14) are both determined by x_i. To make the learning process consistent with the loss function used in single-label classification, we use the variant, i.e., (15), for multilabel classification in this paper and denote it as the softmax loss function for consistency. Correspondingly, the $\mathcal{G}$-softmax loss function is defined as

$$\ell = -\sum_{i=1}^{m} y_i \log\left(\frac{1}{1+\exp\left(-x_i^+ - \lambda\Phi\left(x_i^+;\mu_i^+,\sigma_i^+\right)\right)}\right) + (1-y_i)\log\left(1 - \frac{1}{1+\exp\left(-x_i^- - \lambda\Phi\left(x_i^-;\mu_i^-,\sigma_i^-\right)\right)}\right). \tag{16}$$

In this way, we can model the distributions of $\{x_i^+\}$ and $\{x_i^-\}$ by (μ_i^+, σ_i^+) and (μ_i^-, σ_i^-), respectively.

We can see that the proposed $\mathcal{G}$-softmax and the softmax function are both straightforward to extend for multilabel classification. In contrast, the L-softmax function may not be easy to adapt to multilabel classification. This is because L-softmax function needs to be aware of the feature related to the ground-truth label so that it is able to impose a margin constraint on the feature, that is

$$p = \frac{\exp(\|W_y\|\|x\|\cos(\tilde{m}\theta_y))}{\exp(\|W_y\|\|x\|\cos(\tilde{m}\theta_y)) + \sum_{j\neq y}\exp(\|W_j\|\|x\|\cos(\theta_j))}$$

where $\tilde{m}$ is an integer representing the margin, y indicates the yth label is the ground-truth label of x, W_y is the yth column of W, and θ_y is the angle between W_y and x. When $j \neq y$, the exponential term is the same as in the softmax function. However, when $j = y$, $\tilde{m}$ is used to guarantee the margin between $\|W_y\|\|x\|\cos(\tilde{m}\theta_y)$ and $\|W_j\|\|x\|\cos(\theta_j)$ $(j \neq y)$. As a consequence, it is hard to use in the MSML, because the L-softmax function will treat the terms in (14) differently.

C. Malleable Learning Rates

The training of a model usually required a series of predefined learning rates. The learning rate is a real value and a function of the current epoch with given starting and final value. There are several popular types of learning rates, e.g., linspace, logspace, and staircase. Usually, the number of epochs with these types of learning rates is not more than 300. Although Huang *et al.* [18] use many more epochs with annealing learning rates, the learning rate is designed as a function of iteration number instead of epoch number. Therefore, it may not generalize to distributed or parallel processing, because the iterations are not processed sequentially. We would like to test the proposed $\mathcal{G}$-Softmax function for an extreme condition, i.e., more epochs, to investigate the stability. In the following, we first describe the three learning rates followed by showing how these learning rates are in correlation to the proposed malleable learning rate. The proposed malleable learning rates can control the curvature of the scheduled learning rates to boost the convergence of the learning process.

The linspace learning rates are generated with a simple linear function, where the learning rate at n epoch, $\eta^{(n)}$, is denoted as $\eta^{(n)} = (a + ((b-a)/(M-1))(n-1)) \times \eta^{(0)}$. Here, M is the maximum epoch number, while a and b are the starting and final values of the learning sequences, respectively. $\eta^{(0)}$ is the initial learning rate. Because of linearity, the changes in the learning rates are constant through all epochs. As the learning rates become smaller when an epoch number increases, it is expected that the training process can converge stably. Logspace learning rates meet this requirement by a log function $\eta^{(n)} = \exp(\log(a) + ((\log(b) - \log(a))/(M-1))(n-1)) \times \eta^{(0)}$.

The logspace learning rate has a gradual descent trace that rapidly becomes stable. On the other hand, the staircase learning rate remains constant for a large number of epochs. As the learning rate is not frequently adjusted, the model learning process may not converge. These problems undermine the sustainable convergence ability of deep learning model. Therefore, we integrate the advantages of these learning rates and propose a malleable learning rate, that is

$$\eta^{(n)} = \begin{cases} \exp\left(\log(a_1) + \dfrac{\log(b_1) - \log(a_1)}{M-1}(n-1)\right) \times \eta^{(0)} \\ \quad n \le n_1 \\ \exp\left(\log(a_2) + \dfrac{\log(b_2) - \log(a_2)}{M-1}(n-1)\right) \times \eta^{(0)} \\ \quad n_1 < n \le n_2 \\ \dots \\ \exp\left(\log(a_n) + \dfrac{\log(b_n) - \log(a_n)}{M-1}(n-1)\right) \times \eta^{(0)} \\ \quad n \le N \end{cases} \tag{17}$$

where n_i is the end epoch of the ith piece of learning rates and $a_n = b_{n-1}$. As shown in (17), the propose learning rate is able to separate piecewise learning rates (i.e., staircase learning rates), yet able to control the shape of each piece (e.g., curvature or degree of bend) by configuring a_i and b_i.

For the experiments using pretrained models with the ImageNet data set [44], the initialization contains well-learned knowledge for Tiny ImageNet, MS COCO, and NUS-WIDE, which are similar to ImageNet in terms of visual content and concept labels. Hence, the training process on these data sets does not need a number of epochs [9], [69]. In this paper, we instead apply malleable learning rates on CIFAR to train the models from scratch.

D. Compactness and Separability

As commonly studied in machine learning [35], [59], [66], intraclass compactness and interclass separability are important characteristics that can reveal some intuition about the learning ability and efficacy of a model. Due to the underlying Gaussian nature of the proposed $\mathcal{G}$-softmax function, the intraclass compactness for a given class c is characterized by the respective standard deviation σ_c, where smaller σ_c indicates that the learned model is more compact. Mathematically, the compactness of a given class c can be represented by $(1/\sigma_c)$.

The interclass separability can be measured by computing the disparity of two models, i.e., the divergence between two Gaussian distributions. In the probability and information theory literature, KLD is commonly used to measure the difference between two probability distributions. In the following, we denote a learned Gaussian distribution $\mathcal{N}_i(\mu_i, \sigma_i^2)$ as $\mathcal{N}_i$. Specifically, given two learned Gaussian distributions $\mathcal{N}_i$ and $\mathcal{N}_j$, the divergence between two distributions is

$$\begin{aligned} \mathcal{D}_{KL}(\mathcal{N}_i \| \mathcal{N}_j) &= -\int \phi_i(x) \log(\phi_j(x))dx + \int \phi_i(x)\log(\phi_i)dx \\ &= \log\frac{\sigma_j}{\sigma_i} + \frac{\sigma_i^2 + (\mu_i - \mu_j)^2}{2\sigma_j^2} - \frac{1}{2} \end{aligned} \tag{18}$$

where ϕ_i and ϕ_j are the probability density functions of the respective class. KLD is always nonnegative. As proven by Gibbs' inequality, KLD is zero if and only if the two distributions are equivalent almost everywhere. To quantify the divergence d_i between the distribution of the ith category and the distributions of the rest of categories, we use the mean of KLDs

$$d_i = \frac{1}{2(m-1)} \sum_{j \ne i} (\mathcal{D}_{KL}(\mathcal{N}_i \| \mathcal{N}_j) + \mathcal{D}_{KL}(\mathcal{N}_j \| \mathcal{N}_i)). \tag{19}$$

Because KLD is asymmetric, we compute the mean of $\mathcal{D}_{KL}(\mathcal{N}_i \| \mathcal{N}_j)$ and $\mathcal{D}_{KL}(\mathcal{N}_j \| \mathcal{N}_i)$ for a fair measurement.

Since compactness indicates the intraclass correlations and separability indicates the interclass correlations, we multiply (which is the $\times$ operator) intraclass compactness with interclass separability to overall quantify how discriminative the features with the same label are. Hence, we define separability-σ ratio r with respect to the ith class as follows:

$$r_i = \text{separability} \times \text{compactness} = \frac{d_i}{\sigma_i}. \tag{20}$$

Since σ of a distribution is inversely proportional to compactness, r_i is also inversely proportional to σ. Ideally, we hope a model's r is as large as possible, which requires separability as large as possible and σ as small as possible at the same time.

IV. Empirical Evaluation

In this section, we provide a comprehensive comparison between the softmax function and the proposed $\mathcal{G}$-softmax function for single-label classification and multilabel classification. Specifically, we evaluate three baseline ConvNets (i.e., VGG, DenseNet, and wide ResNet) on the CIFAR-10 and CIFAR-100 data sets for single-label classification. For multilabel classification, we conduct the experiments with ResNet on the MS COCO data set.

