
Learning towards Minimum Hyperspherical Energy

Weiyang Liu^{1,*}, Rongmei Lin^{2,*}, Zhen Liu^{1,*}, Lixin Liu¹, Zhidong Yu³, Bo Dai¹, Le Song¹

¹Georgia Institute of Technology ²Emory University ³NVIDIA

{wyliu, liuzhen1994}@gatech.edu, rongmei.lin@emory.edu, lsong@cc.gatech.edu

Abstract

Neural networks are a powerful class of nonlinear functions that can be trained end-to-end on various applications. While the over-parametrization nature in many neural networks renders the ability to fit complex functions and the strong representation power to handle challenging tasks, it also leads to highly correlated neurons that can hurt the generalization ability and incur unnecessary computation cost. As a result, how to regularize the network to avoid undesired representation redundancy becomes an important issue. To this end, we draw inspiration from a well-known problem in physics – Thomson problem, where one seeks to find a state that distributes N electrons on a unit sphere as even as possible with minimum potential energy. In light of this intuition, we reduce the redundancy regularization problem to generic energy minimization, and propose a minimum hyperspherical energy (MHE) objective as generic regularization for neural networks. We also propose a few novel variants of MHE, and provide some insights from a theoretical point of view. Finally, we apply networks with MHE regularization to several challenging tasks. Extensive experiments demonstrate the effectiveness of our method, by showing the superior performance with MHE regularization.

1 Introduction

The recent success of deep neural networks has led to its wide applications in a variety of tasks. With the over-parametrization nature and deep layered architecture, current deep networks [13, 44, 40] are able to achieve impressive performance on large-scale problems. Despite such success, having redundant and highly correlated neurons (*e.g.*, weights of kernels/filters in convolutional neural networks (CNNs)) caused by over-parametrization presents an issue [35, 39], which motivated a series of influential works in network compression [9, 1] and parameter-efficient network architectures [15, 17, 60]. These works either compress the network by pruning redundant neurons or directly modify the network architecture, aiming to achieve comparable performance while using fewer parameters. Yet, it remains an open problem to find a unified and principled theory that guides the network compression in the context of optimal generalization ability.

Another stream of works seek to further release the network generalization power by alleviating redundancy through diversification [55, 54, 4, 34] as rigorously analyzed by [57]. Most of these works address the redundancy problem by enforcing relatively large diversity between pairwise projection bases via regularization. Our work broadly falls into this category by sharing similar high-level target, but the spirit and motivation behind our proposed models are distinct. In particular, there is a recent trend of studies that feature the significance of angular learning at both loss and convolution levels [27, 26, 28, 25], based on the observation that the angles in deep embeddings learned by CNNs tend to encode semantic difference. The key intuition is that angles preserve the most abundant and discriminative information for visual recognition. As a result, hyperspherical geodesic distances between neurons naturally plays a key role in this context, and thus, it is intuitively desired to impose discriminativeness by keeping their projections on the hypersphere as far away from

* indicates equal contributions.

each other as possible. While the concept of imposing large angular diversities was also considered in [57, 55, 54, 34], they do not consider diversity in terms of global equidistribution of embeddings on the hypersphere, which fails to achieve the state-of-the-art performances.

Given the above motivation, we draw inspiration from a well-known physics problem, called Thomson problem [46, 41]. The goal of Thomson problem is to determine the minimum electrostatic potential energy configuration of N mutually-repelling electrons on the surface of a unit sphere. We identify the intrinsic resemblance between the Thomson problem and our target, in the sense that diversifying neurons can be seen as searching for an optimal configuration of electron locations. Similarly, we characterize the diversity for a group of neurons by defining a generic hyperspherical potential energy using their pairwise relationship. Higher energy implies higher redundancy, while lower energy indicates that these neurons are more diverse and more uniformly spaced. To reduce the redundancy of neurons and improve the neural networks, we propose a novel *minimum hyperspherical energy* (MHE) regularization framework, where the diversity of neurons is promoted by minimizing the hyperspherical energy in each layer. As verified by comprehensive experiments on multiple tasks, MHE is able to consistently improve the generalization power of neural networks.

MHE faces different situations when it is applied to hidden layers and output layers. For hidden layers, applying MHE straightforwardly may still encourage some degree of redundancy since it will produce co-linear bases pointing to opposite directions (see Fig. 1 middle). In order to avoid such redundancy, we propose the half-space MHE which constructs a group of virtual neurons and minimize the hyperspherical energy of both existing and virtual neurons. For output layers, MHE aims to distribute the classifier neuron¹ as uniformly as possible to improve the inter-class feature separability. Different from MHE in hidden layers, classifier neurons should be distributed in the full space for the best classification performance [27, 26]. An intuitive comparison among the widely used orthonormal regularization, the proposed MHE and half-space MHE is provided in Fig. 1. One can observe that both MHE and half-space MHE are able to uniformly distribute the neurons over the hypersphere and half-space hypersphere, respectively. In contrast, conventional orthonormal regularization tends to group neurons closer, especially when the number of neurons is greater than the dimension.

MHE is originally defined on Euclidean distance, as indicated in Thomson problem. However, we further consider minimizing hyperspherical energy defined with respect to angular distance, which we will refer to as angular-MHE (A-MHE) in the following paper. In addition, we give some theoretical insights of MHE regularization, by discussing the asymptotic behavior and generalization error. Last, we apply MHE regularization to multiple vision tasks, including generic object recognition, class-imbalance learning, and face recognition. In the experiments, we show that MHE is architecture-agnostic and can considerably improve the generalization ability.

2 Related Works

Diversity regularization is shown useful in sparse coding [30, 33], ensemble learning [24, 22], self-paced learning [19], metric learning [56] etc. Early studies in sparse coding [30, 33] show that the generalization ability of codebook can be improved via diversity regularization, where the diversity is often modeled using the (empirical) covariance matrix. More recently, a series of studies have featured diversity regularization in neural networks [57, 55, 54, 4, 34, 53], where regularization is mostly achieved via promoting large angle/orthogonality, or reducing covariance between bases. Our work differs from these studies by formulating the diversity of neurons on the entire hypersphere, therefore promoting diversity from a more global, top-down perspective.

Methods other than diversity-promoting regularization have been widely proposed to improve CNNs [42, 18, 31, 28] and generative adversarial nets (GANs) [3, 32]. MHE can be regarded as a complement that can be applied on top of these methods.

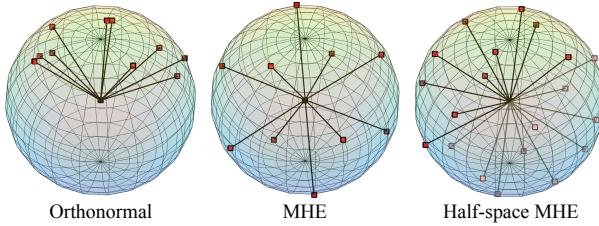


Figure 1: Orthonormal, MHE and half-space MHE regularization. The red dots denote the neurons optimized by the gradient of the corresponding regularization. The rightmost pink dots denote the virtual negative neurons. We randomly initialize the weights of 10 neurons on a 3D Sphere and optimize them with SGD.

¹Classifier neurons are the projection bases of the last layer (*i.e.*, output layer) before input to softmax.

3 Learning Neurons towards Minimum Hyperspherical Energy

3.1 Formulation of Minimum Hyperspherical Energy

Minimum hyperspherical energy defines an equilibrium state of the configuration of neuron's directions. We argue that the power of neural representation of each layer can be characterized by the hyperspherical energy of its neurons, and therefore a minimal energy configuration of neurons can induce better generalization. Before delving into details, we first define the hyperspherical potential energy functional for N neurons with $d+1$ -dimension $\mathbf{W}_N = \{\mathbf{w}_1, \dots, \mathbf{w}_N \in \mathbb{R}^{d+1}\}$ as

$$\mathbf{E}_{s,d}(\hat{\mathbf{w}}_i|_{i=1}^N) = \sum_{i=1}^N \sum_{j=1, j \neq i}^N f_s(\|\hat{\mathbf{w}}_i - \hat{\mathbf{w}}_j\|) = \begin{cases} \sum_{i \neq j} \|\hat{\mathbf{w}}_i - \hat{\mathbf{w}}_j\|^{-s}, & s > 0 \\ \sum_{i \neq j} \log(\|\hat{\mathbf{w}}_i - \hat{\mathbf{w}}_j\|), & s = 0 \end{cases}, \quad (1)$$

where $\|\cdot\|$ denotes Euclidean distance, $f_s(\cdot)$ is a decreasing real-valued function, and $\hat{\mathbf{w}}_i = \frac{\mathbf{w}_i}{\|\mathbf{w}_i\|}$ is the i -th neuron weight projected to the unit hypersphere $\mathbb{S}^d = \{\mathbf{w} \in \mathbb{R}^{d+1} | \|\mathbf{w}\| = 1\}$. We also denote $\hat{\mathbf{W}}_N = \{\hat{\mathbf{w}}_1, \dots, \hat{\mathbf{w}}_N \in \mathbb{S}^d\}$, and $\mathbf{E}_s = \mathbf{E}_{s,d}(\hat{\mathbf{w}}_i|_{i=1}^N)$ for short. There are plenty of choices for $f_s(\cdot)$, but in this paper we use $f_s(z) = z^{-s}$, $s > 0$, known as Riesz s -kernels. Particularly, as $s \rightarrow 0$, $z^{-s} \rightarrow s \log(z^{-1}) + 1$, which is an affine transformation of $\log(z^{-1})$. It follows that optimizing the logarithmic hyperspherical energy $\mathbf{E}_0 = \sum_{i \neq j} \log(\|\hat{\mathbf{w}}_i - \hat{\mathbf{w}}_j\|)$ is essentially the limiting case of optimizing the hyperspherical energy \mathbf{E}_s . We therefore define $f_0(z) = \log(z^{-1})$ for convenience.

