IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 30 November 2023, accepted 12 December 2023, date of publication 25 December 2023,
date of current version 29 December 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3346430

RESEARCH ARTICLE

Deep Self-Supervised Diversity Promoting Learning on Hierarchical Hyperspheres for Regularization

YOUNGSUNG KIM[1], YOONSUK HYUN[2], JAE-JOON HAN[3], (Member, IEEE), EUNHO YANG[4], (Member, IEEE), SUNG JU HWANG[4], AND JINWOO SHIN[4]

[1]Department of Artificial Intelligence, Inha University, Incheon 22212, South Korea
[2]Department of Mathematics, Inha University, Incheon 22212, South Korea
[3]Samsung Advanced Institute of Technology (SAIT), Suwon 16678, South Korea
[4]Korea Advanced Institute of Science and Technology (KAIST), Daejeon 34141, South Korea

Corresponding author: Youngsung Kim (yskim.ee@gmail.com)

This work was supported in part by the Inha University Research Grant, and in part by the Institute of Information and Communications Technology Planning and Evaluation (IITP) Grant funded by the Korea Government [Ministry of Science and ICT (MSIT)] [Artificial Intelligence Convergence Innovation Human Resources Development (Inha University)] under Grant RS-2022-00155915.

ABSTRACT In this paper, we propose a novel approach to enhance the generalization performance of deep neural networks. Our method employs a hierarchical hypersphere-based constraint that organizes weight vectors hierarchically based on observed data. By diversifying the parameter space of hyperplanes in the classification layer, we aim to encourage discriminative generalization. We introduce a self-supervised grouping method designed to unveil hierarchical structures in scenarios with unknown hierarchy information. To maximize distances between weight vectors on multiple hyperspheres, we propose a novel metric that combines discrete and continuous measures. This regularization encourages diverse orientations, consequently leading to improved generalization. Extensive evaluations on datasets, including CUB200-2011, Stanford-Cars, CIFAR-100, and TinyImageNet, consistently demonstrate enhancements in classification performance compared to baseline settings.

INDEX TERMS Diversity promoting, hierarchical hyperspheres, inductive bias, regularization.

I. INTRODUCTION

The pursuit of improved generalization performance in machine learning has led to the widespread adoption of diversity-promoting learning techniques [1], [2]. In diversity-promoting learning, maximizing the pairwise distance between parameters can enhance their margin,[1] which allows projective representation to exhibit the desired discriminative property. Specifically, in the domain of deep neural networks addressing classification tasks, diversity-promoting learning presents a key advantage intuitively. It helps the model effectively generalize to diverse hyperplanes within similar classes during training. This mitigates the risk of the model becoming excessively specialized and ultimately contributes to the reduction of the generalization gap.

This diversity-promoting approach aims to enhance model performance by various means [1], [2], [4]. Techniques include increasing distances between parameters [2], increasing orthogonality [1], reducing parameter covariance [5], or minimizing correlation on feature vectors [6]. Among these methods, diversity-promoting regularization, specifically enforcing large diversity between projection parameters, has shown promising results without altering the underlying model architecture [2], [4], [5]. However, optimizing the objective function with a covariance matrix remains challenging [5]. Recent approaches have proposed diversity-promoting regularization by minimizing the energy of deep neural network parameters on a hypersphere. They use either Euclidean or angular metrics to maximize pairwise distances, achieving improved generalization [2], [4].

The associate editor coordinating the review of this manuscript and approving it for publication was Ines Domingues.

[1]It involves a different margin calculation compared to maximum-margin hyperplanes, such as the Support Vector Machine (SVM), which is determined based on the maximum distance to the nearest training sample of each class [3].

Building on the known efficient regularization techniques on the hypersphere [2], [4], we explore a novel integration of three different learning strategies: hierarchical learning, hyperspherical learning, and discrete metric learning. Firstly, we introduce hierarchical learning to embed semantic structure into the model. Drawing inspiration from the efficient nature of human intelligence [7], we employ a semantic taxonomy to arrange multiple classes hierarchically, thereby enhancing machine intelligence. The efficacy of hierarchical learning has been demonstrated in previous work [8], [9]. Secondly, we apply hyperspherical learning to confine parameters (hyperplane) within a bounded space. By representing parameters on hyperspheres, where points lie at an equidistance from a centroid, we introduce a bounded property facilitating the definition of a hierarchical structure with multiple separated hyperspheres. Finally, we leverage discrete metric learning to increase separability between parameters. Representing vector points as a discontinuous series with discrete representations enables isolation with a certain margin, making it suitable for addressing disconnected/groupwise manifold space problems. For categorization purposes, a discrete metric can force deep networks to learn such representations. Discrete metric spaces are expected to offer the advantage of reducing search efforts to satisfy such constraints. Moreover, importantly, enforcing equidistributed points with maximized pairwise distances while preserving hierarchical semantics across samples (classes) is a nontrivial task.

The key contributions of this paper are as follows:

- We propose a novel approach that applies hierarchical structures, specifically hierarchical angular constraints, to effectively regularize parameters defined on multiple hyperspheres.
- We propose a self-supervised method for introducing hierarchical hyperspheres, minimizing the need for additional parameters, with only centroid parameters indicating superclasses.
- Our exploration includes the use of a discrete angular metric which provides a suitable measurement for the multiple spaces. To maintain representation power, we blend a discrete metric with a continuous metric.
- Through extensive experiments on widely recognized benchmark datasets, we demonstrate the efficacy of our proposed method, particularly in the context of visual object classification.

II. RELATED WORKS

Diversity promotion in the embedding space or model parameters is a widely adopted strategy in machine learning to enhance generalization performance [1], [2], [3], [4], [5], [6], [7], [8], [9], [10], [11], [12], [13], [14]. This approach operates at various levels, including the feature level [1], [6], projection parameter level [2], [4], model ensemble level [14], [15], latent space model level [4], [14], and generative model level [12], [14]. However, certain regularization methods entail additional optimization effort. For example, enlarging pairwise distances between features [1] requires computational efforts due to the covariance matrix involved. In [5], unit-eigenvalue is utilized through singular value decomposition, adding complexity to the optimization process. Alternating direction method of multipliers (ADMM) [2] is employed to optimize the direction and magnitude of parameter vectors alternatively.

Regarding learning on a hypersphere, SphereCov (Hyperspherical Convolution) was proposed to replace the traditional inner-product-based convolution for learning angular representations [16]. Furthermore, regularization techniques leveraging diversity on a hypersphere have been introduced, based on the concept of Minimum Hyperspherical Energy [4].

In the context of hierarchical learning, previous works like [17] and [18] have applied hyperbolic spaces to embed data in deep neural networks, showcasing effective representation learning while preserving latent hierarchies compared to the Euclidean space. In contrast, our proposed method focuses on regularization learning using hierarchical hyperplane parameterization, which serves a distinct purpose from those representation-focused approaches.