A. Data Sets and Evaluation Metrics

To evaluate the proposed $\mathcal{G}$-softmax function for single-label classification, we use the CIFAR-10 [21] and CIFAR-100 data sets, which are widely used in machine learning literature [4], [19], [24], [25], [29], [35], [48], [62]. CIFAR-10 consists of 60 000 color images with 32×32 pixels in 10 classes. Each class has 6000 images, including 5000 training images and 1000 test image. CIFAR-100 has 100 classes and the image resolution is the same as in CIFAR-10. It has 600 images per class, including 500 training images and 100 test images. Moreover, we also use Tiny ImageNet in this paper. It is a variant of ImageNet, which has 200 classes, and each class has 500 training images and 50 validation images.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

For a multilabel classification task, we adopt widely used data sets, i.e., MS COCO [30] and NUS-WIDE [3]. The MS COCO data set is primarily designed for object detection in context, and it is also widely used for multilabel recognition. Therefore, MS COCO is adopted in this paper. It comprises a training set of 82 081 images and a validation set of 40 137 images. The data set covers 80 common object categories, with about 3.5 object labels per image. In this paper, we follow the original split for training and test, respectively. Following [9], [29], [54], and [69], we only use the image labels for training and evaluation. NUS-WIDE consists of 269 648 images with 81 concept labels. We use official train/test split i.e., 161 789 images for training and 107 859 images for evaluation.

We use the same evaluation metrics as in [54] and [69], namely, mean AP (mAP), per-class precision, recall, and F1 score (denoted as C-P, C-R, and C-F1), and overall precision, recall, and F1 score (denoted as O-P, O-R, and O-F1). More concretely, AP is defined as follows:

$$AP_i = \frac{\sum_{k=1}^{R} \hat{P}_i(k)\mathrm{rel}_i(k)}{\sum_{k=1}^{R} \mathrm{rel}_i(k)} \tag{21}$$

where $\mathrm{rel}_i(k)$ is a relevant function that returns 1 if the item at the rank k is relevant to the ith class and returns 0 otherwise. To compute mAP, we collect all predicted probabilities for each class of all the images. The corresponding predicted ith labels over all images are sorted in the descending order. The AP of the ith class is the average of precisions predicted correctly ith labels. $\hat{P}_i(k)$ is the precision ranked at k over all predicted ith labels. R denotes the number of predicted ith labels. Finally, the mAP is obtained by averaging AP over all classes. The other metrics are defined as follows:

$$\begin{aligned}
&\text{C-P} = \frac{1}{C}\sum_i \frac{N_i^c}{N_i^p} \quad \text{O-P} = \frac{\sum_i N_i^c}{\sum_i N_i^p}\\
&\text{C-R} = \frac{1}{C}\sum_i \frac{N_i^c}{N_i^g} \quad \text{O-R} = \frac{\sum_i N_i^c}{\sum_i N_i^g}\\
&\text{C-F1} = 2\,\frac{\text{C-P} \times \text{C-R}}{\text{C-P} + \text{C-R}} \quad \text{O-F1} = 2\,\frac{\text{O-P} \times \text{O-R}}{\text{O-P} + \text{O-R}}
\end{aligned} \tag{22}$$

where N_i^c is the number of images that correctly predicted for the ith class, N_i^p is the number of predicted images for the ith label, and N_i^g is the number of ground-truth images for the ith label. For C-P, C-R, and C-F1, C is the number of labels.

B. Baselines and Experiment Configurations

For the classification task, we adopt softmax and L-softmax [35] as baseline methods for comparison purposes. For multilabel classification, due to the limits of L-softmax as discussed in Section III-B, we only use softmax as the baseline method.

There are a number of ConvNets, such as AlexNet [22], GoogLeNet [50], VGG [47], ResNet [14], wide ResNet [62], and DenseNet [19]. For the experiment on CIFAR-10 and CIFAR-100, we adopt the state-of-the-art wide ResNet and DenseNet as baseline models. In addition, considering that the network structure of wide ResNet and DenseNet is quite different than conventional networks, such as AlexNet and VGG, VGG is taken into account too. Specifically, we use VGG-16 (16-layer model), wide ResNet with 40 convolutional layers and the widening factor of 14, and DenseNet with 100 convolutional layers and the growth rate of 24 in this paper. Our experiments focus on comparing the conventional softmax function with the proposed $\mathcal{G}$-softmax function. The softmax and L-softmax functions are considered as the baseline functions in this paper. For fair comparisons, the experiments are strictly conducted under the same conditions. For all comparisons, we only replace the softmax function in the final layer with the proposed $\mathcal{G}$-softmax function and preserve other parts of the network. In the training stage, we keep most of training hyperparameters, e.g., weight decay, momentum, and so on, the same as in AlexNet [22]. Both the baseline and the proposed $\mathcal{G}$-softmax function would be trained from scratch under the same conditions. In wide ResNet experiments, the batch size for CIFAR-10 and CIFAR-100 are both 128 that is the number used in its original work [62]. In the DenseNet experiments, since its graphics memory usage is considerably higher than wide ResNets, we use 50 as batch size, which leads to fully graphics memory usage for three GPUs. The hardware used in this paper is Intel Xeon E5-2660 CPU and GeForce GTX 1080 Ti. All models are implemented with Torch [5].

We follow the original experimental settings of the baseline models for the training and evaluation of the softmax function and the $\mathcal{G}$-softmax function. For example, in DenseNet, Huang *et al.* [19] train their model in 300 epochs with staircase learning rates. From 1st epoch to 149th epoch, the learning rate is set to 0.1. From 150th epoch to 224th epoch, it is 0.01, and the learning rates of the remaining epochs are 0.001. The wide ResNet model is trained in 200 epochs [62]. The learning rate is initialized to 0.1, and at 60th, 120th, and 160th, it will decrease to 0.02, 0.004, and 0.0008, respectively. To make it comparable to DenseNet, we extend the epochs from 200 to 300 and decrease the learning rate at the 220th and 260th epochs by multiplying 0.2. To avoid *ad hoc* training of hyperparameter settings, we set the weight decay ϵ and momentum γ to be the same as the default hyperparameters in the baselines [19], [62] (i.e., $\epsilon = 5 \times 10^{-4}$ and $\gamma = 0.9$) for the softmax function and the proposed $\mathcal{G}$-softmax function.

For the experiments on Tiny ImageNet, we adopt wide ResNet [62] with 40 convolutional layers and width 14 as the baseline model. The initial learning rate is 0.001 and weight decay is $1\mathrm{e}^{-4}$. The training process consists of 30 epochs with a batch size of 80 and the learning will be decreased to its one-tenth every 10 epoch. Following [18] and [57], we use the ImageNet pretrained weights as an initialization and the input image will be resized to 224×224 to feed the wide ResNet.

In the experiments with malleable learning rates, 1100 epochs are used in training. There are only two phases throughout the whole training, i.e., $1 \le n \le 1000$ and $1000 < n \le 1100$, where $(a_1, b_1) = (0, -8)$ and $(a_2, b_2) = (-8, -9)$.

Different from the softmax function, the proposed $\mathcal{G}$-softmax function has two learnable parameters (i.e., μ and σ) and one hyperparameter (i.e., λ). Without loss of generality, μ and σ are initialized with standard Gaussian distribution (i.e., to 0 and 1). These two parameters would be learned

TABLE I
TOP 1 ERROR RATE (%) ON CIFAR-10 AND CIFAR-100

	# Epoch	CIFAR10	CIFAR100
DSN [19]	300	3.46	17.18
WRN [63]	200	3.80	18.30
L-softmax [36]	80	5.92	29.53
VGG	300	5.69	25.07
VGG L-softmax	300	7.79	32.89
VGG $\mathcal{G}$-softmax	300	5.54	24.92
DSN	300	3.77	19.25
DSN L-softmax	300	4.84	23.22
DSN $\mathcal{G}$-softmax	300	3.67	18.89
WRN	300	3.49	17.66
WRN L-softmax	300	4.27	20.53
WRN $\mathcal{G}$-softmax	300	3.36	17.41
*WRN	1100	3.18	17.60
*WRN L-softmax	1100	4.11	20.45
*WRN $\mathcal{G}$-softmax	1100	**3.14**	**17.04**

TABLE II
TOP 1 ERROR RATE (%) ON THE VALIDATION SET OF TINY IMAGENET

	Top 1 error (%)
Wide-ResNet [63]	39.63
Wide-ResNet SE [18]	32.90
DenseNet [19]	39.09
IGC-V2 [56]	39.02
PyramidNet Shackdrop [58]	31.15
ResNet-101 (Input size: 64×64)	31.66
ResNet-101 (Input size: 224×224)	18.36
ResNet-101 L-Softmax	17.57
ResNet-101 $\mathcal{G}$-Softmax ($\mu = -0.05, \sigma = 1$)	**16.86**
ResNet-101 $\mathcal{G}$-Softmax ($\mu = 0.05, \sigma = 1$)	16.95
ResNet-101 $\mathcal{G}$-Softmax ($\mu = 0, \sigma = 1$)	17.04
ResNet-101 $\mathcal{G}$-Softmax ($\mu = 0, \sigma = 2$)	17.29
ResNet-101 $\mathcal{G}$-Softmax ($\mu = 0, \sigma = 3$)	16.96

through training by (9) and (10). To determine λ, we follow the similar rule where we start from 1 and try the value between $[0, 1]$. As mentioned in Section III, the $\mathcal{G}$-softmax function would be equivalent to the softmax function if $\lambda = 0$. In our experiments, λ is initialized to 1 for CIFAR-10 and CIFAR-100 experiments with DenseNet. In wide ResNet, λ is initialized as 1 for CIFAR-100 experiments and 0.5 for CIFAR-10 experiments.