The goal of the MHE criterion is to minimize the energy in Eq. (1) by varying the orientations of the neuron weights $\mathbf{w}_1, \dots, \mathbf{w}_N$. To be precise, we solve an optimization problem: $\min_{\mathbf{W}_N} \mathbf{E}_s$ with $s \geq 0$. In particular, when $s=0$, we solve the logarithmic energy minimization problem:

$$\arg \min_{\mathbf{W}_N} \mathbf{E}_0 = \arg \min_{\mathbf{W}_N} \exp(\mathbf{E}_0) = \arg \min_{\mathbf{W}_N} \prod_{i \neq j} \|\hat{\mathbf{w}}_i - \hat{\mathbf{w}}_j\|, \quad (2)$$

in which we maximize the product of Euclidean distances. Note that Thomson problem corresponds to minimizing \mathbf{E}_1 , which is a NP-hard problem. Therefore in practice we can only compute its approximate solution by heuristics. In the case of neural networks, such a differentiable objective can be directly optimized via gradient descent.

3.2 Logarithmic Hyperspherical Energy as a Relaxation

Optimizing the original energy in Eq. (1) is equivalent to optimize its logarithmic form $\log \mathbf{E}_s$. To efficiently solve this difficult optimization problem, we can instead optimize the lower bound of $\log \mathbf{E}_s$ as a surrogate energy, by applying Jensen's inequality:

$$\arg \min_{\hat{\mathbf{W}}_N} \left\{ \mathbf{E}_{\log} := \sum_{i=1}^N \sum_{j=1, j \neq i}^N \log \left(f_s(\|\hat{\mathbf{w}}_i - \hat{\mathbf{w}}_j\|) \right) \right\} \quad (3)$$

With $f_s(z) = z^{-s}$, $s > 0$, we observe that \mathbf{E}_{\log} becomes $s \mathbf{E}_0 = -s \sum_{i \neq j} \log(\|\hat{\mathbf{w}}_i - \hat{\mathbf{w}}_j\|)$, which is identical to the logarithmic hyperspherical energy \mathbf{E}_0 up to a multiplicative factor s . Hence solving the maximization of the product of distances \mathbf{E}_0 can also be viewed as a relaxation for \mathbf{E}_s for $s > 0$.

3.3 MHE as Regularization for Neural Networks

Now that we have introduced the formulation of MHE, we propose MHE regularization for neural networks. In supervised neural network learning, the entire objective function is shown as follows:

$$\mathcal{L} = \underbrace{\frac{1}{m} \sum_{j=1}^m \ell(\langle \mathbf{w}_i^{\text{out}}, \mathbf{x}_j \rangle_{i=1}^c, \mathbf{y}_j)}_{\text{training data fitting}} + \underbrace{\lambda_h \cdot \sum_{j=1}^{L-1} \frac{1}{N_j(N_j-1)} \{\mathbf{E}_s\}_j}_{T_h: \text{hyperspherical energy for hidden layers}} + \underbrace{\lambda_o \cdot \frac{1}{N_L(N_L-1)} \mathbf{E}_s(\hat{\mathbf{w}}_i^{\text{out}}|_{i=1}^c)}_{T_o: \text{hyperspherical energy for output layer}} \quad (4)$$

where \mathbf{x}_i is the feature of the i -th training sample entering the output layer, $\mathbf{w}_i^{\text{out}}$ is the classifier neuron for the i -th class in the output fully-connected layer and $\hat{\mathbf{w}}_i^{\text{out}}$ denotes its normalized version. $\{\mathbf{E}_s\}_i$ denotes the hyperspherical energy for the neurons in the i -th layer. c is the number of classes, m is the batch size, L is the number of layers of the neural network, and N_i is the number of neurons in the i -th layer. $\mathbf{E}_s(\hat{\mathbf{w}}_i^{\text{out}}|_{i=1}^c)$ denotes the hyperspherical energy of neurons $\{\hat{\mathbf{w}}_1^{\text{out}}, \dots, \hat{\mathbf{w}}_c^{\text{out}}\}$. The ℓ_2 weight decay is omitted here for simplicity, but we will use it by default in practice. An alternative interpretation of the MHE regularization from decoupled view is given in Appendix C. The different effect of the MHE regularization in hidden layers and output layers are discussed separately.

MHE for hidden layers. To make neurons in the hidden layers more discriminative and less redundant, we propose to use MHE as a form of regularization. MHE encourages the normalized neurons to

be uniformly distributed on a unit hypersphere, which is partially inspired by the observation in [28] that angular difference in neurons preserves semantic (label-related) information. To some extent, MHE maximizes the average angular difference between neurons (specifically, the hyperspherical energy of neurons in every hidden layer). For instance, in CNNs we minimize the hyperpspherical energy of kernels in convolutional and fully-connected layers except the output layer.

MHE for output layers. For the output layer, we propose to enhance the inter-class feature separability with MHE to learn discriminative and well-separated features. For classification tasks, MHE regularization is complementary to the softmax cross-entropy loss in CNNs. The softmax loss focuses more on the intra-class compactness, while MHE encourages the inter-class separability. Therefore, MHE on output layers can induce features with better generalization power.

3.4 MHE in Half Space

Directly applying the MHE formulation may still encounter some redundancy. An example in Fig. 2, with two neurons in a 2-dimensional space, illustrates this potential issue. Directly imposing the original MHE regularization leads to a solution that two neurons are colinear but with opposite directions. To avoid such redundancy, we propose the half-space MHE regularization which constructs some virtual neurons and minimizes the hyperspherical energy of both original and virtual neurons together. Specifically, half-space MHE constructs a colinear virtual neuron with opposite direction for every existing neuron. Therefore, we end up with minimizing the hyperspherical energy with $2N_i$ neurons in the i -th layer (*i.e.*, minimizing $E_s(\{\hat{w}_k, -\hat{w}_k\}|_{k=1}^{2N_i})$). This half-space variant will encourage the neurons to be less correlated and less redundant, as illustrated in Fig. 5. Note that, Half-space MHE can only be used in hidden layers, because the colinear neurons does not constitute redundancy in output layers, as shown in [27]. Still, colinearity is not very likely in high-dimensional spaces, especially when the neurons are optimized to fit training data. This may be the reason that the original MHE regularization still improves the baselines.

3.5 MHE beyond Euclidean Distance

The hyperspherical energy is originally defined based on the Euclidean distance on a hypersphere, which can be viewed as an angular measure. In addition to Euclidean distance, we further consider the geodesic distance on a unit hypersphere as a distance measure for neurons, which is exactly the same as the angle between neurons. Specifically, we consider to use $\arccos(\hat{w}_i^\top \hat{w}_j)$ to replace $\|\hat{w}_i - \hat{w}_j\|$ in hyperspherical energies. Following this idea, we propose angular MHE (A-MHE) as a simple extension, where the hyperspherical energy is rewritten as:

$$E_{s,d}^a(\hat{w}_i|_{i=1}^N) = \sum_{i=1}^N \sum_{j=1, j \neq i}^N f_s(\arccos(\hat{w}_i^\top \hat{w}_j)) = \begin{cases} \sum_{i \neq j} \arccos(\hat{w}_i^\top \hat{w}_j)^{-s}, & s > 0 \\ \sum_{i \neq j} \log(\arccos(\hat{w}_i^\top \hat{w}_j)), & s = 0 \end{cases} \quad (5)$$

which can be viewed as redefining MHE based on geodesic distance on hyperspheres (*i.e.*, angle), and can be used as an alternative to the original hyperspherical energy E_s in Eq. (4). Note that, A-MHE can also be learned in full-space or half-space, leading to similar variants as original MHE. The key difference between MHE and A-MHE lies in the optimization dynamics, because their gradients w.r.t the neuron weights are quite different. A-MHE is also computationally expensive than MHE.

3.6 Mini-batch Approximation for MHE

With a large number of neurons in one layer, calculating MHE can be computationally expensive as it requires computing the pair-wise distances between neurons. To address this issue, we propose to the mini-batch version of MHE to approximate the MHE (either original or half-space) objective.

Mini-batch approximation for MHE on hidden layers. For hidden layers, mini-batch approximation iteratively takes a random batch of neurons as input and minimizes their hyperspherical energy as an approximation to the MHE. Note that the gradients of the mini-batch objective is an unbiased estimation of the original gradient of MHE.

Data-dependent mini-batch approximation for output layers. For the output layer, the data-dependent mini-batch approximation iteratively takes the classifier neurons corresponding to the classes that exist in mini-batches. It minimizes $\frac{1}{m(N-1)} \sum_{i=1}^m \sum_{j=1, j \neq y_i}^N f_s(\|\hat{w}_{y_i} - \hat{w}_j\|)$ in each iteration, where y_i denotes the class label of the i -th sample in each mini-batch, m is the mini-batch size, and N is the number of neurons (in a particular layer).