In the context of metric learning, various metrics can be employed in a loss function. The Hamming distance metric [19] adopts a discrete mapping from the input space onto binary codes for metric learning. The unification of training objectives for deep hashing by incorporating a single classification objective is proposed in [20]. This is demonstrated through the maximization of cosine similarity between continuous codes and binary orthogonal targets under a cross-entropy loss.

In summary, existing diversity-promoting methods primarily focus on continuous distances or require additional complicated optimization strategies (e.g., covariance matrix or ADMM). Hyperspherical methods, on the other hand, operate on a single hypersphere and concentrate on parameter operations (e.g., SphereConv) or on continuous metric space only. In the context of discrete metric learning, either discrete or continuous metrics are adopted in a single training loss function. In contrast to these existing methods, our proposed methods function as a regularization for the parameters on the last layer by reparameterizing the hyperplanes in multiple hyperspherical spaces, which are hierarchically organized in a discrete blended metric space.

III. HIERARCHICAL HYPERSPHERES: MODELING DISCONNECTED MANIFOLDS

Real-world data often exhibit complex structures residing on disconnected manifolds, where the global manifold is formed by the disjoint union of multiple individual manifolds [21]. In this context, we propose addressing this challenge by decomposing a single space into multiple separated hyperspheres. Building upon these ideas, we introduce our novel approach, *Hierarchical Hyperspheres*, in the method section. This approach enhances the modeling of disconnected manifolds with the aim of improving performance and

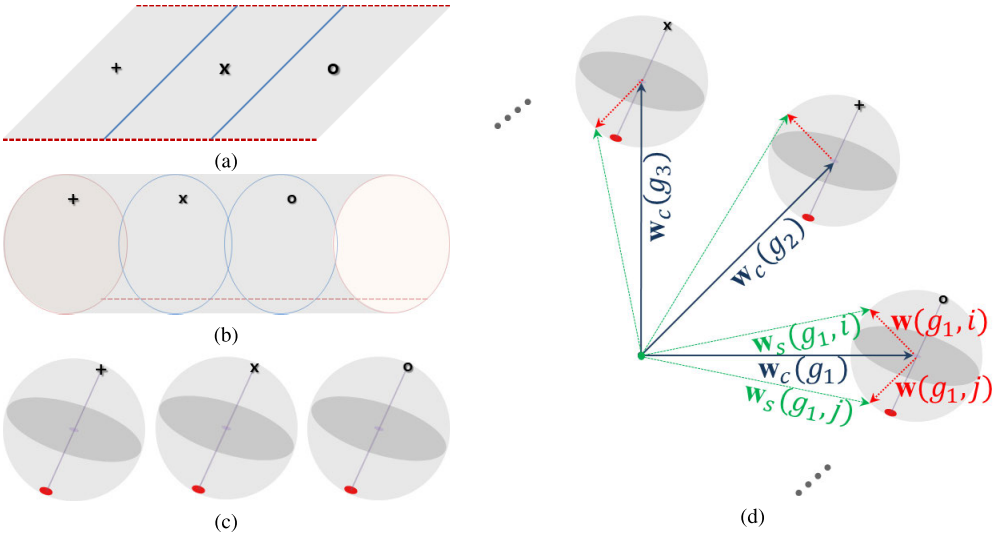


FIGURE 1. (a),(b), and (c) Multiple (hyper)spheres as quotient spaces of a topology space on Euclidean space might be found by gluing process with identifying points. Those separated hyperspheres are assumed to be under the quotient space conditions [10]. (d) Within an individual (hyper)sphere, projection parameters (weight vectors) in deep neural networks are defined to preserve a hierarchical structure. $\mathbf{w_s}$ and $\mathbf{w_c}$ represent the surface and center parameters ($\mathbf{w}_{\text{surface}}, \mathbf{w}_{\text{center}}$), respectively. As described in IV-A1, the i-th parameter of group g_k is defined as $\mathbf{w}(g_k, i) := \mathbf{w}_{\text{surface}}(g_k, i) - \mathbf{w}_{\text{center}}(g_k)$. The space can be formed in a series: (a)→(b)→(c)→(d).

facilitating the exploration of hierarchical structures in neural network learning tasks.

A. CONSTRUCTION OF DISCONNECTED MANIFOLDS VIA EQUIVALENT RELATIONS

As it is impractical to measure pairwise distances between high-dimensional vectors embedding the hierarchical structure within a single space, we introduce an identification space where multiple manifolds are isolated using equivalence relations [10]. In this context, we denote the d-sphere, $\mathbb{S}^d$, as the set of points satisfying $\mathbb{S}^d = \mathbf{w} \in \mathbb{R}^{d+1} : \|\mathbf{w}\| = 1$, centered at the origin. By employing multiple identifying relations, we construct several separated hyperspheres, each characterized by a center parameter $\mathbf{w}_c$ and a surface parameter $\mathbf{w}_s$. These parameters define a projection parameter $\mathbf{w}$ that constitutes a hypersphere space, as illustrated in Fig. 1. This construction facilitates the representation of hierarchical structures within the data.

B. PRIOR DISTRIBUTION AND REGULARIZATION

To achieve a uniform distribution of parameters on the unit hyperspheres, we sample the parameters from a Gaussian normal distribution [22], [23]. This choice is motivated by the spherical symmetry of the normal distribution [22], which promotes a balanced spread of parameters.[2] From a Bayesian perspective, neural networks with Gaussian priors induce l^2-norm regularization (weight decay) [24]. This regularization further emphasizes the importance of enforcing parameters to have Gaussian priors in hyperspherical learning within neural networks. Notably, parameters calculated through the difference arithmetic operation with two parameters on the normal Gaussian distribution follow a *normal difference distribution*.

In summary, our proposed Hierarchical Hyperspheres approach offers an effective solution for modeling disconnected manifolds, enabling enhanced exploration of hierarchical structures, and contributing to improved performance in neural network learning tasks.

IV. PROPOSED METHOD

In deep neural networks with constrained optimization, a regularization function $\mathcal{R}(\boldsymbol{\theta})$, with a regularization multiplier $\lambda > 0$, is added to the loss function to form a single objective function denoted as $\mathcal{J}_{\mathcal{R}}(\boldsymbol{\theta})$. This objective function comprises a loss term $\mathcal{L}(\mathbf{x}, \boldsymbol{\theta})$ and the regularization term. The goal is to optimize $\mathcal{J}_{\mathcal{R}}(\boldsymbol{\theta})$ to find optimal values of parameters $\boldsymbol{\theta}$ minimizing the loss $\mathcal{L}$. For classification tasks, commonly used is the cross-entropy loss. This paper proposes a novel regularization formulation $\mathcal{R}$ that preserves a hierarchical structure (explained in Section IV-A).