For the experiments on MS COCO, we refer to the state-of-the-art works [9], [14] to set the weight decay and momentum to $1e^{-4}$ and 0.9, respectively. The model would be trained with the learning rate $1e^{-5}$ in 8 epochs on the MS COCO validation set. In the experiments, we experiment with various initializations of μ and σ to observe how these factors influence the learning. λ is initialized as 1. Since we follow the convention of multilabel classification [9], [69], we use the pretrained weights to initialize the ConvNet and this is different from the initializations in the experiments on CIFAR-10 and CIFAR-100. This difference enables the model to determine μ and σ in a data-driven way, that is, empirically computing the μ and σ from data with the pretrained weights. The image size used in this paper is the same as the one used in [9], i.e., 448×448, while the minibatch size is 16, which is limited by the number of the GPUs.

For the experiments on NUS-WIDE, we use the same experimental setting as the one on MS COCO.

C. Notations

We denote a model with the $\mathcal{G}$-softmax function as *model $\mathcal{G}$-softmax*, e.g., ResNet-101 $\mathcal{G}$-softmax. To simplify notations, we omit softmax following the model name because we assume that the models work with the softmax function by default. For example, ResNet implies that the ResNet model works with the softmax function. In Table I and Figs. 3–7, RSN, DSN, and WRN stand for ResNet, DenseNet, and wide ResNet, respectively.

D. Evaluations on CIFAR

The performances of the softmax function and the $\mathcal{G}$-softmax function are listed in Table I in terms of top 1 error rate. For the convenient purpose, DenseNet and wide ResNet are denoted as DSN and WRN, respectively. The proposed $\mathcal{G}$-softmax function outperforms the softmax and L-softmax functions over all evaluated scenarios.

On CIFAR-10, VGG with the $\mathcal{G}$-softmax function achieves a 5.54% error rate, while the error rates of the softmax and L-softmax functions are 5.69% and 7.79%, respectively. Consistently, VGG with the $\mathcal{G}$-softmax function achieves the similar improvement on CIFAR-100. DenseNet reports their best error rate on CIFAR-10 and CIFAR-100 with 190 convolutional layers and 40 growth rate (denoted as DSN-BC-190-40) [19]. However, DenseNet with this configuration consumes huge graphics memory due to the large depth number, which would occupy about 30 GB of graphics memory to process a batch of 10 images on 3 GPUs. Therefore, we adopt a moderate setting, i.e., DSN-100-24, in our experiments to process as large batch size as possible, i.e., 50 on CIFAR-10 and 32 on CIFAR-100. Under this configuration, the $\mathcal{G}$-softmax function achieves a 3.70% error rate, which is better than the error rate 3.77% of the softmax function and the error rate 4.84% of the L-softmax function, on CIFAR-10. In addition, the error rate of the $\mathcal{G}$-softmax function is decreased to 18.89% compared with the error rate 19.25% of the softmax function and the error rate of 23.22% of the L-softmax function on CIFAR-100. In wide ResNet experiments, the baseline consistently achieves better performances than the baseline of DenseNet on both CIFAR-10 and CIFAR-100, where the $\mathcal{G}$-softmax function further improves the performances to achieve error rate 3.36% on CIFAR-10 and 17.41% on CIFAR-100. As shown in Table I, although the structures of the three model are distinct to each other, the $\mathcal{G}$-softmax function generalizes to these models and improves the respective performances. Applying malleable learning rates with wide ResNet $\mathcal{G}$-softmax can further improve the performances, i.e., 3.14% on CIFAR-10 and 17.04% on CIFAR-100.

E. Evaluations on Tiny ImageNet

Table II reports the error rates of softmax, L-softmax, and the proposed $\mathcal{G}$-softmax function on Tiny ImageNet. We present the error rates of ResNet with input image size 64×64 and 224×224, where 224×224 is used in the setting of training on ImageNet and the training of the initialized ResNet fed with this image size leads to a lower error rate of 18.36%.

TABLE III
PERFORMANCES ON THE VALIDATION SET OF MS COCO

	C-P	C-R	C-F1	O-P	O-R	O-F1	mAP
VGG MCE [29]	-	-	-	-	-	-	70.2
Weak sup (GMP) [41]	-	-	-	-	-	-	62.8
CNN-RNN [53]	66.0	55.6	60.4	69.2	66.4	67.8	-
RGNN [68]	-	-	-	-	-	-	73.0
WELDON [10]	-	-	-	-	-	-	68.8
Multi-CNN [66]	54.8	51.4	53.1	56.7	58.6	57.6	60.4
CNN+LSTM [66]	62.1	51.2	56.1	68.1	56.6	61.8	61.8
MCG-CNN+LSTM [66]	64.2	53.1	58.1	61.3	59.3	61.3	64.4
RLSD [66]	67.6	57.2	62.0	70.1	63.4	66.5	68.2
Pairwise ranking [28]	73.5	56.4	-	76.3	61.8	-	-
MIML-FCN [59]	-	-	-	-	-	-	66.2
RDAR [55]	79.1	58.7	67.4	84.0	63.0	72.0	72.2
ResNet-101 (GAP, ImgSize: 224×224, bz=96, lr=1e-3) [70]	80.8	63.4	69.5	82.2	68.0	74.4	75.2
SRN [70]	81.6	65.4	71.2	82.7	69.9	75.8	77.1
ResNet-101 (GAP, ImgSize: 448×448, unknown bz and lr) [9]	-	-	-	-	-	-	72.5
WILDCAT [9]	-	-	-	-	-	-	80.7
ResNet-101 (GMP, ImgSize: 448×448, bz=16, lr=1e-5)	81.3	70.2	74.1	81.9	74.3	77.9	80.6
ResNet-101 $\mathcal{G}$-softmax w/ ($\mu = 0$, $\sigma = 1$)	82.4	69.3	74.1	83.2	73.3	77.9	80.8
ResNet-101 $\mathcal{G}$-softmax w/ ($\mu = 0$, $\sigma = 0.5$)	82.7	69.3	74.0	83.5	72.9	77.8	81.1
ResNet-101 $\mathcal{G}$-softmax w/ ($\mu = 0$, $\sigma = 5$)	80.6	71.0	74.4	81.3	**74.7**	77.8	80.9
ResNet-101 $\mathcal{G}$-softmax w/ ($\mu = -0.1$, $\sigma = 1$)	81.3	**74.5**	**74.5**	83.4	73.8	**78.3**	**81.3**
ResNet-101 $\mathcal{G}$-softmax w/ ($\mu = 0.1$, $\sigma = 1$)	**83.2**	68.6	73.7	**84.3**	72.3	77.9	81.1

TABLE IV
PERFORMANCES ON THE VALIDATION SET OF NUS-WIDE

	C-P	C-R	C-F1	O-P	O-R	O-F1	mAP
LSEP [28]	**66.7**	45.9	54.4	**76.8**	65.7	70.8	-
Order-free RNN [2]	59.4	50.7	54.7	69.0	71.4	70.2	-
ResNet-101 [70]	65.8	51.9	55.7	75.9	69.5	72.5	59.8
ResNet-101	62.0	56.3	56.9	74.7	71.4	73.0	59.9
ResNet-101 $\mathcal{G}$-softmax ($\mu = 0$, $\sigma = 1$)	62.5	**56.5**	**57.8**	74.7	**71.7**	**73.2**	60.3
ResNet-101 $\mathcal{G}$-softmax ($\mu = 0$, $\sigma = 2$)	62.3	56.0	57.2	74.9	71.3	73.0	60.3
ResNet-101 $\mathcal{G}$-softmax ($\mu = 0$, $\sigma = 3$)	62.9	55.9	57.1	74.9	71.2	73.0	60.1
ResNet-101 $\mathcal{G}$-softmax ($\mu = -0.05$, $\sigma = 1$)	63.2	55.8	57.1	74.9	71.3	73.1	60.0
ResNet-101 $\mathcal{G}$-softmax ($\mu = 0.05$, $\sigma = 1$)	62.0	55.9	57.3	74.9	71.4	73.1	**60.4**

The proposed $\mathcal{G}$-softmax function with various (μ, σ) leads to overall lower error rates than the softmax and L-softmax functions. In particular, $(\mu = -0.05, \sigma = 1)$ achieves the lowest error rate of 16.86%.