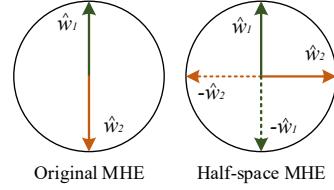


Figure 2: Half-space MHE.

3.7 Discussions

Connections to scientific problems. The hyperspherical energy minimization has close relationships with scientific problems. When $s=1$, Eq. (1) reduces to Thomson problem [46, 41] (in physics) where one needs to determine the minimum electrostatic potential energy configuration of N mutually-repelling electrons on a unit sphere. When $s=\infty$, Eq. (1) becomes Tammes problem [45] (in geometry) where the goal is to pack a given number of circles on the surface of a sphere such that the minimum distance between circles is maximized. When $s=0$, Eq. (1) becomes Whyte’s problem where the goal is to maximize product of Euclidean distances as shown in Eq. (2). Our work aims to make use of important insights from these problems to improve neural networks.

Comparison to orthogonality/angle-promoting regularization. Promoting orthogonality or large angles between bases has been a popular choice for encouraging diversity. Probably the most related and widely used one is the orthonormal regularization which aims to minimize $\|\mathbf{W}^\top \mathbf{W} - \mathbf{I}\|_F$, where \mathbf{W} denotes the weights of a group of neurons with each column being one neuron and \mathbf{I} is an identity matrix. However, these methods models diversity regularization at a more local level, while MHE regularization seeks to model the problem in a more top-down manner.

Weighted MHE? Since we use the normalized neurons for MHE, a natural question may arise: what if we use the original (*i.e.*, unnormalized) neurons to compute MHE? This is essentially to compute a weighted MHE where the norm of the neuron is taken into consideration. Empirically, if we keep the norm of neuron weights in MHE, then the norm will simply keep increasing instead of changing its angular distance with the other neurons in order to minimize the objective. Therefore, applying weighted MHE to neural networks may reduce the recognition accuracy, further validating the importance of the hyperspherical (*i.e.*, angular) diversity, as has been emphasized in [26, 28, 25].

4 Theoretical Insights

Different from a rigorous theoretical analysis, our goal here is to leverage existing theoretical results from [36, 21, 11, 23, 10, 21, 7] and provide theoretical yet intuitive understandings about MHE.

4.1 Asymptotic Behavior

This subsection shows how the hyperspherical energy behaves asymptotically. Specifically, as $N \rightarrow \infty$, we show that the solution $\hat{\mathbf{W}}_N$ is uniformly distributed on \mathbb{S}^d when the hyperspherical energy defined in Eq. (1) achieves its minimum.

Definition 1 (minimal hyperspherical s -energy). *We define the minimal s -energy for N points on the unit hypersphere $\mathbb{S}^d = \{w \in \mathbb{R}^{d+1} \mid \|w\|=1\}$ as*

$$\epsilon_{s,d}(N) := \inf_{\hat{\mathbf{W}}_N \subset \mathbb{S}^d} \mathbf{E}_{s,d}(\hat{\mathbf{w}}_i |_{i=1}^N) \quad (6)$$

where the infimum is taken over all possible $\hat{\mathbf{W}}_N$ on \mathbb{S}^d . Any configuration of $\hat{\mathbf{W}}_N$ to attain the infimum is called an s -extremal configuration. Usually $\epsilon_{s,d}(N)=\infty$ if N is greater than d and $\epsilon_{s,d}(N)=0$ if $N=0, 1$.

We discuss the asymptotic behavior ($N \rightarrow \infty$) in three cases: $0 < s < d$, $s=d$, and $s > d$. We first write the energy integral as $I_s(\mu) = \iint_{\mathbb{S}^d \times \mathbb{S}^d} \|\mathbf{u} - \mathbf{v}\|^{-s} d\mu(\mathbf{u}) d\mu(\mathbf{v})$, which is taken over all probability measure μ supported on \mathbb{S}^d . With $0 < s < d$, $I_s(\mu)$ is minimal when μ is the spherical measure $\sigma^d = \mathcal{H}^d(\cdot)|_{\mathbb{S}^d} / \mathcal{H}^d(\mathbb{S}^d)$ on \mathbb{S}^d , where $\mathcal{H}^d(\cdot)$ denotes the d -dimensional Hausdorff measure. When $s \geq d$, $I_s(\mu)$ becomes infinity, which therefore requires different analysis. In general, we can say all s -extremal configurations asymptotically converge to uniformly distribution on a hypersphere, as stated in Theorem 1. This asymptotic behavior has been heavily studied in [36, 21, 11].

Theorem 1 (asymptotic uniform distribution on hypersphere). *Any sequence of optimal s -energy configurations $(\hat{\mathbf{W}}_N^*)|_2^\infty \subset \mathbb{S}^d$ is asymptotically uniformly distributed on \mathbb{S}^d in the sense of the weak-star topology of measures, namely*

$$\frac{1}{N} \sum_{\mathbf{v} \in \hat{\mathbf{W}}_N^*} \delta_{\mathbf{v}} \rightarrow \sigma^d, \quad \text{as } N \rightarrow \infty \quad (7)$$

where $\delta_{\mathbf{v}}$ denotes the unit point mass at \mathbf{v} , and σ^d is the spherical measure on \mathbb{S}^d .

Theorem 2 (asymptotics of the minimal hyperspherical s -energy). *We have that $\lim_{N \rightarrow \infty} \frac{\epsilon_{s,d}(N)}{p(N)}$ exists for the minimal s -energy. For $0 < s < d$, $p(N) = N^2$. For $s=d$, $p(N) = N^2 \log N$. For $s > d$, $p(N) = N^{1+s/d}$. Particularly if $0 < s < d$, we have $\lim_{N \rightarrow \infty} \frac{\epsilon_{s,d}(N)}{N^2} = I_s(\sigma^d)$.*

Theorem 2 tells us the growth power of the minimal hyperspherical s -energy when N goes to infinity. Therefore, different potential power s leads to different optimization dynamics. In the light of the behavior of the energy integral, MHE regularization will focus more on local influence from neighborhood neurons instead of global influences from all the neurons as the power s increases.

4.2 Generalization and Optimality

As rigorously proved in [52], in one-hidden-layer ReLU neural network, the diversity of units can effectively eliminate the spurious local minima despite the non-convexity in learning dynamics of neural networks. Following such argument, our MHE regularization, which encourages the diversity of neurons, naturally matches the theoretical intuition in [52], and can effectively promote the generalization of the corresponding neural networks. This is also well verified by our comprehensive experiments in Section 5. While hyperspherical energy is minimized such that neurons become increasingly more uniformly distributed on a hypersphere, then such hyperspherical diversity is closely related to the generalization error, as shown in main results of [52].

5 Applications and Experiments

5.1 Improving Network Generalization

We use MHE to improve the generalization of CNNs and study how MHE performs in a variety of scenarios. First, we perform ablation study and some exploratory experiments on MHE. Then we apply MHE to large-scale object recognition and class-imbalance learning. For all the experiments on CIFAR-10 and CIFAR-100 in the paper, we use moderate data augmentation, following [13, 25]. For ImageNet-2012, we follow the same data augmentation in [28]. We train all the networks using SGD with momentum 0.9, and the network initialization follows [12]. All the networks use BN [18] and ReLU if not otherwise specified. Experimental details are given in each subsection and Appendix A.

5.1.1 Ablation Study and Exploratory Experiments

Variants of MHE. We first evaluate all different variants of MHE on CIFAR-10 and CIFAR-100. The compared variants include the original MHE (with the power $s=0, 1, 2$) and half-space MHE (with the power $s=0, 1, 2$) in both Euclidean and angular distance. In this experiment, all the methods use the CNN-9 (see Appendix A). The results in Table 1 show that all the variants of MHE performs consistently better than the baseline. Specifically, the half-space MHE has significant performance gain compared to the other MHE variants, and MHE in Euclidean and angular distance perform similarly. MHE with $s=2$ performs best among $s=0, 1, 2$. In the following experiments, we use $s=2$ and Euclidean distance in both MHE and half-space MHE if not otherwise specified.

Network width. We evaluate MHE with different network width. We use CNN-9 as our base network, and change its filter number in Conv1.x, Conv2.x and Conv3.x (see Appendix A). We experiment different network width including 16/32/64, 32/64/128, 64/128/256 and 128/256/512. Results is given in Table 2. One can observe that both MHE and half-space MHE consistently outperforms the baseline, showing the stronger generalization. Interestingly, both MHE and half-space MHE have more significant gain while the filter number is smaller in each layer, indicating that our MHE regularization can help the network to make better use of the neurons. Moreover, the half-space MHE is consistently better than MHE, showing the necessity of reducing colinearity redundancy.

Network depth. We perform experiments with different network depth to better evaluate the performance of MHE. We fix the filter number in Conv1.x, Conv2.x and Conv3.x to 64, 128 and 256, respectively. We compare 6-layer CNN, 9-layer CNN and 15-layer CNN. The results are given in Table 3. Both MHE and half-space MHE performs significantly better than the baseline. More interestingly, baseline CNN-15 can not converge, while CNN-15 is able to converge reasonably well if we use MHE to regularize the network. Moreover, we also see that half-space MHE can consistently show better generalization than MHE with different network depth.