When there are multiple layers, the set of parameters is denoted as a set of matrices $\mathbf{W}_i$, where each matrix is in $\mathbb{R}^{(d_{i-1}+1)\times p_i}$. The elements of $\mathbf{W}_i$ are denoted by $\mathbf{w}_j$, and each $\mathbf{w}_j$ is a weight vector in $\mathbb{R}^{d_{i-1}+1}$. The index j ranges from 1 to p_i, and i ranges from 1 to L, representing the layers in a neural network. In this paper, we are focusing on the weight parameters at the last layer L for the classification task.

[2]This spherical symmetry makes the normal distribution well-suited for scenarios where we want a distribution that is isotropic, meaning that it looks the same in all directions. In the context of sampling parameters for hyperspheres, using a normal distribution helps ensure that the sampled points are evenly distributed around the mean, contributing to a more uniform coverage on the hypersphere.

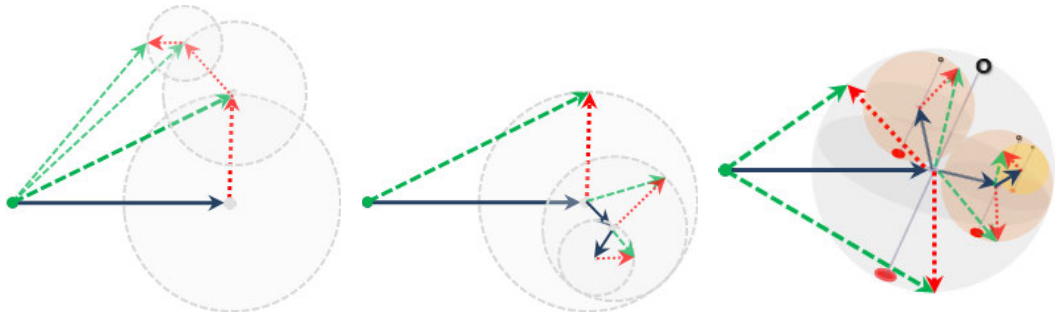


FIGURE 2. A series of levelwise (l) spheres with a radius (r_l) is shown: (a) A radius of an overall area converges to $\frac{r_0}{1-\delta}$ ($= \sum_l^\infty r_0\delta^l$: the sum of radius series) as l goes to infinity where l denotes a level, r_0 is their initial radius, and δ is the ratio between radiuses (r_l, r_{l-1}), $\frac{r_l}{r_{l-1}} < 1$. (b) The radius of the overall area is bounded to the initial radius r_0 of spheres. This bears a resemblance to the process of repeat of *Hypersphere packing* which arranges non-overlapping spheres within a containing space. (c) 2-sphere is defined following (b) which is appropriate to model a hierarchical structure. This can be generalized with a hypersphere ($\mathbb{S}^d$, $d \geq 3$) in a higher dimensional space.

A. HIERARCHICAL AND HYPERSPHERICAL HYPOTHESES

We introduce a projection parameter (weight vector) denoted as $\mathbf{w} \in \mathbb{R}^{d+1}$ to perform transformations on a vector $\mathbf{v} \in \mathbb{R}^{d+1}$, resulting in an embedding space defined by a Euclidean metric. The transformation can be represented as $\mathbf{v} \mapsto \mathbf{w}^T\mathbf{v} \in \mathbb{R}^{d+1}$. To ensure that $\mathbf{w}$ resides on a d-sphere, we impose the constraint $\|\mathbf{w}\| = 1$, where $\|\cdot\|$ denotes the Euclidean norm (l_2-norm).

Geometrically, we redefine the parameter vector $\mathbf{w}$ as the difference between two vectors: $\mathbf{w} := \mathbf{w}_{\text{surface}} - \mathbf{w}_{\text{center}}$, where $\mathbf{w}_{\text{surface}} \in \mathbb{R}^{d+1}$ and $\mathbf{w}_{\text{center}} \in \mathbb{R}^{d+1}$ represent the surface and center parameters, respectively. The parameter vector $\mathbf{w}$ lies on the d-sphere $\mathbb{S}^d(\mathbf{w}_{\text{center}})$.[3]

1) HIERARCHICAL PARAMETERIZATION WITH LEVELWISE AND GROUPWISE STRUCTURE

We consider a hierarchical structure comprising multiple levels and multiple groups, denoted as l and g_k respectively. The hierarchical parameters are defined using both levelwise and groupwise structures.

a: LEVELWISE STRUCTURE

The projection parameter $\mathbf{w}$ is defined at level l as follows:

$$\mathbf{w}_l := \mathbf{w}_{\text{surface},l} - \mathbf{w}_{\text{center},l} \tag{1}$$

where $\mathbf{w}_l$ lies on the d-sphere centered at $\mathbf{w}_{\text{center},l}$. In this paper, we work in a higher-dimensional space than shown in Fig. 2.

In the levelwise setting, we represent the surface and center parameters at level l additively, based on the center parameter from the previous level $l-1$: $\mathbf{w}_{\text{center},l} \leftarrow \mathbf{w}_{\text{center},l-1} + \mathbf{w}_{\text{connecting},l}$, where $\mathbf{w}_{\text{center},l-1}$ is the accumulated sum of connecting parameters from levels 1 to $l-1$ ($\mathbf{w}_{\text{center},l-1} = \sum_{i=1}^{l-1} \mathbf{w}_{\text{connecting},i}$), and $\mathbf{w}_{\text{connecting},l}$ denotes a connecting parameter from $\mathbf{w}_{\text{center},l-1}$ to $\mathbf{w}_{\text{center},l}$. To simplify the notation, we use $\mathbf{w}_{l,l-1}$ to represent $\mathbf{w}_{\text{connecting},l}$, and express the center vector at level l as $\mathbf{w}_{\text{center},l} := \mathbf{w}_{\text{center},l-1} + \mathbf{w}_{\text{center},l,l-1}$, and the surface vector as $\mathbf{w}_{\text{surface},l} := \mathbf{w}_{\text{center},l-1} + \mathbf{w}_{\text{surface},l,l-1}$. Both the center and surface parameters at the current level depend on the center parameter at the previous level. Consequently, Eq. 1 can be equivalently expressed as

$$\mathbf{w}_l := \mathbf{w}_{\text{surface},l,l-1} - \mathbf{w}_{\text{center},l,l-1}. \tag{2}$$

We use the subscript $l, l-1$ to indicate parameters at level l connected from the center parameter at level $l-1$. The levelwise parameters $\mathbf{w}_{\text{surface},l,l-1}$ and $\mathbf{w}_{\text{center},l,l-1}$ are later used to define the groupwise structure.