F. Evaluations on MS COCO

As shown in Table III, ResNet-101 $\mathcal{G}$-softmax with an initialization of Gaussian distributions $(-0.1, 1)$ for (μ, σ) achieves the best performance over three metrics (i.e., C-F1, O-F1, and mAP). The proposed $\mathcal{G}$-softmax functions are initialized in two straightforward ways. One is to set (μ, σ) to the standard Gaussian distribution parameter (0, 1), while the other one is to empirically compute (μ, σ) from the data. Both approaches achieve better mAPs (80.8% and 81.0%) than the state-of-the-art model [9] (80.7%). To comprehensively understand the effects of μ, σ, we initialize them with other values, i.e., (±0, 0.5), (±0, 5), and (±0.1, 1), By comparing with the performance of ResNet-101 $\mathcal{G}$-softmax with (0, 1), we can see the respective influences of μ, σ. Overall, the four initializations lead to better performances than the initialization of $(0, 1)$ and the initialization of $(-0.1, 1)$ yields the best performance over C-F1, O-F1, and mAP. An observation on μ is that smaller σ leads to higher precision but lower recall. For example, the O-P of $\sigma = 0.5$ is 83.5%, whereas the one of $\sigma = 5$ is 81.3%. Nevertheless, the O-R of $\sigma = 0.5$ is 72.9%, whereas the one of $\sigma = 5$ is 74.7%. According to metrics (22), we can infer that small σ yields less N_i^c and N_i^p than large σ. The change in N_i^c is relatively smaller than the one in N_i^p and these effects of decreasing σ lead to higher precision but lower recall.

G. Evaluations on NUS-WIDE

The experimental results of NUS-WIDE are consistent with the experimental results of MS COCO, as shown in Table IV. The proposed $\mathcal{G}$-softmax function overall outperforms the softmax function over all metrics. Specifically, the setting $(\mu = 0.05, \sigma = 1)$ achieves the best mAP 60.4%.

V. ANALYSIS

In this section, we discuss the influence of the proposed $\mathcal{G}$-softmax function on prediction by presenting a visual comparison with the softmax and the L-softmax function. Then, we further quantify the influences caused by the softmax, L-softmax, and the proposed $\mathcal{G}$-softmax function in terms of intraclass compactness and interclass separability. Moreover, the analysis of the significance of the AP differences between the softmax function and the proposed $\mathcal{G}$-softmax function on MS COCO and NUS-WIDE is provided. Last but not least,

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

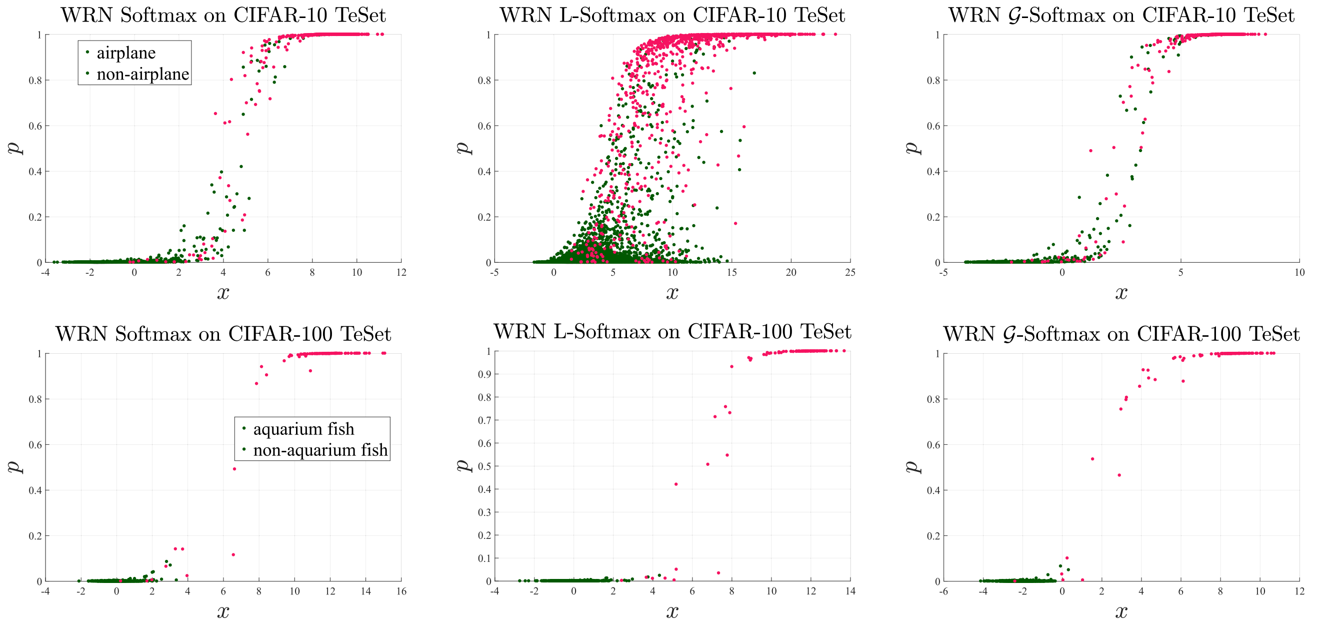


Fig. 3. Prediction versus feature based on all the test images with the ground-truth class “airplane” in CIFAR-10 and “aquarium fish” in CIFAR-100, respectively. Softmax, L-softmax, and the proposed $\mathcal{G}$-softmax function are used with wide ResNet for comparison purposes. The first row consists of the plots of the experiments on CIFAR-10, while the second row consists of the plots of the experiments on CIFAR-100. Given all images with the ground-truth class “airplane,” the corresponding ConvNet would extract the deep features $x \in \mathbb{R}^m$, $m = 10$ in CIFAR-10 and pass them to the predictor for computing the predictions p. Here, the prediction confidence p_1 is corresponding to the ground-truth class, where $p_{i\neq 1}$ are the predictions as other classes. For clarity, we consider all points $(x_i, p_i), i \neq 1$ as “nonairplane” points and plot them in a scatter plot. In this way, the differences in the mapping between the ground-truth class and other impostor classes are visualized. In the CIFAR-100 experiments, the same procedures are underwent, but the ground-truth class is “aquarium fish.” Due to space constraint, we only show the results of the first 10 classes in CIFAR-100.

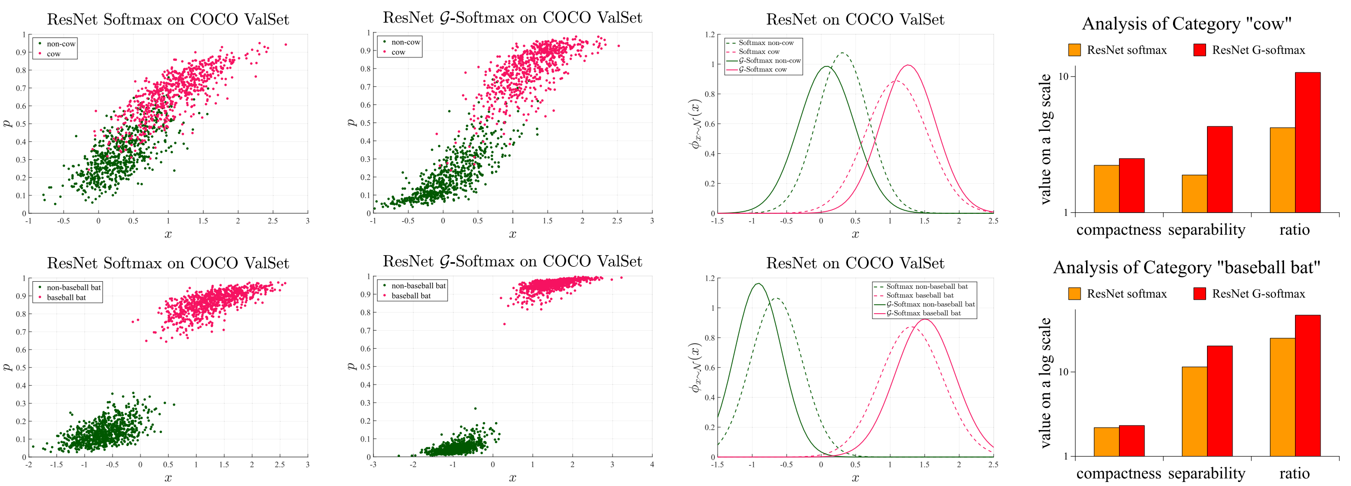


Fig. 4. Feature analysis of the softmax function and the $\mathcal{G}$-softmax function on MSCOCO. The first row is the results of category “cow,” while the second row is the results of category “baseball bat.” For each row, from left to right, the first and second columns are x versus p plots. The third column is the Gaussian distributions of x in the first and second columns. The last column is the corresponding compactness, separability, and ratio. These plots show that the $\mathcal{G}$-softmax function improves both the compactness (the positive curve in the third column becomes taller and narrower) and the separability (the positive curve and the negative curve are farer away).

we analyze the correlations between compactness (separability and ratio) and AP on MS COCO and NUS-WIDE.

A. Influence of the $\mathcal{G}$-Softmax Function on ConvNets

In the literature, there are many works [26], [37], [64] that analyze ConvNets using visualization. In this paper, our hypothesis is related to the distributions of the activations of deep layers. Therefore, we analyze the proposed $\mathcal{G}$-softmax function from the aspect of the mapping between x and p. Given images with a certain label c out of m labels, ConvNets would generate the final feature $x \in \mathbb{R}^m$ preceding to the process of the softmax function. Each x_i in x represents the corresponding confidence for the predicted label i. By the idea of winner-takes-all in the softmax function, the corresponding label i that has the highest value p of the softmax function would be marked as the prediction. We hope that the predicted label is the ground-truth label, i.e., $i = c$, and name $j, j \neq c$ imposter labels. Ideally, the imposter feature x_j is expected to be lower and far away from the ground-truth feature x_i so as to enlarge the probability of correct prediction.