Method	CIFAR-10			CIFAR-100		
	$s=2$	$s=1$	$s=0$	$s=2$	$s=1$	$s=0$
MHE	6.22	6.74	6.44	27.15	27.09	26.16
Half-space MHE	6.28	6.54	6.30	25.61	26.30	26.18
A-MHE	6.21	6.77	6.45	26.17	27.31	27.90
Half-space A-MHE	6.52	6.49	6.44	26.03	26.52	26.47
Baseline			7.75			28.13

Table 1: Testing error (%) of different MHE on CIFAR-10/100.

Method	16/32/64	32/64/128	64/128/256	128/256/512
Baseline	47.72	38.64	28.13	24.95
MHE	36.84	30.05	26.75	24.05
Half-space MHE	35.16	29.33	25.96	23.38

Table 2: Testing error (%) of different width on CIFAR-100.

Method	CNN-6	CNN-9	CNN-15
Baseline	32.08	28.13	N/C
MHE	28.16	26.75	26.9
Half-space MHE	27.56	25.96	25.84

Table 3: Testing error (%) of different depth on CIFAR-100. N/C: not converged.

Ablation study. Since the current MHE regularizes the neurons in the hidden layers and the output layer simultaneously, we perform ablation study for MHE to further investigate where the gain comes from. This experiment uses the CNN-9. The results are given in Table 4. “H” means that we apply MHE to all the hidden layers, while “O” means that we apply MHE to the output layer. Because the half-space MHE can not be applied to the output layer, so there is “N/A” in the table. In general, we find that applying MHE to both the hidden layers and the output layer yields the best performance, and using MHE in the hidden layers usually produces better accuracy than using MHE in the output layer.

Hyperparameter experiment. We evaluate how the selection of hyperparameter affects the performance. We experiment with different hyperparameters from 10^{-2} to 10^2 on CIFAR-100 with the CNN-9. HS-MHE denotes the half-space MHE. We evaluate MHE variants by separately applying MHE to the output layer (“O”), MHE to the hidden layers (“H”), and the half-space MHE to the hidden layers (“H”). The results in Fig. 3 show that the our MHE is not very hyperparameter-sensitive and can consistently outperform the baseline. The half-space MHE can consistently outperform the original MHE under different hyperparameter settings. Moreover, applying MHE only to hidden layers can yields better accuracy than applying MHE only to output layer.

MHE for ResNets. Besides the standard CNN, we also evaluate MHE on ResNet-32 to show that our MHE is architecture-agnostic and can improve accuracy on multiple types of architectures. Detailed architecture settings are given in Appendix A. The results on CIFAR-10 and CIFAR-100 are given in Table 5. One can observe that applying MHE to ResNet also achieves considerable improvement. Most importantly, adding MHE regularization will not affect the original settings of the architecture, and it can improve the network generalization at a very small computational cost.

5.1.2 Large-scale Object Recognition

We evaluate MHE on large-scale ImageNet-2012 datasets. Specifically, we perform experiment using ResNets, and then report the top-1 validation error (center crop) in Table 6. From the results, we still observe that both MHE and half-space MHE yields consistently better recognition accuracy than the baseline and the orthonormal regularization (after tuning its hyperparameter). To better evaluate the consistency of MHE’s performance gain, we use two ResNets with different depth: ResNet-18 and ResNet-34. On ResNet-18 and ResNet-34, both MHE and half-space MHE show better generalization power, and half-space MHE performs slightly better than full-space MHE.

5.1.3 Class-imbalance Learning

Because MHE aims to maximize the hyperspherical margin between different classifier neurons in the output layer, we can naturally apply MHE to class-imbalance learning where the number of training samples in different classes is imbalanced. We demonstrate the power of MHE in class-imbalance learning through a toy experiment. We first randomly throw away 98% training data for digit 0 in MNIST (only 100 samples are preserved for digit 0), and then train a 6-layer CNN on this imbalance MNIST dataset. To visualize the learned features, we set the output feature dimension as 2. The learned features and classifier neurons on training set is shown in Fig. 4 where each color denotes a digit and red arrows denote the classifier neurons. From Fig. 4, one can observe that the CNN without MHE only tends to ignore the imbalanced class (digit 0) and the learned classifier neuron is highly biased to another digit. In contrast, the CNN with MHE can learn reasonably separable distribution even if digit 0 only has 2% samples compared to the other classes. Using MHE can improve the accuracy on the full testing set from 88.5% to 98%. Most importantly, the classifier neuron for digit 0 is also properly learned.

Method	H O × √	H O √ ×	H O √ √
MHE	26.85	26.55	26.16
Half-space MHE	N/A	26.28	25.61
A-MHE	27.8	26.56	26.17
Half-space A-MHE	N/A	26.64	26.03
Baseline		28.13	

Table 4: Ablation study on CIFAR-100.

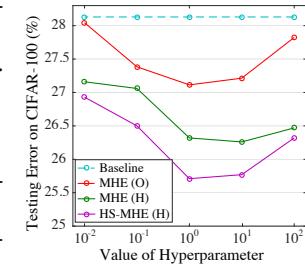


Figure 3: Hyperparameter.

Method	CIFAR-10	CIFAR-100
ResNet-110-original [13]	6.61	25.16
ResNet-1001 [14]	4.92	22.71
ResNet-1001 (64 batch) [14]	4.64	-
baseline	5.19	22.87
MHE	4.72	22.19
Half-space MHE	4.66	22.04

Table 5: Error (%) of ResNet-32.

Method	ResNet-18	ResNet-34
baseline	33.95	30.04
Orthonormal	33.61	29.75
MHE	33.50	29.60
Half-space MHE	33.45	29.50

Table 6: Top1 error (%) on ImageNet.

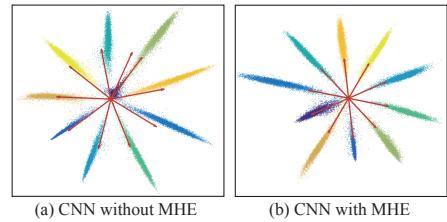


Figure 4: Class-imbalance learning on MNIST.

We experiment MHE in two data imbalance settings on CIFAR-10: 1) single class imbalance (S) - All classes have the same number of images but one single class has significantly less number, and 2) multiple class imbalance (M) - The number of images decreases as the class index decreases from 9 to 0. We use CNN-9 for all the compared regularizations. Details are provided in Appendix A. In Table 7, we report the error rate on the whole testing set. In addition, we report the error rate (denoted by Err. (S)) on the imbalance class (single imbalance setting) in the full testing set. From the results, one can observe that CNN-9 with MHE is able to effectively perform recognition when classes are imbalanced. Even only given a small portion of training data in a few classes, CNN-9 with MHE can achieve competitive accuracy on the full testing set, showing MHE’s superior generalization power.

5.2 SphereFace+: Improving Inter-class Feature Separability via MHE for Face Recognition

We have shown that full-space MHE for output layers can encourage classifier neurons to distribute more evenly on hypersphere and improve inter-class feature separability. Intuitively, the classifier neurons serve as the approximate center for features from each class, and can therefore guide the feature learning. We also observe that open-set face recognition (*e.g.*, face verification) requires the feature centers to be as separable as possible [26]. This connection inspires us to apply MHE to face recognition. Specifically, we propose *SphereFace+* by applying MHE to SphereFace [26]. The objective of SphereFace, angular softmax loss (ℓ_{SF}) that encourages intra-class feature compactness, is naturally complimentary to that of MHE. The objective function of SphereFace+ is defined as

$$\mathcal{L}_{SF+} = \underbrace{\frac{1}{m} \sum_{j=1}^m \ell_{SF}(\langle \mathbf{w}_i^{\text{out}}, \mathbf{x}_j \rangle_{i=1}^c, \mathbf{y}_j, m_{SF})}_{\text{Angular softmax loss: promoting intra-class compactness}} + \lambda_M \cdot \underbrace{\frac{1}{m(N-1)} \sum_{i=1}^m \sum_{j=1, j \neq y_i}^N f_s(\|\hat{\mathbf{w}}_{y_i}^{\text{out}} - \hat{\mathbf{w}}_j^{\text{out}}\|)}_{\text{MHE: promoting inter-class separability}} \quad (8)$$

where c is the number of classes, m is the mini-batch size, N is the number of classifier neurons, \mathbf{x}_i the deep feature of the i -th face (y_i is its groundtruth label), and $\mathbf{w}_i^{\text{out}}$ is the i -th classifier neuron. m_{SF} is a hyperparameter for SphereFace, controlling the degree of intra-class feature compactness. Because the face dataset usually has thousands of identities, we will use the data-dependent mini-batch approximation MHE in the output layer, as shown in Eq. (8). MHE completes a missing piece for SphereFace by promoting the inter-class separability. Our SphereFace+ consistently outperforms SphereFace, and achieves state-of-the-art performance on LFW [16] and MegaFace [20] datasets.

m_{SF}	LFW		MegaFace	
	SphereFace	SphereFace+	SphereFace	SphereFace+
1	96.35	97.15	39.12	45.90
2	98.87	99.05	60.48	68.51
3	98.97	99.13	63.71	66.89
4	99.26	99.32	70.68	71.30

Table 8: Accuracy (%) on SphereFace-20 network.

m_{SF}	LFW		MegaFace	
	SphereFace	SphereFace+	SphereFace	SphereFace+
1	96.93	97.47	41.07	45.55
2	99.03	99.22	62.01	67.07
3	99.25	99.35	69.69	70.89
4	99.42	99.47	72.72	73.03

Table 9: Accuracy (%) on SphereFace-64 network.