b: GROUPWISE STRUCTURE

Using the group notation g_k, we can rewrite the center parameter $\mathbf{w}_{\text{center},l,l-1}$ in Eq. 2 as $\mathbf{w}_{\text{center}}(g_k)$. The d-sphere of group g_k is defined as $\mathbb{S}^d(\mathbf{w}_{\text{center}}(g_k))$. Each group forms a group set with a levelwise setting, denoted as $\mathcal{G}_l := \{g_k\}_{k=1}^{|\mathcal{G}_l|}$, where $\mathcal{G}_l \subseteq \mathcal{Y}_{l-1}$, and $\mathcal{Y}_{l-1}$ denotes the batch label set at level $l-1$. Note that the group set $\mathcal{G}_l$ at level l depends on the group set at level $l-1$, denoted as $\mathcal{G}_{l-1} := \{g_{k'}\}_{k'=1}^{|\mathcal{G}_{l-1}|}$, where $\mathcal{G}_{l-1} \subseteq \mathcal{Y}_{l-2}$. This relationship between group sets extends across levels. We provide an adjacency indication (or probability distribution) $\{0,1\}^{|\mathcal{Y}_l|\times|\mathcal{G}_l|}$ based on these groupwise relationships. Thus, the i-th parameter of group g_k on $\mathbb{S}^d(\mathbf{w}_{\text{center}}(g_k))$ is defined as follows:

$$\mathbf{w}(g_k, i) := \mathbf{w}_{\text{surface}}(g_k, i) - \mathbf{w}_{\text{center}}(g_k), \tag{3}$$

where $\mathbf{w}_{\text{surface}}(g_k, i)$ and $\mathbf{w}_{\text{center}}(g_k)$ on $\mathbb{S}^d(\mathbf{w}_{\text{center}}(g_{k'}))$ are calculated based on a center parameter $\mathbf{w}_{\text{center}}(g_{k'})$ at level $l-1$. Here, $i = 1, \ldots, |g_k|$, and $|g_k|$ denotes the number of surface parameter vectors in group g_k.

[3] We denote the d-sphere as $\mathbb{S}^d(\mathbf{w}_{\text{center}}) := \{\mathbf{w}_{\text{surface}} - \mathbf{w}_{\text{center}} \in \mathbb{R}^{d+1} : \|\mathbf{w}_{\text{surface}} - \mathbf{w}_{\text{center}}\| = 1\}$. Although we assume a radius of 1 for simplicity, the parameter vector can have a radius $r > 0$.

2) HIERARCHICAL-HYPERSPHERICAL REGULARIZATION

In this section, we define a regularization term using the hierarchical parameters described above. The regularization term $\mathcal{R}(\boldsymbol{\theta})$ is given as follows:

$$\mathcal{R}(\boldsymbol{\theta}) := \sum_{l} \lambda_l (\mathcal{R}_l(\mathbf{w}) + \mathcal{R}_l(\mathbf{w}_{\text{center}})) + \sum_{l} \mathcal{C}_l \quad (4)$$

where $\mathcal{R}_l(\mathbf{w})$ represents the regularization term for projection parameters $\mathbf{w}(g_k, i)\ \forall i$ within the same group g_k, $\mathcal{R}_l(\mathbf{w}_{\text{center}})$ represents the regularization term for center parameters $\mathbf{w}_{\text{center}}(g_k)\ \forall k$ across groups on $\mathbb{S}^d(\mathbf{w}_{\text{center}}(g_{k'}))$ in the same level l, $\lambda_l > 0$, and $\mathcal{C}_l$ applies geometry-aware constraints across spheres. $\mathcal{R}_l(\mathbf{w})$ and $\mathcal{R}_l(\mathbf{w}_{\text{center}})$ are defined as follows:

$$\mathcal{R}_l(\mathbf{w}) := \frac{1}{N} \sum_{\{g_k \in \mathcal{G}_l\}} \frac{2}{N_p} \sum_{\{i \neq j \in g_k\}} d(\mathbf{w}(g_k, i), \mathbf{w}(g_k, j)) \quad (5)$$

and

$$\mathcal{R}_l(\mathbf{w}_{\text{center}}) := \frac{2}{N_c} \sum_{\{g_i \neq g_j \in \mathcal{G}_l\}} d(\mathbf{w}_{\text{center}}(g_i), \mathbf{w}_{\text{center}}(g_j)) \quad (6)$$

where N denote the total number of groups in $\mathcal{G}_l$, defined as $N = |\{g_k \in \mathcal{G}_l\}|$. Additionally, we define N_p as the total number of pairwise combinations within each group g_k, given by $N_p = |\{i \neq j \in g_k\}|(|\{i \neq j \in g_k\}| - 1)$. Similarly, N_c represents the total number of pairwise combinations between different groups in $\mathcal{G}_l$, computed as $N_c = |\{g_i \neq g_j \in \mathcal{G}_l\}|(|\{g_i \neq g_j \in \mathcal{G}_l\}| - 1)$. In our context, $d(\cdot, \cdot)$ denotes the distance metric used to measure the difference between parameters, which is defined in Section IV-C.

When given a minibatch of inputs ($m_{\mathbf{x}}$: a set of inputs $\{\mathbf{x}_i\}$), the regularization term becomes: $\mathbb{E}[\mathcal{R}(\boldsymbol{\theta})] = \frac{1}{|m_{\mathbf{x}}|} \sum_{\mathbf{x}_i \in m_{\mathbf{x}}} \mathcal{R}(\boldsymbol{\theta}; \mathbf{x}_i)$. Here, $\mathcal{R}(\boldsymbol{\theta}; \mathbf{x}_i)$ represents the regularization term evaluated on the input $\mathbf{x}_i$ with the parameter set $\boldsymbol{\theta}$.

The constraint term $\mathcal{C}_l$ helps construct geometry-aware relational parameters between different spheres at the same level and across levels. Multiple constraints are defined as $\mathcal{C}_l := \sum_k \lambda_k \mathcal{C}_{l,k}$, where $\mathcal{C}_{l,k}$ is the k-th constraint between parameters at the l-th and the $(l-1)$-th level, and $\lambda_k > 0$ is a Lagrange multiplier. We omit this constraint term in this paper.

B. SELF-SUPERVISED HIERARCHICAL PARAMETERIZATION

In this section, we present the self-supervised method for calculating the adjacency indication which captures the groupwise relations over different levels (as discussed in Section IV-A1). The goal is to find superclasses (groups) based on the inferred confusion probabilities of pairwise categories, which reflect their uncertainty of discrimination.

1) GROUPWISE RELATIONS

Given the training data $\mathcal{D}$ along with their corresponding target labels $\mathcal{Y}_l$ at the l-th level, we create a group set $\{g_k\}_{k=1}^K$, where K represents the number of groups, and $\{g_k\}_{k=1}^K \subseteq \mathcal{G}_l \subseteq \mathcal{Y}_{l-1}$. Based on this group mapping, we can define the cross-level adjacency from $\mathcal{Y}_l$ to $\mathcal{Y}_{l-1}$.