To investigate the influence of the trained $\mathcal{G}$-softmax function on the training set and test set, we inspect the relationship

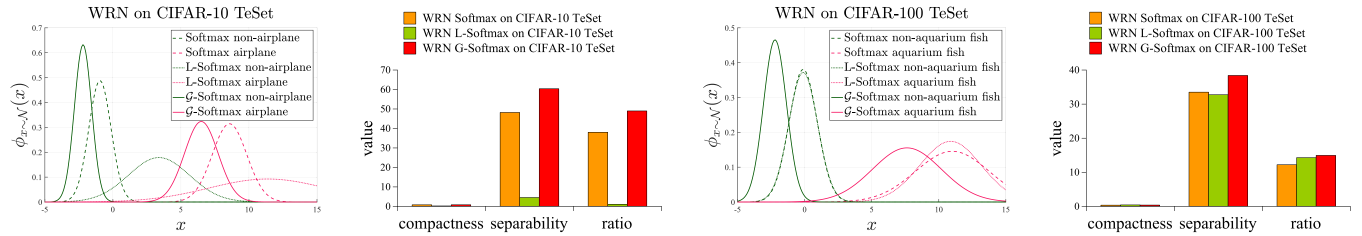


Fig. 5. Gaussian distributions of x and corresponding compactness, separability, and ratio on the test set of Fig. 3. In the CIFAR-10 experiments, given all the testing images with respect to ground-truth class "airplane," given based on x_1, we compute the empirical μ_1 and σ_1 so that the compactness, separability, and ratio can be computed. A similar procedure is conducted in the CIFAR-100 experiments. It can be seen that the ratios of the proposed $\mathcal{G}$-softmax function are overall better than the ones of softmax and L-softmax functions.

between features x and predictions p on CIFAR-10 and CIFAR-100, as shown in Fig. 3. To remove unnecessary interference from the patterns of other classes, we fix the prediction of a subset of the training set and the test set of CIFAR-10 from a single class. For example, given all images with the ground-truth class label "airplane," the ConvNet would generate the deep features $x \in \mathbb{R}^m$, $m = 10$ in CIFAR-10, and pass them to the predictor for computing the predictions p. Note that, here, we denote x_1 as the feature of the class "airplane" and all $x_j, (j \neq 1)$ are considered the features with respect to "nonairplane." Similarly, we also plot the scattered points with respect to the images with label "aquarium fish" on CIFAR-100.

As shown in Fig. 3, the range of x of the proposed $\mathcal{G}$-softmax function is different from the range of x of the softmax and L-softmax functions. Most of the imposter features x_j of the proposed $\mathcal{G}$-softmax function are distributed in the range $[-5, 0]$, whereas x_j of the softmax and L-softmax functions spreads out. In the test set of CIFAR-10, the range of x_c of the proposed $\mathcal{G}$-softmax function approximately spans from 0 to 9, whereas the range of the softmax function is $[0, 11]$ and the range of the L-softmax function is $[0, 24]$. In the test set of CIFAR-100, the range of x_c of the proposed $\mathcal{G}$-softmax function approximately spans from 0 to 11, whereas the range of the softmax function is $[0, 15]$ and the range of the L-softmax function is $[0, 14]$.

Fig. 4 with respect to two categories on MS COCO shows a consistent pattern. In category "cow" and "baseball bat," the positive features of ResNet-101 $\mathcal{G}$-softmax, i.e., the features related to "cow" and "baseball bat," are closer to each other than the ones of ResNet-101 with the softmax function.

To quantitatively understand the distributions of the scattered points in Fig. 3, we empirically compute μ and σ of the points with respect to the softmax function, the L-softmax function, and the proposed $\mathcal{G}$-softmax function. With these distribution parameters, we further compute the compactness, separability, and ratio, as shown in Fig. 5.

The proposed $\mathcal{G}$-softmax function influences the kurtosis of the Gaussian distributions of x of class "airplane" (CIFAR-10) or "aquarium fish" (CIFAR-100) compared with the softmax function. In other words, the curves of the distributions with respect to the proposed $\mathcal{G}$-softmax function are narrower and taller than the ones with respect to the softmax function on both CIFAR-10 and CIFAR-100. In particular, the distributions with respect to the L-softmax function yields a flatter and wider curves than the softmax function and the proposed $\mathcal{G}$-softmax function on CIFAR-10 and CIFAR-100. With the distribution parameters, the intraclass compactness, interclass separability, and separability-σ ratio can be computed and visualized in the bar plots in Fig. 5. Overall, the proposed $\mathcal{G}$-softmax function achieves better intraclass compactness, interclass separability, and separability-σ ratio than the softmax function and the L-softmax function on both CIFAR-10 and CIFAR-100.

Fig. 6 shows a more comprehensive analysis of intraclass compactness, interclass separability, and separability-σ ratio for each class on CIFAR-10. We can see that the proposed $\mathcal{G}$-softmax function improves intraclass compactness, interclass separability, and separability-σ ratio in most of the classes over the softmax function and the L-softmax function. Due to the limitation of space, we do the similar analysis on the first 10 classes on CIFAR-100, as shown in Fig. 7. In contrast to Fig. 6, where the L-softmax function yields the lowest intraclass compactness, interclass separability, and separability-σ ratio on both the training and test set of CIFAR-10, the L-softmax function yields the highest intraclass compactness, interclass separability, and separability-σ ratio in most of the classes on the training set but still yields the lowest intraclass compactness, interclass separability, and separability-σ ratio in most of the classes on the test set. This implies that it may overfit the training data. Again, the proposed $\mathcal{G}$-softmax function consistently yields better intraclass compactness, interclass separability, and separability-σ ratio in most of the classes.

We also analyze intraclass compactness, interclass separability, and separability-σ ratio for multilabel classification on MS COCO. The experimental results of MS COCO show a consistent pattern with those of CIFAR. For example, the x versus p plots of category "baseball bat" in Fig. 4 show that x of ResNet $\mathcal{G}$-softmax is more compact than that of ResNet. Consistently, the Gaussian distribution of ResNet $\mathcal{G}$-softmax with respect to the positive x is taller and narrower than that of ResNet. The compactness of ResNet with respect to class "baseball bat" is 2.1, while the compactness of ResNet $\mathcal{G}$-softmax is 2.3. Fig. 8 shows the average compactnesses of ResNet and ResNet $\mathcal{G}$-softmax over all 80 categories on the MS COCO validation set. The average compactness of ResNet is 2.6, while the average compactness of ResNet $\mathcal{G}$-softmax is 2.8. The separability of the proposed $\mathcal{G}$-softmax function between categories "noncow" and "cow" is 4.3, which is significantly greater than 1.8 (i.e., the separability of the softmax function). The average separability over all

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

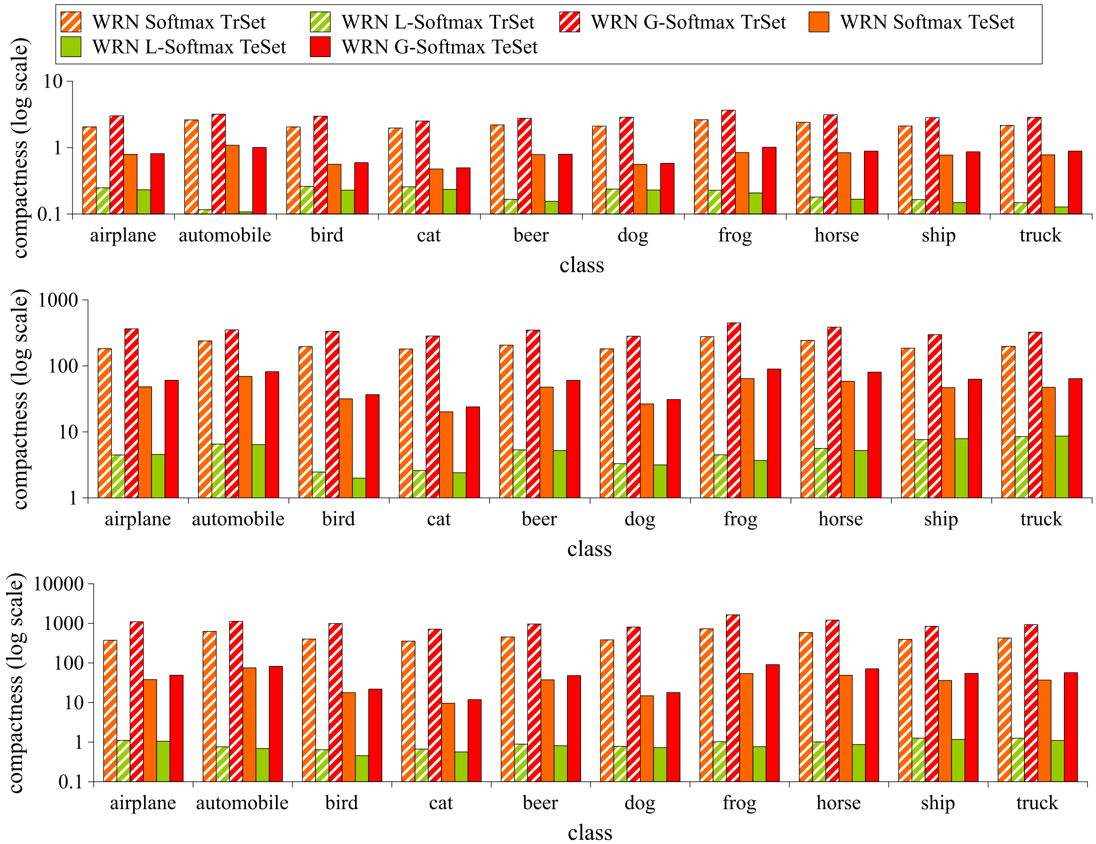


Fig. 6. Analysis on CIFAR-10 test set in terms of compactness, separability, and separability-σ ratio over each class with wide ResNet. We can see that the proposed $\mathcal{G}$-softmax function improves compactness, separability, and ratio on both training and test sets in most categories. As discussed in Section III, the compactness is defined as the reciprocal of σ.