Performance under different m_{SF} . We evaluate SphereFace+ with two different architectures (SphereFace-20 and SphereFace-64) proposed in [26]. Specifically, SphereFace-20 and SphereFace-64 are 20-layer and 64-layer modified residual networks, respectively. We train our network with the publicly available CASIA-Webface dataset [58], and then test the learned model on LFW and MegaFace dataset. In MegaFace dataset, the reported accuracy indicates rank-1 identification accuracy with 1 million distractors. All the results in Table 8 and Table 9 are computed without model ensemble and PCA. One can observe that SphereFace+ consistently outperforms SphereFace by a considerable margin on both LFW and MegaFace datasets under all different settings of m_{SF} . Moreover, the performance gain generalizes across different architectures.

Comparison to state-of-the-art methods. We also compare our methods with some widely used loss functions. All these compared methods use SphereFace-64 network that are trained with CASIA dataset. All the results are given in Table 10 computed without model ensemble and PCA. Compared to the other state-of-the-art methods, SphereFace+ achieves the best accuracy on LFW dataset, while being comparable to the best accuracy on MegaFace dataset. Current state-of-the-art face recognition methods [47, 26, 49, 5, 29] usually only focus on compressing the intra-class features, which makes MHE a potentially useful tool in order to further improve these face recognition methods.

Method	LFW	MegaFace
Softmax Loss	97.88	54.86
Softmax+Contrastive [43]	98.78	65.22
Triplet Loss [38]	98.70	64.80
L-Softmax Loss [27]	99.10	67.13
Softmax+Center Loss [51]	99.05	65.49
CosineFace [49, 47]	99.10	75.10
SphereFace	99.42	72.72
SphereFace+ (ours)	99.47	73.03

Table 10: Comparison to state-of-the-art.

6 Concluding Remarks

We borrow the idea from physics and propose a novel regularization method, minimum hyperspherical energy (MHE), to encourage the diversity of neural representation. Neural nets with MHE show consistent improvements on performance in our experiments. Finally, MHE provides a new point of view on the role of neuron weights and may have many additional potential applications in model compression, image generation, etc.

Acknowledgements

We would like to thank NVIDIA corporation for donating Titan Xp GPUs to support our research. We also thank Tuo Zhao (Georgia Tech) for the valuable discussions and suggestions.

References

- [1] Alireza Aghasi, Nam Nguyen, and Justin Romberg. Net-trim: A layer-wise convex pruning of deep neural networks. In *AISTATS*, 2017.
- [2] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.
- [3] Andrew Brock, Theodore Lim, James M Ritchie, and Nick Weston. Neural photo editing with introspective adversarial networks. In *ICLR*, 2017.
- [4] Michael Cogswell, Faruk Ahmed, Ross Girshick, Larry Zitnick, and Dhruv Batra. Reducing overfitting in deep networks by decorrelating representations. In *ICLR*, 2016.
- [5] Jiankang Deng, Jia Guo, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. *arXiv preprint arXiv:1801.07698*, 2018.
- [6] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NIPS*, 2014.
- [7] Mario Götz and Edward B Saff. Note on d—extremal configurations for the sphere in r^{d+1} . In *Recent Progress in Multivariate Approximation*, pages 159–162. Springer, 2001.
- [8] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *NIPS*, 2017.
- [9] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. In *ICLR*, 2016.
- [10] DP Hardin and EB Saff. Minimal riesz energy point configurations for rectifiable d-dimensional manifolds. *arXiv preprint math-ph/0311024*, 2003.
- [11] DP Hardin and EB Saff. Discretizing manifolds via minimum energy points. *Notices of the AMS*, 51(10):1186–1194, 2004.
- [12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *ICCV*, 2015.
- [13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.
- [14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *ECCV*, 2016.
- [15] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.
- [16] Gary B Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical report, Technical Report, 2007.

- [17] Forrest N Iandola, Song Han, Matthew W Moskewicz, Khalid Ashraf, William J Dally, and Kurt Keutzer. SqueezeNet: Alexnet-level accuracy with 50x fewer parameters and < 0.5 mb model size. *arXiv preprint arXiv:1602.07360*, 2016.
- [18] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015.
- [19] Lu Jiang, Deyu Meng, Shou-I Yu, Zhenzhong Lan, Shiguang Shan, and Alexander Hauptmann. Self-paced learning with diversity. In *NIPS*, 2014.
- [20] Ira Kemelmacher-Shlizerman, Steven M Seitz, Daniel Miller, and Evan Brossard. The megaface benchmark: 1 million faces for recognition at scale. In *CVPR*, 2016.
- [21] Arno Kuijlaars and E Saff. Asymptotics for minimal discrete energy on the sphere. *Transactions of the American Mathematical Society*, 350(2):523–538, 1998.
- [22] Ludmila I Kuncheva and Christopher J Whitaker. Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine learning*, 51(2):181–207, 2003.
- [23] Naum Samoilovich Landkof. *Foundations of modern potential theory*, volume 180. Springer, 1972.
- [24] Nan Li, Yang Yu, and Zhi-Hua Zhou. Diversity regularized ensemble pruning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 2012.
- [25] Weiyang Liu, Zhen Liu, Zhiding Yu, Bo Dai, Rongmei Lin, Yisen Wang, James M Rehg, and Le Song. Decoupled networks. *CVPR*, 2018.
- [26] Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song. Sphereface: Deep hypersphere embedding for face recognition. In *CVPR*, 2017.
- [27] Weiyang Liu, Yandong Wen, Zhiding Yu, and Meng Yang. Large-margin softmax loss for convolutional neural networks. In *ICML*, 2016.
- [28] Weiyang Liu, Yan-Ming Zhang, Xinguo Li, Zhiding Yu, Bo Dai, Tuo Zhao, and Le Song. Deep hyperspherical learning. In *NIPS*, 2017.
- [29] Yu Liu, Hongyang Li, and Xiaogang Wang. Rethinking feature discrimination and polymerization for large-scale recognition. *arXiv preprint arXiv:1710.00870*, 2017.
- [30] Julien Mairal, Francis Bach, Jean Ponce, and Guillermo Sapiro. Online dictionary learning for sparse coding. In *ICML*, 2009.
- [31] Dmytro Mishkin and Jiri Matas. All you need is a good init. In *ICLR*, 2016.
- [32] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *ICLR*, 2018.
- [33] Ignacio Ramirez, Pablo Sprechmann, and Guillermo Sapiro. Classification and clustering via dictionary learning with structured incoherence and shared features. In *CVPR*, 2010.
- [34] Pau Rodríguez, Jordi Gonzalez, Guillem Cucurull, Josep M Gonfaus, and Xavier Roca. Regularizing cnns with locally constrained decorrelations. In *ICLR*, 2017.
- [35] Aruni RoyChowdhury, Prakhar Sharma, Erik Learned-Miller, and Aruni Roy. Reducing duplicate filters in deep neural networks. In *NIPS workshop on Deep Learning: Bridging Theory and Practice*, 2017.
- [36] Edward B Saff and Amo BJ Kuijlaars. Distributing many points on a sphere. *The mathematical intelligencer*, 19(1):5–11, 1997.
- [37] Tim Salimans and Diederik P Kingma. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In *NIPS*, 2016.
- [38] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *CVPR*, 2015.
- [39] Wenling Shang, Kihyuk Sohn, Diogo Almeida, and Honglak Lee. Understanding and improving convolutional neural networks via concatenated rectified linear units. In *ICML*, 2016.
- [40] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv:1409.1556*, 2014.

- [41] Steve Smale. Mathematical problems for the next century. *The mathematical intelligencer*, 20(2):7–15, 1998.
- [42] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *JMLR*, 15(1):1929–1958, 2014.
- [43] Yi Sun, Xiaogang Wang, and Xiaoou Tang. Deep learning face representation from predicting 10,000 classes. In *CVPR*, 2014.
- [44] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *CVPR*, 2015.
- [45] Pieter Merkus Lambertus Tammes. On the origin of number and arrangement of the places of exit on the surface of pollen-grains. *Recueil des travaux botaniques néerlandais*, 27(1):1–84, 1930.
- [46] Joseph John Thomson. Xxiv. on the structure of the atom: an investigation of the stability and periods of oscillation of a number of corpuscles arranged at equal intervals around the circumference of a circle; with application of the results to the theory of atomic structure. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 7(39):237–265, 1904.
- [47] Feng Wang, Weiyang Liu, Haijun Liu, and Jian Cheng. Additive margin softmax for face verification. *arXiv preprint arXiv:1801.05599*, 2018.
- [48] Feng Wang, Xiang Xiang, Jian Cheng, and Alan L Yuille. Normface: L2 hypersphere embedding for face verification. *arXiv preprint arXiv:1704.06369*, 2017.
- [49] Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Zhifeng Li, Dihong Gong, Jingchao Zhou, and Wei Liu. Cosface: Large margin cosine loss for deep face recognition. *arXiv preprint arXiv:1801.09414*, 2018.
- [50] David Warde-Farley and Yoshua Bengio. Improving generative adversarial networks with denoising feature matching. In *ICLR*, 2017.
- [51] Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao. A discriminative feature learning approach for deep face recognition. In *ECCV*, 2016.
- [52] Bo Xie, Yingyu Liang, and Le Song. Diverse neural network learns true target functions. *arXiv preprint arXiv:1611.03131*, 2016.
- [53] Di Xie, Jiang Xiong, and Shiliang Pu. All you need is beyond a good init: Exploring better solution for training extremely deep convolutional neural networks with orthonormality and modulation. *arXiv:1703.01827*, 2017.
- [54] Pengtao Xie, Yuntian Deng, Yi Zhou, Abhimanyu Kumar, Yaoliang Yu, James Zou, and Eric P Xing. Learning latent space models with angular constraints. In *ICML*, 2017.
- [55] Pengtao Xie, Aarti Singh, and Eric P Xing. Uncorrelation and evenness: a new diversity-promoting regularizer. In *ICML*, 2017.
- [56] Pengtao Xie, Wei Wu, Yichen Zhu, and Eric P Xing. Orthogonality-promoting distance metric learning: convex relaxation and theoretical analysis. In *ICML*, 2018.
- [57] Pengtao Xie, Jun Zhu, and Eric Xing. Diversity-promoting bayesian learning of latent variable models. In *ICML*, 2016.
- [58] Dong Yi, Zhen Lei, Shengcai Liao, and Stan Z Li. Learning face representation from scratch. *arXiv:1411.7923*, 2014.
- [59] Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, and Yu Qiao. Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE Signal Processing Letters*, 23(10):1499–1503, 2016.
- [60] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. *arXiv preprint arXiv:1707.01083*, 2017.