2) INFERRED CONFUSION PROBABILITIES

Given a training set consisting of pairs $\{\mathbf{x}_i, y_i\}, i = 1 \ldots n$, where $\mathbf{x}_i \in \mathcal{D}$ is the input vector, and $y_i \in \{1, \ldots, K\} \subseteq \mathcal{Y}_l$ denotes the target label at the leaf level, with K being the number of target categories. From a discriminative viewpoint, we can infer the posterior probability $p(y|\mathbf{x}; \boldsymbol{\theta})$ over the target label using the learned parameters $\boldsymbol{\theta}$ of the deep networks.

The confusion probability is defined as the conditional (posterior) probability over labels other than the ground-truth label (y_i) for the given input vector ($\mathbf{x}_i$), i.e., $p(\tilde{y}|\mathbf{x}_i; \boldsymbol{\theta}), \forall \tilde{y} \in \{1, \ldots, K\} \setminus y_i$, where $|\{1, \ldots, K\} \setminus y_i| = K - 1$. If the confusion probability between pairwise categories is large, it indicates that the inference model either confuses a pair of categories or those categories are semantically similar. Based on this assumption, categories with high uncertainty are assigned to the same group.

3) CONSTRUCTION OF THE CONFUSION PROBABILITY MATRIX

We construct the confusion probability matrix $\mathbf{M}$ by calculating pairwise confusions over all given categories. Here, $\mathbf{M}(y_i, k)$ is given by $\sum_{i=1}^{n} \frac{1}{|k=y_i|} Pr(y_i, k)$ for all k. Here, $Pr(y_i, k) := p(y_i, k|\mathbf{x}_i; \boldsymbol{\theta})$, $Pr(k, k) = 0$, and $k = 1, \ldots, K$. Since the confusion probability matrix $\mathbf{M}$ is asymmetric, we symmetrize it by taking the average of the matrix and its transpose, i.e., $\frac{\mathbf{M}+\mathbf{M}^T}{2}$. Next, we focus on the pairs from the upper triangle part of this symmetrized matrix, excluding the diagonal elements, i.e., $\{Pr(i, j)|\ i < j\}$. Among the $\frac{(K-1)(K-2)}{2}$ elements representing pairs of different categories, we adopt a simple grouping strategy. Pairs of categories with a confusion probability larger than the threshold (e.g., the median value of the confusion distribution) are grouped together for $k = 1, \ldots, K$. Meanwhile, categories with confusion probabilities smaller than the threshold form individual groups.

4) INFERENCE NETWORK AND LEARNING

In deep neural networks, we calculate the confusion probability matrix using the *inference network* ($\mathcal{H} \sim \boldsymbol{\theta}$) at the leaf level, where $\mathcal{H}(\mathbf{x}_i) \mapsto \mathbb{R}^K$. Here, $\boldsymbol{\theta}$ represents the parameter set for $\mathcal{H}$, which includes W, $\mathbf{W}_L$, and other relevant parameters, and K denotes the number of categories at the leaf level. To calculate the confusion probability, we use a softmax function:

$$\sigma(\mathbf{x}) := P(y = j\,|\mathbf{x}) = \frac{\exp \mathcal{H}(\mathbf{x}; \mathbf{W}_L(j))}{\sum_k \exp \mathcal{H}(\mathbf{x}, \mathbf{W}_L(k)))} \quad (7)$$

Next, using the threshold-based grouping algorithm (explained in Algorithm 1), we segment the categories and define stochastic groups with the number of groups $\tilde{K}$. For improved learning effectiveness, we introduce an inference network for grouping ($\tilde{\mathcal{H}} \sim \boldsymbol{\phi}$), where $\tilde{\mathcal{H}}(\mathbf{x}_i) \mapsto \mathbb{R}^{\tilde{K}}$, and the

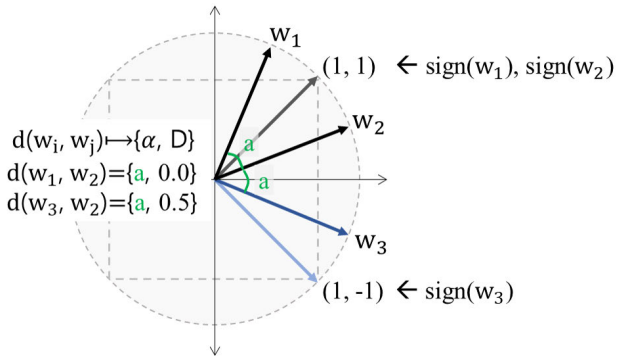


FIGURE 3. While angular distances α between a pair of vectors $\{\mathbf{w}_1, \mathbf{w}_2\}$ and $\{\mathbf{w}_2, \mathbf{w}_3\}$ have the same angle $\mathsf{a} > 0$, the discrete metric D between vectors are the different each other, 0.0 and 0.5. Thus, $d(\mathbf{w}_1, \mathbf{w}_2) < d(\mathbf{w}_2, \mathbf{w}_3)$ where a pair $\{\mathbf{w}_2, \mathbf{w}_3\}$ has a different sign of components on a horizontal axis but not that in a pair $\{\mathbf{w}_1, \mathbf{w}_2\}$.

parameter set for $\tilde{\mathcal{H}}$ is denoted as $\boldsymbol{\phi}$. The parameter sets for $\mathcal{H}$ and $\tilde{\mathcal{H}}$ are separate, but the parameter set W is shared between them. Thus, the inference involves both $\mathcal{H}$ and $\tilde{\mathcal{H}}$, denoted as $\{\mathcal{H}, \tilde{\mathcal{H}}\} \sim \{\boldsymbol{\theta}, \boldsymbol{\phi}\}$. The learning process is achieved through minimizing the loss function:

$$\arg\min_{\boldsymbol{\theta},\boldsymbol{\phi}} \mathcal{L}(\sigma(\mathcal{H}(\mathbf{x}_i)), y_i) + \lambda_g \mathcal{L}_g(\sigma(\tilde{\mathcal{H}}(\mathbf{x}_i)), y_i') \; \forall i \tag{8}$$

where $\mathcal{L}$ represents the cross-entropy loss, $\mathcal{L}_g$ is the cross-entropy based grouping loss, $\lambda_g > 0$, and $\boldsymbol{\theta} := \{\mathsf{W}, \mathbf{W}_L, \dots\}$ and $\boldsymbol{\phi} := \{\mathsf{W}, \mathbf{W}_L', \dots\}$ represent the parameter sets for $\mathcal{H}$ and $\tilde{\mathcal{H}}$ respectively. By utilizing the given training samples and their target labels at the leaf level, a hierarchical structure can be generated to a predefined depth. In a stochastic gradient optimization setting, the posterior inference and grouping are recursively repeated, incorporating a combination of grouping, classification, and regularization. This iterative process is expected to converge to a near-optimal state, where the parameters at individual levels are evenly distributed.

C. DISCRETE AND CONTINUOUS ANGULAR DISTANCE METRIC

The discrete metric aligns well with our groupwise definition introduced earlier. In Fig. 3, we observe that two pairs of parameters can exhibit different discrete distances while sharing the same continuous distances. This shows that the discrete metric captures unique aspects of the parameter distribution beyond the continuous distances.