80 categories on MS COCO is shown in Fig. 8. The average separability (4.5) of the proposed $\mathcal{G}$-softmax function is greater than the average separability (4.2) of the softmax function. Similar to intraclass compactness and interclass separability, the average ratio of the proposed $\mathcal{G}$-softmax function is higher than that of the softmax function.

B. Significance of Difference Between Softmax and $\mathcal{G}$-Softmax

As aforementioned discussion about the influence of the proposed $\mathcal{G}$-softmax function, we further quantify the difference of prediction performance caused by the influence. Specifically, we study the difference of AP between the softmax function and the proposed $\mathcal{G}$-softmax function on MS COCO and NUS-WIDE, which are richer in visual content and visual semantics than CIFAR and Tiny ImageNet. First, APs of the softmax function and the proposed $\mathcal{G}$-softmax function with respect to each class are computed. Particularly, the proposed $\mathcal{G}$-softmax functions with each pair of μ and σ in Tables III and IV are used for analysis. With APs of the softmax function and APs of the proposed $\mathcal{G}$-softmax function with a specific μ and σ, the paired sample t-test will be used to compute p value denoted as p-val, which indicates the probability, assuming that the null hypothesis was true. When p-val ≤ 0.05, this implies that the pair of two series of APs are significantly different. Table V shows such an analysis on MS COCO and NUS-WIDE. We can see that p-val of the softmax function and the proposed $\mathcal{G}$-softmax function with $\mu = 0$ and $\sigma = 0.5$ is less than 0.05 in the experiments on MS COCO. This implies that the resulting APs of the proposed $\mathcal{G}$-softmax function are significantly different than those of the softmax function. In contrast, in the experiments on NUS-WIDE, the proposed $\mathcal{G}$-softmax functions in Table IV are significantly different from the softmax function in terms of APs other than the proposed $\mathcal{G}$-softmax function with $\mu = -0.05$ and $\sigma = 1$.

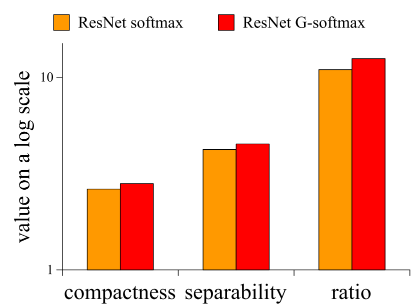


Fig. 8. Average compactness, separability, and ratio over all 80 categories on the MS COCO validation set. The $\mathcal{G}$-softmax function gives rise to the improvements on all metrics.

C. Correlations Between Compactness/Separability/ Ratio and APs

In this paper, we study intraclass compactness and interclass separability for each class in the data sets. A question comes

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

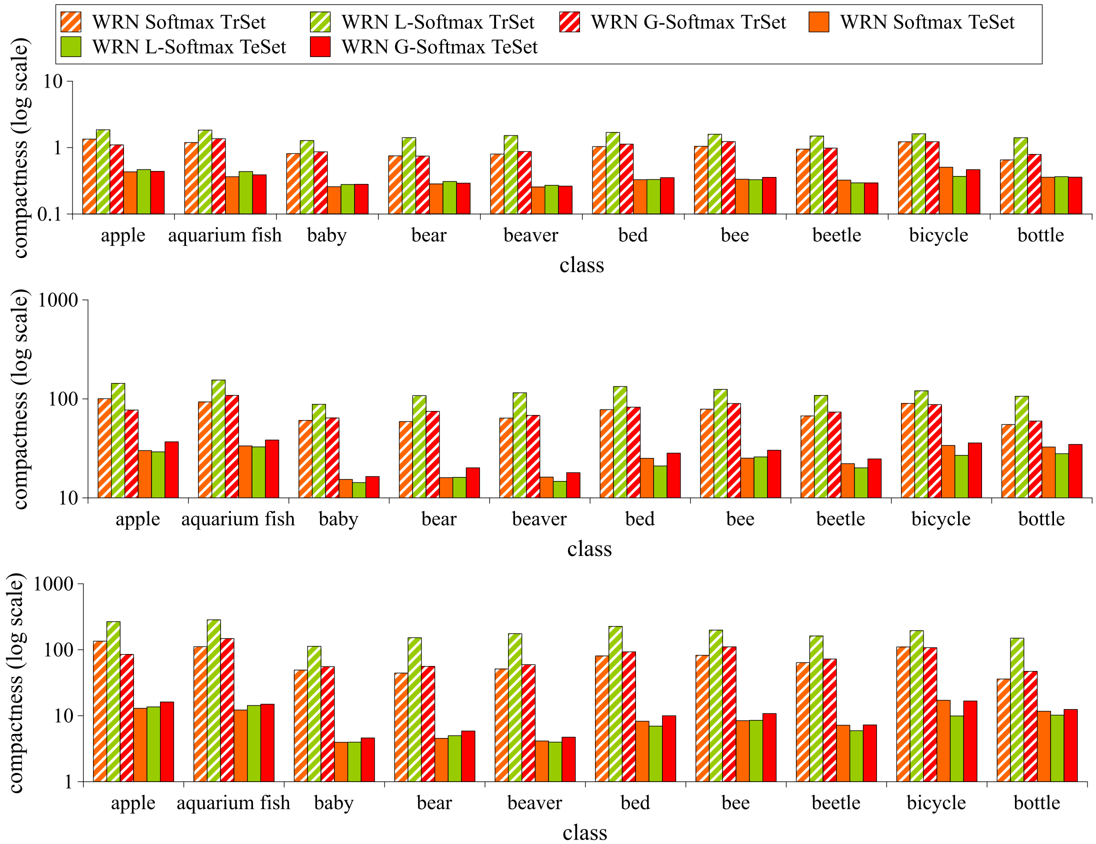


Fig. 7. Analysis on CIFAR-100 test set in terms of compactness, separability, and separability-σ ratio with wide ResNet. For clarity, we present the analyses on the first 10 classes. Although L-softmax achieves better scores over compactness, separability, and ratio on the training set, it has much lower scores on the test set. This implies that it overfits the training set. In contrast, we can see that the proposed $\mathcal{G}$-softmax function improves compactness, separability, and ratio on both training and test sets in most categories.

TABLE V

ANALYSIS OF THE SIGNIFICANCE OF THE PREDICTION DIFFERENCES BETWEEN THE SOFTMAX FUNCTION AND VARIOUS $\mathcal{G}$-SOFTMAX FUNCTIONS IN TABLES III AND IV. THE SIGNIFICANCE OF APS WITH RESPECT TO THE SOFTMAX FUNCTION AND THE PROPOSED $\mathcal{G}$-SOFTMAX FUNCTION IS COMPUTED BY THE PAIRED SAMPLE T-TEST. THE RESULTING p-VAL $\in [0, 1]$ REPORTED IN THE TABLE IS THE PROBABILITY, ASSUMING THAT THE NULL HYPOTHESIS WAS TRUE. IF p-VAL IS EQUAL TO OR LESS THAN 0.05, IT IMPLIES THAT THERE IS A SIGNIFICANT DIFFERENCE BETWEEN THE SOFTMAX FUNCTION AND THE PROPOSED $\mathcal{G}$-SOFTMAX FUNCTION IN COMPACTNESS (SEPARABILITY OR RATIO). IN THE EXPERIMENTS ON MS COCO, THE DIFFERENCE OF APS BETWEEN THE SOFTMAX FUNCTION AND THE PROPOSED $\mathcal{G}$-SOFTMAX FUNCTION WITH $\mu = 0$ AND $\sigma = 0.5$ IS STATISTICALLY SIGNIFICANT (p-VAL < 0.05). IN THE EXPERIMENTS ON NUS-WIDE, BESIDES THE PROPOSED $\mathcal{G}$-SOFTMAX FUNCTION WITH $\mu = -0.05$ AND $\sigma = 1$, THE DIFFERENCES OF APS BETWEEN THE SOFTMAX FUNCTION AND THE PROPOSED $\mathcal{G}$-SOFTMAX FUNCTIONS ARE STATISTICALLY SIGNIFICANT