A Experimental Details

Layer	CNN-6	CNN-9	CNN-15
Conv1.x	$[3 \times 3, 64] \times 2$	$[3 \times 3, 64] \times 3$	$[3 \times 3, 64] \times 5$
Pool1		2×2 Max Pooling, Stride 2	
Conv2.x	$[3 \times 3, 128] \times 2$	$[3 \times 3, 128] \times 3$	$[3 \times 3, 128] \times 5$
Pool2		2×2 Max Pooling, Stride 2	
Conv3.x	$[3 \times 3, 256] \times 2$	$[3 \times 3, 256] \times 3$	$[3 \times 3, 256] \times 5$
Pool3		2×2 Max Pooling, Stride 2	
Fully Connected	256	256	256

Table 11: Our plain CNN architectures with different convolutional layers. Conv1.x, Conv2.x and Conv3.x denote convolution units that may contain multiple convolution layers. E.g., $[3 \times 3, 64] \times 3$ denotes 3 cascaded convolution layers with 64 filters of size 3×3 .

Layer	ResNet-32 for CIFAR-10/100	ResNet-18 for ImageNet-2012	ResNet-34 for ImageNet-2012
Conv0.x	N/A	$[7 \times 7, 64]$, Stride 2 3×3 , Max Pooling, Stride 2	$[7 \times 7, 64]$, Stride 2 3×3 , Max Pooling, Stride 2
Conv1.x	$\begin{bmatrix} [3 \times 3, 64] \times 1 \\ [3 \times 3, 64] \times 5 \end{bmatrix}$	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 3$
Conv2.x	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 5$	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 4$
Conv3.x	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 5$	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 6$
Conv4.x	N/A	$\begin{bmatrix} 3 \times 3, 512 \\ 3 \times 3, 512 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 512 \\ 3 \times 3, 512 \end{bmatrix} \times 3$
		Average Pooling	

Table 12: Our ResNet architectures with different convolutional layers. Conv0.x, Conv1.x, Conv2.x, Conv3.x and Conv4.x denote convolution units that may contain multiple convolutional layers, and residual units are shown in double-column brackets. Conv1.x, Conv2.x and Conv3.x usually operate on different size feature maps. These networks are essentially the same as [13], but some may have different number of filters in each layer. The downsampling is performed by convolutions with a stride of 2. E.g., $[3 \times 3, 64] \times 4$ denotes 4 cascaded convolution layers with 64 filters of size 3×3 , and S2 denotes stride 2.

General settings. The network architectures used in the paper are elaborated in Table 11 Table 12. For CIFAR-10 and CIFAR-100, we use batch size 128. We start with learning rate 0.1, divide it by 10 at 20k, 30k and 37.5k iterations, and terminate training at 42.5k iterations. For ImageNet-2012, we use batch size 64 and start with learning rate 0.1. The learning rate is divided by 10 at 150k, 300k and 400k iterations, and the training is terminated at 500k iterations. Note that, for all the compared methods, we always use the best possible hyperparameters to make sure that the comparison is fair. The baseline has exactly the same architecture and training settings as the one that MHE uses, and the only difference is an additional MHE regularization. For full-space MHE in hidden layers, we set λ_h as 10 for all experiments. For half-space MHE in hidden layers, we set λ_h as 1 for all experiments. For MHE in output layers, we set λ_o as 1 for all experiments. We use $1e-5$ for the orthonormal regularization. If not otherwise specified, standard ℓ_2 weight decay ($1e-4$) is applied to all the neural network including baselines and the networks that use MHE regularization. A very minor issue for the hyperparameters λ_h is that it may increase as the number of layers increases, so we can potentially further divide the hyperspherical energy for the hidden layers by the number of layers. It will probably change the current optimal hyperparameter setting by a constant multiplier. For notation simplicity, we do not explicitly write out the weight decay term in the loss function in the main paper. Note that, all the neuron weights in the neural network are not normalized, and the MHE will normalize the neuron weights while computing the regularization loss. As a result, *MHE does not need to modify any component of the original neural networks, and it can simply be viewed as an extra regularization loss that can boost the performance*.

Class-imbalance learning. There are 50000 training images in the original CIFAR-10 dataset, with 5000 images per class. For the single class imbalance setting, we keep original images of class 1-9 and randomly throw away 90% images of class 0. The total number of training images in this setting is 45500. For the multiple class imbalance setting, we set the number of each class equals to $500 \times (\text{class_index} + 1)$. For instance, class 0 has 500 images, class 1 has 1000 images and class 9 has 5000 images. The total number of training images in this setting is 27500.

SphereFace+. SphereFace+ uses the same face detection and alignment method [59] as SphereFace [26]. The testing protocol on LFW and MegaFace is also the same as SphereFace. We use exactly the same preprocessing as in the SphereFace repository. Detailed network architecture settings of SphereFace-20 and SphereFace-64 can be found in [26]. Specifically, we use full-space MHE with Euclidean distance and $s = 2$ in the output layer. Essentially, we treat MHE as an additional loss function which aims to enlarge the inter-class angular distance of features and serves a complimentary role to the angular softmax in SphereFace. Note that, for the results of CosineFace [49], we directly use the results (with the same training settings and without using feature normalization) reported in the paper. Since ours also does not perform feature normalization, it is a fair comparison. With feature normalization, we find that the performance of SphereFace+ will also be improved significantly. However, feature normalization makes the results more tricky, because it will involve another hyperparameter that controls the projection radius of feature normalization.

B Proof of Theorem 1 and Theorem 2

Theorem 1 and Theorem 2 are natural results from classic potential theory [23] and spherical configuration [10, 21, 7]. We discuss the asymptotic behavior ($N \rightarrow \infty$) in three cases: $0 < s < d$, $s = d$, and $s > d$. We first write the energy integral as

$$I_s(\mu) = \iint_{\mathbb{S}^d \times \mathbb{S}^d} \|\mathbf{u} - \mathbf{v}\|^{-s} d\mu(\mathbf{u}) d\mu(\mathbf{v}), \quad (9)$$

which is taken over all probability measure μ supported on \mathbb{S}^d . With $0 < s < d$, $I_s(\mu)$ is minimal when μ is the spherical measure $\sigma^d = \mathcal{H}^d(\cdot)|_{\mathbb{S}^d} / \mathcal{H}^d(\mathbb{S}^d)$ on \mathbb{S}^d , where $\mathcal{H}^d(\cdot)$ denotes the d -dimensional Hausdorff measure. When $s \geq d$, $I_s(\mu)$ becomes infinity, which therefore requires different analysis.

First, the classic potential theory [23] can directly give the following results for the case where $0 < s < d$:

Lemma 1. *If $0 < s < d$,*

$$\lim_{N \rightarrow \infty} \frac{\varepsilon_{s,d}(N)}{N^2} = I_s\left(\frac{\mathcal{H}^d(\cdot)|_{\mathbb{S}^d}}{\mathcal{H}^d(\mathbb{S}^d)}\right), \quad (10)$$

where I_s is defined in the main paper. Moreover, any sequence of optimal hyperspherical s -energy configurations $(\hat{\mathbf{W}}_N^*)|_2^\infty \subset \mathbb{S}^d$ is asymptotically uniformly distributed in the sense that for the weak-star topology measures,

$$\frac{1}{N} \sum_{\mathbf{v} \in \hat{\mathbf{W}}_N^*} \delta_{\mathbf{v}} \rightarrow \sigma^d, \quad \text{as } N \rightarrow \infty \quad (11)$$

where $\delta_{\mathbf{v}}$ denotes the unit point mass at \mathbf{v} , and σ^d is the spherical measure on \mathbb{S}^d .

which directly concludes Theorem 1 and Theorem 2 in the case of $0 < s < d$.