To maximize the discrete distances between parameter pairs, it is beneficial to have their individual dimensions with different signs. This arrangement promotes an isolated and diverse distribution of parameters, leading to larger discrete distances. By encouraging diversity in the parameter space, the discrete metric can be leveraged effectively in optimization and learning scenarios, enhancing the learning process and enabling better generalization.

A discrete angular distance can be derived from a discrete product. The discrete product of a pair of parameter vectors $\mathbf{w}_i$ and $\mathbf{w}_j$ in $\mathbb{R}^{d+1}$ can be computed using the sign function as follows:

$$\mathsf{D} := \frac{1}{d+1} \sum_{k}^{d+1} \text{sign}(\mathbf{w}_i(k)) \cdot \text{sign}(\mathbf{w}_j(k)), \tag{9}$$

where $\text{sign}(w) := \begin{cases} 1, & \text{if } w \geq 0 \\ -1, & \text{otherwise,} \end{cases}$, $-1 \leq \mathsf{D} \leq 1$, and $\mathbf{w}(k) \in \mathbb{R}$ denotes the k-th element of $\mathbf{w}$. This is a normalized version of the Hamming distance. The discrete angular distance of the above product can be calculated using the arccosine function: $\alpha_{\mathsf{D}} = \frac{1}{\pi} \arccos \mathsf{D}$, where $0 \leq \alpha_{\mathsf{D}} \leq 1$. Eq. 9 is formulated based on a binary discrete product (referred to as D_2). For a ternary discrete product (referred to as D_3), the function $\text{sign}(x)$ with three terms is used, where $\text{sign}(x) \in \{-1, 0, 1\}$.

1) PROPOSED BLENDED METRICS

However, the discrete distance could underestimate the approximation of the model distribution due to its limited representation as the sign function is scale-invariant. To address this, we propose to blend the discrete distance metric ($\alpha_{\mathsf{D}} = \frac{1}{\pi} \arccos \mathsf{D}$) with a continuous angular distance metric ($\alpha = \frac{1}{\pi} \arccos\left(\frac{\mathbf{w}_i \cdot \mathbf{w}_j}{|\mathbf{w}_i||\mathbf{w}_j|}\right)$, $0 \leq \alpha \leq 1$) into a single metric. We use Pythagorean means, which consist of the arithmetic mean (A), the geometric mean (G), and the harmonic mean (H), to combine the two distance metrics. Pythagorean means using a pair of angular distances are defined as follows:

$$\begin{aligned} A(\alpha_{\mathsf{D}}, \alpha) &:= \frac{\alpha_{\mathsf{D}} + \alpha}{2}, \\ G(\alpha_{\mathsf{D}}, \alpha) &:= \sqrt{\alpha_{\mathsf{D}} \alpha}, \\ H(\alpha_{\mathsf{D}}, \alpha) &:= \frac{2\alpha_{\mathsf{D}} \alpha}{\alpha_{\mathsf{D}} + \alpha}. \end{aligned} \tag{10}$$

In the angular distance[4] using a pair of angles $\{\alpha_{\mathsf{D}}, \alpha\}$, we adopt a reversed form $1 - d(\alpha_{\mathsf{D}}, \alpha)$ to maximize an angle in the minimization formulation, instead of using $(\cdot)^{-s}$ where $s = 1, 2, \dots$, as used in the Thomson problem that utilizes s-energy [25]. The cosine similarity using the pair of angles is defined as follows:

$$\begin{aligned} \cos A &:= \cos\left(A(\alpha_{\mathsf{D}}, \alpha)\pi\right) = \cos\left(\frac{\alpha_{\mathsf{D}} + \alpha}{2}\pi\right), \\ \cos G &:= \cos\left(G(\alpha_{\mathsf{D}}, \alpha)^2 \pi\right) = \cos\left(\alpha_{\mathsf{D}} \alpha \pi\right), \\ \cos H &:= \cos\left(H(\alpha_{\mathsf{D}}, \alpha)\pi\right) = \cos\left(\frac{2\alpha_{\mathsf{D}}\alpha}{\alpha_{\mathsf{D}} + \alpha}\pi\right), , \end{aligned} \tag{11}$$

and these cosine similarity functions are normalized with $\frac{\cos(\cdot)+1}{2}$ to have a value within the range $[0, 1]$. Finally, Pythagorean means of cosine similarities can be calculated

[4] In $0 \leq \alpha \leq 1$, the angle and its cosine value have an inverse relationship: $0 \leq \alpha \leq 1 \rightarrow 1 \geq \cos(\alpha\pi) \geq -1$.

Algorithm 1 Self-Supervised Grouping Algorithm

Input: Data $\mathbf{x}_i$, label y_i, $i = 1, \ldots, n$
Initialize $\{\mathsf{W}, \mathbf{W}_L\} \sim \mathcal{N}(0, 1)$
repeat
  # *Confusion probability matrix calculation*
  $\mathbf{M} \leftarrow \sum_{i=1}^{n} P(y_i, k)/|k = y_i| \; \forall k$
  $\mathbf{M} \leftarrow$ Upper-Triangle(Symmetric($\mathbf{M}$)),
  # *Find clusters*
  $\mathcal{I} := \{(j, m) \mid (\mathbf{M} > \tau)\}$ *: an index set indicating high confusion*,
  $\tau \leftarrow$ median($\mathbf{M}$)*: threshold for selection*,
  $\mathcal{G} \leftarrow \{\}$*: a group list*, $k' \leftarrow 0$*: the number of superclasses*, η*: maximum number*
  for $(j, m) \in \mathcal{I} \; \forall j, m$ **do**
    if $j \notin \mathcal{G}$ **then**
      $\mathcal{G}[k'].append(j)$
    end if
    if $(m \notin \mathcal{G})$ **then**
      $\mathcal{G}[k'].append(m)$ ($\leftarrow \mathcal{G}[k'].size() \leq \eta$ *condition can be added*)
    end if
    $k' \leftarrow k' + 1$
  end for
  # Assign clusters for remained classes
  for $(k \notin \mathcal{G}) \; \forall k$ **do**
    $\mathcal{G}[k'].append(k)$
    $k' \leftarrow k' + 1$
  end for
  $K' \leftarrow k'$ *: The number of superclasses*
  $y_i' \leftarrow \mathcal{G}(y_i)$ *: Assign superclass label*
  Initialize $\mathbf{W}_L' \sim \mathcal{N}(0, 1)$
  Training $\arg\min_{\bar{\mathsf{W}}} \mathcal{L} + \lambda_g \mathcal{L}_g$, $\bar{\mathsf{W}} := \{\mathsf{W}, \mathbf{W}_L, \mathbf{W}_L'\}$
until Max #epochs

as follows:

$$A := \frac{\cos \alpha_{\mathsf{D}} \pi + \cos \alpha \pi + 2}{4},$$
$$G := \frac{(\cos \alpha_{\mathsf{D}} \pi + 1)(\cos \alpha \pi + 1)}{4},$$
$$H := \frac{(\cos \alpha_{\mathsf{D}} \pi + 1)(\cos \alpha \pi + 1)}{\cos \alpha_{\mathsf{D}} + \cos \alpha + 2}. \tag{12}$$

The metric functions defined in (10) and its variants satisfy metric conditions: non-negativity, symmetry, and triangle inequality. The distance using the above metric function between any two parameter points is bounded because the hypersphere is a compact manifold. These proposed blended metrics (A, G, and H) are used in the experimental section.