	MS COCO
$\mathcal{G}$-softmax(0,1)	0.9060
$\mathcal{G}$-softmax(0,0.5)	**0.0359**
$\mathcal{G}$-softmax(0,5)	0.0548
$\mathcal{G}$-softmax(-0.1,1)	0.0764
$\mathcal{G}$-softmax(0.1,1)	0.3160

	NUS-WIDE
$\mathcal{G}$-softmax(0,1)	0.0049
$\mathcal{G}$-softmax(0,2)	**0.0001**
$\mathcal{G}$-softmax(0,3)	0.0098
$\mathcal{G}$-softmax(-0.05,1)	0.6773
$\mathcal{G}$-softmax(0.05,1)	0.0131

up, that is, how are intraclass compactness and interclass separability correlated with APs in the proposed $\mathcal{G}$-softmax function? Note that intraclass compactness and interclass separability may not be influential when the values of them are low. Hence, we only inspect the classes with the best average intraclass compactness, interclass separability, or separability-σ ratio across various $\mathcal{G}$-softmax functions. On the one hand, we have intraclass compactnesses (interclass separabilities or separability-σ ratios) of these classes with respect to each of $\mathcal{G}$-softmax functions in Tables III and IV. On the other hand, we have the APs yielded by each $\mathcal{G}$-softmax functions in Tables III and IV. With the compactness/separabilities/ratios and the corresponding APs of a certain class yielded by various $\mathcal{G}$-softmax functions, we use the Pearson correlation method to quantify the correlation between the three factors and AP and report the Pearson correlation coefficients and the corresponding p-values in Table VI. We can observe that overall intraclass compactness, interclass separability, or separability-σ ratio are linearly correlated with AP to a significance level of 0.05. This implies that the improvement of intraclass compactness and interclass separability will lead to the improvement of AP.

VI. CONCLUSION

In this paper, we propose a Gaussian-based softmax function, namely, $\mathcal{G}$-softmax, which uses cumulative probability function to improve features' intraclass compactness and interclass separability. The proposed $\mathcal{G}$-softmax function is

TABLE VI
CORRELATIONS BETWEEN COMPACTNESS (SEPARABILITY AND RATIO) AND AP ACROSS VARIOUS PROPOSED $\mathcal{G}$-SOFTMAX FUNCTIONS IN TABLES III AND IV ON MS COCO AND NUS-WIDE. AS EACH CLASS HAS ITS OWN UNDERLYING DISTRIBUTION, WE FIRST FIND THE CLASS WITH THE BEST AVERAGE COMPACTNESS, SEPARABILITY, OR RATIO. THEN, THE COMPACTNESSES (SEPARABILITY OR RATIO) OF THIS CLASS ACROSS VARIOUS $\mathcal{G}$-SOFTMAX FUNCTIONS IN TABLES III AND IV ARE USED TO COMPUTE THE PEARSON CORRELATION WITH THE CORRESPONDING APS OF VARIOUS $\mathcal{G}$-SOFTMAX FUNCTIONS. THE PEARSON CORRELATION COEFFICIENT AND THE CORRESPONDING VALUE ARE REPORTED AS (ρ, p-VAL) IN THE TABLE. ρ IS IN $[-1, 1]$. WHEN $\rho = 1$, IT INDICATES THAT COMPACTNESS (SEPARABILITY OR RATIO) IS PERFECTLY LINEARLY CORRELATED WITH AP. WE CAN SEE THAT COMPACTNESS (SEPARABILITY OR RATIO) OF THESE CLASSES IS LINEARLY CORRELATED WITH APS TO A SIGNIFICANCE LEVEL OF 0.05

	MS COCO	NUS-WIDE
Correlation(compactness, AP)	(0.9472, 0.0144)	(0.9635, 0.0083)
Correlation(separability, AP)	(0.9791, 0.0036)	(0.9045, 0.0349)
Correlation(ratio, AP)	(0.9636, 0.0083)	(0.9702, 0.0062)

simple to implement and can easily replace the softmax function. For evaluation purposes, classification data sets (i.e., CIFAR-10, CIFAR-100, and Tiny ImageNet) and multilabel classification data sets (i.e., MS COCO and NUS-WIDE) are used in this paper. The experimental results show that the proposed $\mathcal{G}$-softmax function improves the state-of-the-art ConvNet models. Moreover, in our analysis, it is observed that high intraclass compactness and interclass separability are linearly correlated with AP on MS COCO and NUS-WIDE.

REFERENCES

[1] X. Chang *et al.*, "Semantic pooling for complex event analysis in untrimmed videos," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 8, pp. 1617–1632, Aug. 2017.

[2] S.-F. Chen *et al.*, (2017). "Order-free RNN with visual attention for multi-label classification." [Online]. Available: https://arxiv.org/abs/1707.05495

[3] T.-S. Chua *et al.*, "NUS-WIDE: A real-world Web image database from National University of Singapore," in *Proc. CIVR*, Jul. 2009, p. 48.

[4] D. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (ELUs)," in *Proc. ICLR*, 2016, pp. 1–14.

[5] R. Collobert, K. Kavukcuoglu, and C. Farabet, "Torch7: A MATLAB-like environment for machine learning," in *Proc. BigLearn, NIPS Workshop*, 2011, pp. 1–6.

[6] K. Crammer *et al.*, "Online passive-aggressive algorithms," *J. Mach. Learn. Res.*, vol. 7, pp. 551–585, Dec. 2006.

[7] K. Crammer, M. Dredze, and F. Pereira, "Exact convex confidence-weighted learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2009, pp. 345–352.

[8] M. Dredze, K. Crammer, and F. Pereira, "Confidence-weighted linear classification," in *Proc. ICML*, 2008, pp. 264–271.

[9] T. Durand *et al.*, "WILDCAT: Weakly supervised learning of deep convnets for image classification, pointwise localization and segmentation," in *Proc. CVPR*, Jul. 2017, pp. 5957–5966.

[10] T. Durand, N. Thome, and M. Cord, "WELDON: Weakly supervised learning of deep convolutional neural networks," in *Proc. CVPR*, Jun. 2016, pp. 4743–4752.

[11] K. Fu *et al.*, "Aligning where to see and what to tell: Image captioning with region-based attention and scene-specific contexts," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 12, pp. 2321–2334, Dec. 2017.

[12] R. Girshick *et al.*, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. CVPR*, Jun. 2014, pp. 580–587.

[13] I. J. Goodfellow *et al.*, "Maxout networks," in *Proc. ICML*, 2013, pp. 1319–1327.

[14] K. He *et al.*, "Deep residual learning for image recognition," in *Proc. CVPR*, Jun. 2016, pp. 770–778.

[15] G. E. Hinton *et al.* (2012). "Improving neural networks by preventing co-adaptation of feature detectors." [Online]. Available: https://arxiv.org/abs/1207.0580

[16] W. Hong, J. Yuan, and S. Das Bhattacharjee, "Fried binary embedding for high-dimensional visual features," in *Proc. CVPR*, Jul. 2017, pp. 2749–2757.

[17] W. Hou *et al.*, "Blind image quality assessment via deep learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 6, pp. 1275–1286, Jun. 2015.

[18] G. Huang *et al.*, "Snapshot ensembles: Train 1, get M for free," in *Proc. ICLR*, 2017, pp. 1–14.

[19] G. Huang *et al.*, "Densely connected convolutional networks," in *Proc. CVPR*, Jun. 2017, pp. 4700–4708.

[20] O. Kallenberg, *Foundations of Modern Probability* (Probability Its Applications). New York, NY, USA: Springer-Verlag, 1997.

[21] A. Krizhevsky, "Learning multiple layers of features from tiny images," Univ. Toronton, Toronton, ON, Canada, Tech. Rep. 4, 2009.

[22] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[23] Y. LeCun *et al.*, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[24] C.-Y. Lee, P. W. Gallagher, and Z. Tu, "Generalizing pooling functions in convolutional neural networks: Mixed, gated, and tree," in *Proc. AISTATS*, 2016, pp. 464–472.

[25] C.-Y. Lee *et al.*, "Deeply-supervised nets," in *Proc. AISTATS*, 2015, pp. 562–570.

[26] K. Lenc and A. Vedaldi, "Understanding image representations by measuring their equivariance and equivalence," in *Proc. CVPR*, Jun. 2015, pp. 991–999.

[27] Y. Li, Y. Song, and J. Luo, "Improving pairwise ranking for multi-label image classification," in *Proc. CVPR*, Jul. 2017, pp. 1837–1845.

[28] Z. Li, *et al.*, "Improving multi-label classification using scene cues," *Multimedia Tools Appl.*, vol. 77, no. 5, pp. 6079–6094, 2018.

[29] M. Lin, Q. Chen, and S. Yan, "Network in network," in *Proc. ICLR*, 2014, pp. 1–10.