For the case where $s = d$, we have from [21, 7] the following results:

Lemma 2. *Let $\mathcal{B}^d := \bar{B}(0, 1)$ be the closed unit ball in \mathbb{R}^d . For $s = d$,*

$$\lim_{N \rightarrow \infty} \frac{\varepsilon_{s,d}(N)}{N^2 \log N} = \frac{\mathcal{H}^d(\mathcal{B}^d)}{\mathcal{H}^d(\mathbb{S}^d)} = \frac{1}{d} \frac{\Gamma(\frac{d+1}{2})}{\sqrt{\pi} \Gamma(\frac{d}{2})}, \quad (12)$$

and any sequence $(\hat{\mathbf{W}}_N^*)|_2^\infty \subset \mathbb{S}^d$ of optimal s -energy configurations satisfies Eq. 11.

which concludes the case of $s = d$. Therefore, we are left with the case where $s > d$. For this case, we can use the results from [10]:

Lemma 3. *Let $A \subset \mathbb{R}^d$ be compact with $\mathcal{H}_d(A) > 0$, and $\tilde{\mathbf{W}}_N = \{\mathbf{x}_{k,N}\}_{i=1}^N$ be a sequence of asymptotically optimal N -point configurations in A in the sense that for some $s > d$,*

$$\lim_{N \rightarrow \infty} \frac{\mathbf{E}_s(\tilde{\mathbf{W}}_N)}{N^{1+s/d}} = \frac{C_{s,d}}{\mathcal{H}^d(A)^{s/d}} \quad (13)$$

or

$$\lim_{N \rightarrow \infty} \frac{\mathbf{E}_s(\tilde{\mathbf{W}}_N)}{N^2 \log N} = \frac{\mathcal{H}^d(\mathcal{B}^d)}{\mathcal{H}^d(A)}. \quad (14)$$

where $C_{s,d}$ is a finite positive constant independent of A . Let δ_x be the unit point mass at the point x . Then in the weak-star topology of measures we have

$$\frac{1}{N} \sum_{i=1}^N \delta_{x_{i,N}} \rightarrow \frac{\mathcal{H}^d(\cdot)|_A}{\mathcal{H}^d(A)}, \quad \text{as } N \rightarrow \infty. \quad (15)$$

The results naturally prove the case of $s > d$. Combining these three lemmas, we have proved Theorem 1 and Theorem 2.

C Understanding MHE from Decoupled View

Inspired by decoupled networks [25], we can view the original convolution as the multiplication of the angular function $g(\cdot)$ and the magnitude function $h(\cdot)$:

$$f(\mathbf{w}, \mathbf{x}) = \|\mathbf{w}\| \|\mathbf{x}\| \cdot \cos(\theta) \quad (16)$$

where θ is the angle between the kernel \mathbf{w} and the input \mathbf{x} . From the equation above, we can see that the norm of the kernel and the direction (*i.e.*, angle) of the kernel affect the inner product similarity differently. Typically, weight decay is to regularize the kernel by minimizing its ℓ_2 norm, while there is no regularization on the direction of the kernel. Therefore, MHE is able to complete this missing piece by promoting angular diversity. By combining MHE to a standard neural networks (*e.g.*, CNNs), the regularization term becomes

$$\mathcal{L}_{\text{reg}} = \underbrace{\lambda_w \cdot \frac{1}{\sum_{j=1}^L N_j} \sum_{j=1}^L \sum_{i=1}^{N_j} \|\mathbf{w}_i\|}_{\text{Weight decay: regularizing the magnitude of kernels}} + \underbrace{\lambda_h \cdot \sum_{j=1}^{L-1} \frac{1}{N_j(N_j - 1)} \{\mathbf{E}_s\}_j + \lambda_o \cdot \frac{1}{N_L(N_L - 1)} \mathbf{E}_s(\hat{\mathbf{w}}_i^{\text{out}}|_{i=1}^c)}_{\text{MHE: regularizing the direction of kernels}} \quad (17)$$

where \mathbf{x}_i is the feature of the i -th training sample entering the output layer, $\mathbf{w}_i^{\text{out}}$ is the classifier neuron for the i -th class in the output fully-connected layer and $\hat{\mathbf{w}}_i^{\text{out}}$ denotes its normalized version. $\{\mathbf{E}_s\}_i$ denotes the hyperspherical energy for the neurons in the i -th layer. c is the number of classes, m is the batch size, L is the number of layers of the neural network, and N_i is the number of neurons in the i -th layer. $\mathbf{E}_s(\hat{\mathbf{w}}_i^{\text{out}}|_{i=1}^c)$ denotes the hyperspherical energy of neurons $\{\hat{\mathbf{w}}_1^{\text{out}}, \dots, \hat{\mathbf{w}}_c^{\text{out}}\}$ in the output layer. λ_w , λ_h and λ_o are weighting hyperparameters for these three regularization terms.

From the decoupled view, we can see that MHE is actually very meaningful in regularizing the neural networks, which also serves a complementary role to weight decay.

D Regularizing SphereNets with MHE

SphereNets [28] are a family of network networks that learns on hyperspheres. The filters in SphereNets only focus on the hyperspherical (*i.e.*, angular) difference. One can see that the intuition of SphereNets well matches that of MHE, so MHE can serve as a natural and effective regularization for SphereNets. Because SphereNets throw away all the magnitude information of filters, the weight decay can no longer serve as a form of regularization for SphereNets, which makes MHE a very useful regularization for SphereNets. Originally, we use the orthonormal regularization $\|\mathbf{W}^\top \mathbf{W} - \mathbf{I}\|_F^2$ to regularize SphereNets, where \mathbf{W} is the weight matrix of a layer with each column being a vectorized filter and \mathbf{I} is an identity matrix. We compare MHE, half-space MHE and orthonormal regularization for SphereNets. In this section, all the SphereNets use the same architecture as the CNN-9 in Table 11, the training setting is also the same as CNN-9. We only evaluate SphereNets with cosine SphereConv. Note that, $s = 0$ is actually the logarithmic hyperspherical energy (a relaxation of the original hyperspherical energy). From Table 13, we observe that SphereNets with MHE can outperform both the SphereNet baseline and SphereNets with the orthonormal regularization, showing that MHE is not only effective in standard CNNs but also very suitable for SphereNets.

Method	CIFAR-10			CIFAR-100		
	$s=2$	$s=1$	$s=0$	$s=2$	$s=1$	$s=0$
MHE	5.71	5.99	5.95	27.28	26.99	27.03
Half-space MHE	6.12	6.33	6.31	27.17	27.77	27.46
A-MHE	5.91	5.98	6.06	27.07	27.27	26.70
Half-space A-MHE	6.14	5.87	6.11	27.35	27.68	27.58
SphereNet with Orthonormal Reg.		6.13			27.95	
SphereNet Baseline		6.37			28.10	

Table 13: Testing error (%) of SphereNet with different MHE on CIFAR-10/100.

E Improving AM-Softmax with MHE

We also perform some preliminary experiments for applying MHE to additive margin softmax loss [47] which is a recently proposed well-performing objective function for face recognition. The loss function of AM-Softmax is given as follows:

$$\mathcal{L}_{\text{AMS}} = -\frac{1}{n} \sum_{i=1}^n \log \frac{e^{s \cdot (\cos \theta_{(\mathbf{x}_i, \mathbf{w}_{y_i})} - m_{\text{AMS}})}}{e^{s \cdot (\cos \theta_{(\mathbf{x}_i, \mathbf{w}_{y_i})} - m_{\text{AMS}})} + \sum_{j=1, j \neq y_i}^c e^{s \cdot \cos \theta_{(\mathbf{x}_i, \mathbf{w}_j)}}} \quad (18)$$

where y_i is the label of the training sample \mathbf{x}_i , n is the mini-batch size, m_{AMS} is the hyperparameter that controls the degree of angular margin, and $\theta_{(\mathbf{x}_i, \mathbf{w}_j)}$ denotes the angle between the training sample \mathbf{x}_i and the classifier neuron \mathbf{w}_j . s is the hyperparameter that controls the projection radius of feature normalization [48, 47]. Similar to our SphereFace+, we combine full-space MHE to the output layer to improve the inter-class feature separability. It is essentially following the same intuition of SphereFace+ by adding an additional loss function to AM-Softmax loss.

Experiments. We perform a preliminary experiment to study the benefits of MHE for improving AM-Softmax loss. We use the SphereFace-20 network and trained on CASIA-WebFace dataset (training settings are exactly the same as SphereFace+ in the main paper and [26]). The hyperparameters s, m_{AMS} for AM-Softmax loss exactly follow the best setting in [47]. AM-Softmax achieves 99.26% accuracy on LFW, while combining MHE with AM-Softmax yields 99.37% accuracy on LFW. Such performance gain is actually very significant in face verification, which further validates the superiority of MHE.

F Improving GANs with MHE

We propose to improve the discriminator of GANs using MHE. It has been pointed out in [32] that the function space from which the discriminators are learned largely affects the performance of GANs. Therefore, it is of great importance to learn a good discriminator for GANs. As a recently proposed regularization to stabilize the training of GANs, spectral normalization (SN) [32] encourages the Lipschitz constant of each layer’s weight matrix to be one. Since MHE exhibits significant performance gain for CNNs as a regularization, we expect MHE can also improve the training of GANs by regularizing its discriminator. As a result, we perform a preliminary evaluation on applying MHE to GANs.