D. GRADIENT AND BACKPROPAGATION

As the sign function is not differentiable at the value 0, we use the straight-through estimator (STE) [26] in the backward pass of the neural networks for the sign function in the discrete metric. The derivative of the sign function is substituted with $1_{|w| \leq 1}$ in the backward pass, known as the saturated STE. Similarly, as the derivative of $\arccos(x)$ ($\frac{-1}{\sqrt{1-x^2}}$) is undefined at the value $x = \pm 1$, we apply clamping to the cosine function to restrict $x \in [-0.99, 0.99]$, where $x = \cos(\alpha\pi)$, and $0 \leq \alpha \leq 1$.

V. EXPERIMENTS

A. EXPERIMENTAL SETUP

1) DATASETS

We conducted experiments on the proposed method using four publicly available datasets such as CUB200-2011 [27], Stanford-Cars [28], CIFAR-100 [29], and TinyImageNet [30]. For CUB200-2011 and Stanford-Cars, we use shorten names, CUB200 and Cars respectively, hereafter. CUB200 and Cars datasets are for fine-grained visual categorization (recognizing bird species or car models). CIFAR-100 and TinyImageNet datasets are used for object classification. The fine-grained visual categorization datasets show low inter-class variances, but not in object classification datasets. For CUB200 and CIFAR-100, we experimented using given superclass labels which are defined by a human annotator to compare with that using superclasses defined by our self-supervised method. Statistics of datasets in detail is provided in Table 5 in Appendix.

TABLE 1. Test accuracy (%) along with different metrics for regularization on CUB200 using ResNet-18. Plain (Baseline) without additional regularization shows 71.95 % accuracy. The top three accuracy values are presented in bold.

	Metrics of regularization on the last layer							
	U-Euc2	Ang2	Cos	D_2	D_3	A	G	H
Plain (Baseline)	71.90	71.35	71.88	71.69	71.69	72.59	72.19	71.98
Supervised Hierarchy (Ours)	75.85	76.23	76.23	76.66	76.78	76.73	**77.06**	**77.06**
Self-Supervised Hierarchy (Ours)	76.59	76.73	76.01	76.64	76.64	**77.02**	76.68	76.68

TABLE 2. Test accuracy (%) along with different metrics for regularization on Cars using ResNet-18. Plain (Baseline) without additional regularization shows 87.05 % accuracy. The top three accuracy values are presented in bold.

	Metrics on the last layer							
	U-Euc2	Ang2	Cos	D_2	D_3	A	G	H
Plain (Baseline)	87.70	87.26	87.98	87.17	87.17	87.36	87.31	87.20
Self-Supervised Hierarchy (Ours)	88.06	87.91	88.12	88.18	**88.48**	**88.58**	88.45	**88.57**

TABLE 3. Test accuracy (%) along with different metrics for regularization on CIFAR-100 using ResNet-18 (without pre-training). Plain (Baseline) without additional regularization shows 70% accuracy. The top three accuracy values are presented in bold.

	Metrics on the last layer							
	U-Euc2	Ang2	Cos	D_2	D_3	A	G	H
Plain (Baseline)	70.27	70.18	70.34	70.76	70.58	70.54	70.45	70.10
Supervised Hierarchy (Ours)	70.58	**70.77**	69.88	69.80	69.74	70.64	70.67	70.07
Self-Supervised Hierarchy (Ours)	69.96	70.24	69.92	70.08	70.35	**70.90**	**70.97**	70.39

TABLE 4. Test accuracy (%) along with different metrics for regularization on TinyImageNet using ResNet-18. Plain (Baseline) without additional regularization shows 64.54 % accuracy. The top three accuracy values are presented in bold.

	Metrics on the last layer							
	U-Euc2	Ang2	Cos	D_2	D_3	A	G	H
Plain (Baseline)	64.30	63.68	64.23	64.46	64.46	64.81	64.73	64.36
Self-Supervised Hierarchy (Ours)	64.34	64.29	64.96	65.38	65.35	**65.86**	**65.77**	**65.89**

2) DEEP NEURAL NETWORK MODELS AND TRAINING SETTING

We used the deep residual neural network (*ResNet*) [31]. For the datasets, CUB200, Cars, and TinyImageNet, where an input size is 224 × 224, the original ResNet[5] (ResNet-18 and ResNet-50 shown in Appendix) is used. For CIFAR-100, to fit their small size input (32 × 32 pixels) to ResNet, a smaller kernel size (3 instead of 7) at the first convolutional layer and a smaller stride (1 instead of 2) at the first block than that from the original are used. Consequently, the pretrained parameters are not used due to their network modification.