[30] T.-Y. Lin *et al.*, "Microsoft COCO: Common objects in context," in *Computer Vision* (Lecture Notes in Computer Science), vol. 8693. Zürich, Switzerland: Springer, 2014, pp. 740–755.

[31] W. Liu and I. Tsang, "On the optimality of classifier chain for multi-label classification," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 712–720.

[32] W. Liu and I. Tsang, "Large margin metric learning for multi-label prediction," in *Proc. 29th AAAI Conf. Artif. Intell.*, 2015, pp. 1–7.

[33] W. Liu and I. W. Tsang, "Making decision trees feasible in ultrahigh feature and label dimensions," *J. Mach. Learn. Res.*, vol. 18, no. 81, pp. 1–36, 2017.

[34] W. Liu, I. W. Tsang, and K.-R. Müller, "An easy-to-hard learning paradigm for multiple classes and multiple labels," *J. Mach. Learn. Res.*, vol. 18, no. 1, pp. 1–38, 2017.

[35] W. Liu *et al.*, "Large-margin softmax loss for convolutional neural networks," in *Proc. ICML*, 2016, pp. 507–516.

[36] W. Liu *et al.*, "Metric learning for multi-output tasks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 2, pp. 408–422, Feb. 2019.

[37] A. Mahendran and A. Vedaldi, "Understanding deep image representations by inverting them," in *Proc. CVPR*, Jun. 2015, pp. 5188–5196.

[38] A. Menon, *et al.*, "Characterization of a class of sigmoid functions with applications to neural networks," *Neural Netw.*, vol. 9, no. 5, pp. 819–835, Jul. 1996.

[39] H. Noh, S. Hong, and B. Han, "Learning deconvolution network for semantic segmentation," in *Proc. ICCV*, Dec. 2015, pp. 1520–1528.

[40] M. Oquab *et al.*, "Is object localization for free?—Weakly-supervised learning with convolutional neural networks," in *Proc. CVPR*, Jun. 2015, pp. 685–694.

[41] C. Potthast *et al.*, "Active multi-view object recognition: A unifying view on Online feature selection and view planning," *Robot. Auto. Syst.*, vol. 84, pp. 31–47, Oct. 2016.

[42] J. Read *et al.*, "Classifier chains for multi-label classification," in *Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discovery Databases*, 2009, pp. 254–269.

[43] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain," *Psychol. Rev.*, vol. 65, no. 6, p. 386, 1958.

[44] O. Russakovsky *et al.*, "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015.

[45] L. Shao, D. Wu, and X. Li, "Learning deep and wide: A spectral method for learning deep networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 12, pp. 2303–2308, Dec. 2014.

[46] X. Shen *et al.*, "Multilabel prediction via cross-view search," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 9, pp. 4324–4338, Sep. 2017.

[47] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. ICLR*, 2015, pp. 1–14.

[48] J. T. Springenberg, *et al.*, "Striving for simplicity: The all convolutional net," in *Proc. ICLR*, 2015, pp. 1–14.

[49] N. Srivastava *et al.*, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.

[50] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proc. CVPR*, Jun. 2015, pp. 1–9.

[51] L. Wan *et al.*, "Regularization of neural networks using dropconnect," in *Proc. ICML*, 2013, pp. 1058–1066.

[52] J. Wang *et al.*, "CNN-RNN: A unified framework for multi-label image classification," in *Proc. CVPR*, Jun. 2016, pp. 2285–2294.

[53] J. Wang, P. Zhao, and S. C. H. Hoi, "Exact soft confidence-weighted learning," in *Proc. ICML*, 2012, pp. 121–128.

[54] Z. Wang *et al.*, "Multi-label image recognition by recurrently discovering attentional regions," in *Proc. ICCV*, Oct. 2017, pp. 464–472.

[55] G. Xie *et al.*, "Interleaved structured sparse convolutional neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8847–8856.

[56] H. Xu *et al.*, "End-to-end learning of driving models from large-scale video datasets," in *Proc. CVPR*, Jul. 2017, pp. 2174–2182.

[57] Y. Yamada *et al.*, "Shakedrop regularization," in *Int. Conf. Learn. Represent.*, 2018, pp. 1–11.

[58] H. Yang *et al.*, "MIML-FCN+: Multi-instance multi-label learning via fully convolutional networks with privileged information," in *Proc. CVPR*, Jun. 2017, pp. 5996–6004.

[59] L. Yang *et al.*, "An efficient algorithm for local distance metric learning," in *Proc. AAAI*, 2006, pp. 543–548.

[60] F. Yu, V. Koltun, and T. Funkhouser, "Dilated residual networks," in *Proc. CVPR*, Jul. 2017, pp. 472–480.

[61] Y. Yuan, L. Mou, and X. Lu, "Scene recognition by manifold regularized deep learning architecture," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 10, pp. 2222–2233, Oct. 2015.

[62] S. Zagoruyko and N. Komodakis, "Wide residual networks," in *Proc. BMVC*, 2016, pp. 1–15.

[63] M. D. Zeiler and R. Fergus, "Stochastic pooling for regularization of deep convolutional neural networks," in *Proc. ICLR*, 2013, pp. 1–9.

[64] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Computer Vision* (Lecture Notes in Computer Science), vol. 8689. Zürich, Switzerland: Springer, 2014, pp. 818–833.

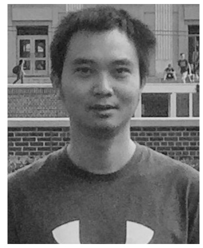
[65] J. Zhang *et al.*, "Multi-label image classification with regional latent semantic dependencies," *IEEE Trans. Multimedia*, vol. 20, no. 10, pp. 2801–2813, Oct. 2018.

[66] W. Zhang *et al.*, "Optimal dimensionality of metric space for classification," in *Proc. ICML*, 2007, pp. 1135–1142.

[67] R. Zhao *et al.*, "Regional gating neural networks for multi-label image classification," in *Proc. BMVC*, 2016, pp. 1–12.

[68] C. Zhou and J. Yuan, "Multi-label learning of part detectors for heavily occluded pedestrian detection," in *Proc. ICCV*, Jun. 2017, pp. 3486–3495.

[69] F. Zhu *et al.*, "Learning spatial regularization with image-level supervisions for multi-label image classification," in *Proc. CVPR*, Jul. 2017, pp. 2027–2036.



Yan Luo received the B.Sc. degree in computer science from Xi'an, China, in 2008. He is currently pursuing the Ph.D. degree with the Department of Computer Science and Engineering, University of Minnesota at Twin Cities, Minneapolis, MN, USA.

In 2013, he joined the Sensor-enhanced Social Media (SeSaMe) Centre, Interactive and Digital Media Institute, National University of Singapore, as a Research Assistant. In 2015, he joined the Visual Information Processing Laboratory at the National University of Singapore as a Ph.D. Student. He worked in the industry for several years on distributed system. His current research interests include computer vision, computational visual cognition, and deep learning.

Yongkang Wong (M'09) received the B.Eng. degree from The University of Adelaide, Adelaide, SA, Australia, and the Ph.D. degree from The University of Queensland, Brisbane, QLD, Australia.

He was a Graduate Researcher with the NICTA's Queensland Laboratory, Brisbane, from 2008 to 2012. He is a Senior Research Fellow of the School of Computing, National University of Singapore (NUS), Singapore. He is also the Assistant Director of the NUS Centre for Research in Privacy Technologies. His current research interests include the areas of image/video processing, machine learning, and social scene analysis.

Mohan Kankanhalli (F'14) received the B.Tech. degree from the IIT Kharagpur, Kharagpur, India, and the M.S. and Ph.D. degrees from the Rensselaer Polytechnic Institute, Troy, NY, USA.

He is the Provost's Chair Professor with the Department of Computer Science, National University of Singapore (NUS), Singapore. He is also the Director of the NUS Centre for Research in Privacy Technologies and also the Dean of the School of Computing, NUS. His current research interests include multimedia computing, multimedia security, image/video processing, and social media analysis.

Dr. Kankanhalli is active in the multimedia research community, and is on the editorial boards of several journals.

Qi Zhao (M'04) received the Ph.D. degree in computer engineering from the University of California at Santa Cruz, Santa Cruz, CA, USA, in 2009.

She was an Assistant Professor with the Department of Electrical and Computer Engineering and the Department of Ophthalmology, National University of Singapore, Singapore. She was a Post-Doctoral Researcher with the Computation and Neural Systems, Division of Biology, California Institute of Technology, Pasadena, CA, USA, from 2009 to 2011. She is currently an Assistant Professor with the Department of Computer Science and Engineering, University of Minnesota at Twin Cities, Minneapolis, MN, USA. She has published more than 50 journal and conference papers in top computer vision, machine learning, and cognitive neuroscience venues and edited a book *Computational and Cognitive Neuroscience of Vision* (Springer), which provides a systematic and comprehensive overview of vision from various perspectives, ranging from neuroscience to cognition, and from computational principles to engineering developments. Her current research interests include computer vision, machine learning, cognitive neuroscience, and mental disorders.