Specifically, for all methods except WGAN-GP [8], we use the standard objective function for the adversarial loss:

$$V(G, D) := \mathbb{E}_{\mathbf{x} \sim q_{\text{data}}(\mathbf{x})}[\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z} \sim p(\mathbf{z})}[\log(1 - D(G(\mathbf{z})))], \quad (19)$$

where $\mathbf{z} \in \mathbb{R}^{d_z}$ is a latent variable, $p(\mathbf{z})$ is the normal distribution $\mathcal{N}(0, I)$, and $G : \mathbb{R}^{d_z} \rightarrow \mathbb{R}^{d_0}$ is a deterministic generator function. We set d_z to 128 in all the experiments. For the updates of G , we used the alternate cost proposed by [6] $-\mathbb{E}_{\mathbf{z} \sim p(\mathbf{z})}[\log(D(G(\mathbf{z})))]$ as used in [6, 50]. For the updates of D , we used the original cost function defined in Eq. (19).

Recall from [32] that spectral normalization normalizes the spectral norm of the weight matrix \mathbf{W} such that it makes the Lipschitz constraint $\sigma(\mathbf{W})$ to be one:

$$\bar{\mathbf{W}}_{\text{SN}}(\mathbf{W}) := \frac{\mathbf{W}}{\sigma(\mathbf{W})}. \quad (20)$$

We apply MHE to the discriminator of standard GANs (with the original loss function in [6]) for image generation on CIFAR-10. In general, our experimental settings and training strategies (including architectures in Table 15) exactly follow spectral normalization [32]. For MHE, we use the half-space variant with Euclidean distance (Eq. (1)). We first experiment regularizing the discriminator using MHE alone, and it yields comparable performance to SN and orthonormal regularization. Moreover, we also regularize the discriminator simultaneously using both MHE and SN, and it can give much better results than using either SN or MHE alone. The results in Table 14 show that MHE is potentially very useful for training GANs.

Method	Inception score
Real data	11.24±.12
Weight clipping	6.41±.11
GAN-gradient penalty (GP)	6.93±.08
WGAN-GP [8]	6.68±.06
Batch Normalization [18]	6.27±.10
Layer Normalization [2]	7.19±.12
Weight Normalization [37]	6.84±.07
Orthonormal [3]	7.40±.12
SN-GANs [32]	7.42±.08
MHE (ours)	7.32±.10
MHE+SN (ours) [32]	7.59±.08

Table 14: Inception scores with unsupervised image generation on CIFAR-10.

F.1 Network Architecture for GAN

We give the detailed network architectures in Table 15 that are used in our experiments for the generator and the discriminator.

Table 15: Our CNN architectures for image Generation on CIFAR-10. The slopes of all leaky ReLU (lReLU) functions in the networks are set to 0.1.

$z \in \mathbb{R}^{128} \sim \mathcal{N}(0, I)$
dense $\rightarrow M_g \times M_g \times 512$
4×4 , stride=2 deconv. BN 256 ReLU
4×4 , stride=2 deconv. BN 128 ReLU
4×4 , stride=2 deconv. BN 64 ReLU
3×3 , stride=1 conv. 3 Tanh

(a) Generator ($M_g = 4$ for CIFAR10).

RGB image $x \in \mathbb{R}^{M \times M \times 3}$
3×3 , stride=1 conv 64 lReLU
4×4 , stride=2 conv 64 lReLU
3×3 , stride=1 conv 128 lReLU
4×4 , stride=2 conv 128 lReLU
3×3 , stride=1 conv 256 lReLU
4×4 , stride=2 conv 256 lReLU
3×3 , stride=1 conv. 512 lReLU
dense $\rightarrow 1$

(b) Discriminator ($M = 32$ CIFAR10).

F.2 Comparison of Random Generated Images

We provide some randomly generated images for comparison between baseline GAN and GAN regularized by both MHE and SN. The generated images are shown in Fig. 5.

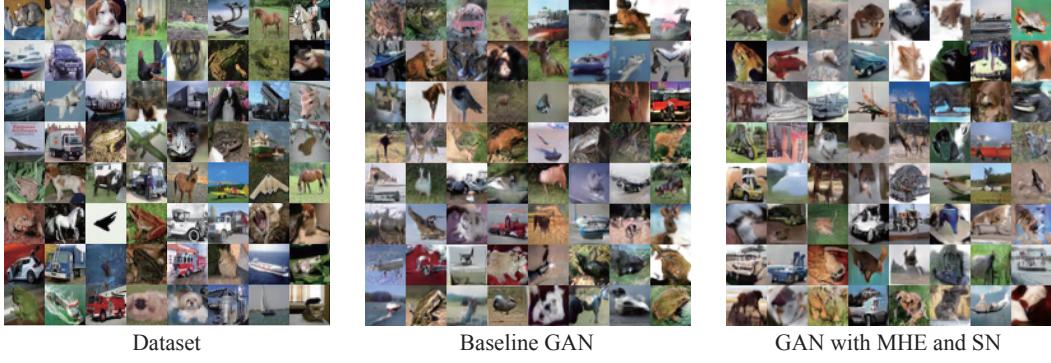


Figure 5: Results of generated images.

G More Results on Class-imbalance Learning

G.1 Class-imbalance learning on CIFAR-100

We perform additional experiments on CIFAR-100 to further validate the effectiveness of MHE in class-imbalance learning. In the CNN used in the experiment, we only apply MHE (*i.e.*, full-space MHE) to the output layer, and use MHE or half-space MHE in the hidden layers. In general, the experimental settings are the same as the main paper. We still use CNN-9 (which is a 9-layer CNN from Table 11) in the experiment. Slightly differently from CIFAR-10 in the main paper, the two data imbalance settings on CIFAR-100 include 1) 10-class imbalance (denoted as Single in Table 16) - All classes have the same number of images but 10 classes (index from 0 to 9) have significantly less number (only 10% training samples compared to the other normal classes), and 2) multiple class imbalance (denoted by Multiple in Table 16) - The number of images decreases as the class index decreases from 99 to 0. For the multiple class imbalance setting, we set the number of each class equals to $5 \times (\text{class_index} + 1)$. Experiment details are similar to the CIFAR-10 experiment, which is specified in Appendix A. The results in Table 16 show that MHE consistently improves CNNs in class-imbalance learning on CIFAR-100. In most cases, half-space MHE performs better than full-space MHE.

Method	Single	Multiple
Baseline	31.43	38.39
Orthonormal	30.75	37.89
MHE	29.30	37.07
Half-space MHE	29.40	36.52
A-MHE	30.16	37.54
Half-space A-MHE	29.60	37.07

Table 16: Error rate (%) on imbalanced CIFAR-100.

G.2 2D CNN Feature Visualization

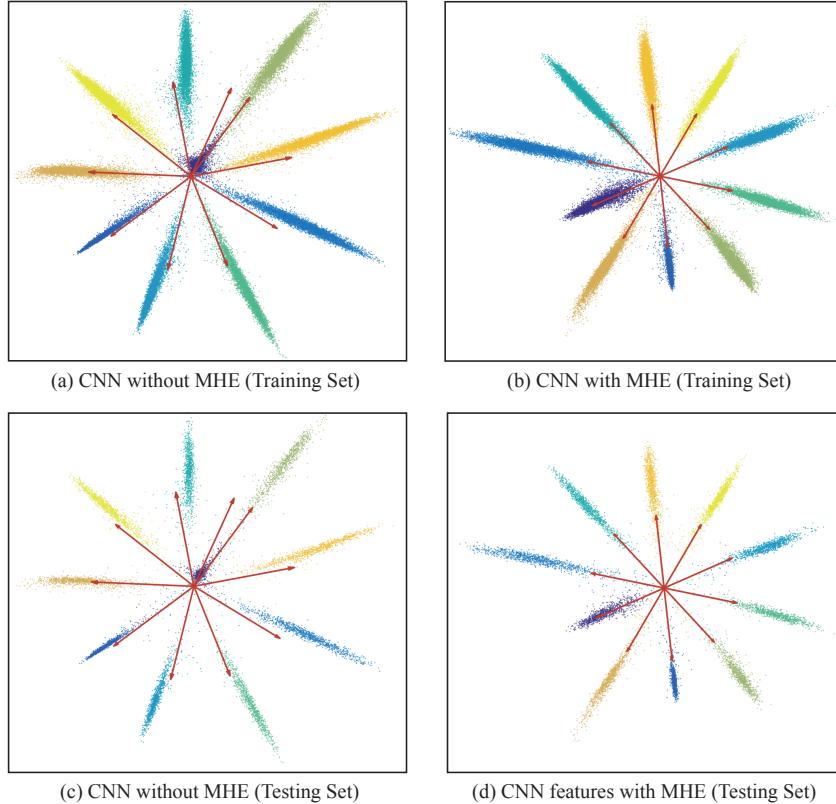


Figure 6: 2D CNN features with or without MHE on both training set and testing set. The features are computed by setting the output feature dimension as 2, similar to [27]. Each point denotes the 2D feature of a data point, and each color denotes a class. The red arrows are the classifier neurons of the output layer.

The experimental settings are the same as the main paper. We supplement the 2D feature visualization on testing set in Fig. 6. The visualized features on both training set and testing set well demonstrate the superiority of MHE in class-imbalance learning. In the CNN without MHE, the classifier neuron of the imbalanced training data is highly biased towards another class, and therefore can not be properly learned. In contrast, the CNN with MHE can learn uniformly distributed classifier neurons, which greatly improves the network's generalization ability.