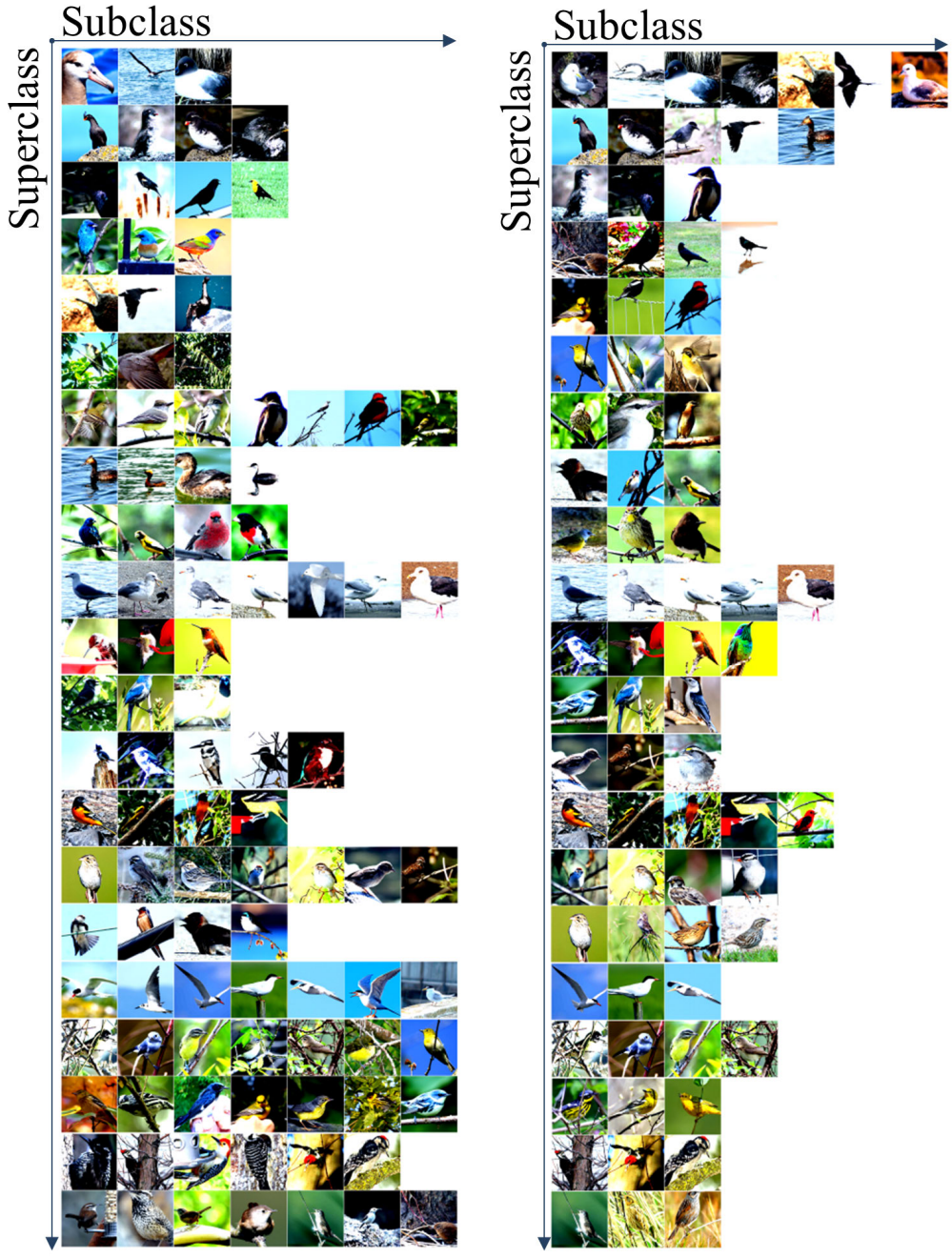
In training of the deep neural network, the hierarchical structure between a superclass and a subclass using the self-supervised grouping algorithm in Section IV-B were searched over all training samples. We used the stochastic gradient descent (SGD) optimization with a minibatch for training of the deep neural network. Even though the global hierarchical structure is defined over training sample distribution, a stochastic or partial hierarchical structure can used if partial labels are shown within each mini-batch. Mini-batches, 512, were used in the SGD optimizer. Even though the SGD is known as an unbiased estimation, a stochastic hierarchical structure could affect the overall approximation performance upon the class distribution within the mini-batch. We applied the hierarchical regularization in the FC layer of the ResNet. Settings in more detail are provided in Appendix.

[5]follows a model defined at the pytorch library.

3) BASELINE AND PROPOSED METHOD SETTING

We designate the configuration without hierarchical regularization on the last layer of deep neural networks as the 'Plain (baseline)'. In our proposed methods ('ours'), we incorporate hierarchical regularization (Eq. 4) on the last layer. Within 'ours,' we employ two hierarchy construction strategies: one utilizing given labels ('Supervised') and the other employing labels obtained through the proposed self-supervised method ('Self-Supervised').

For hierarchical regularization, we compare the proposed blended metrics (A, G, H) with the discrete metrics (D_2, D_3)



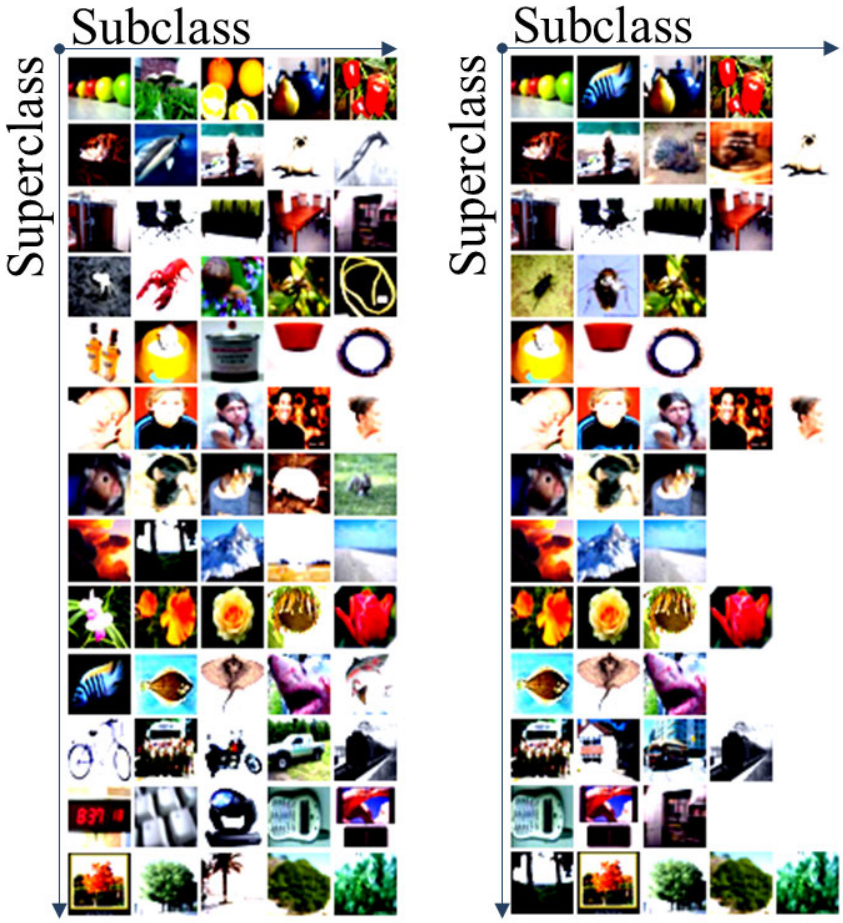
(a) Supervised (b) Self-Supervised

FIGURE 4. Grouped images of CUB200 dataset are shown. Each row shows images from a superclass which contains maximum seven and minimum three subclasses (21 superclasses are shown). (a) Supervised (human annotation) categorization shown over 70 in total. (b) the proposed self-supervised categorization show over 72 in total found where their test accuracy is 76.33 % shown. An individual image consists of 224 × 224 pixels.

and other continuous metrics such as Euclidean distance with unit-length projection ('U-Euc2'), angular distance ('Ang2'), and cosine similarity ('Cos') which are commonly used in the related works [1], [2]. The continuous metrics with order 2 (in U-Euc2 and Ang2), derived from Riesz s-energy, exhibit higher accuracy and are defined as follows:

- U-Euc2: $\sum_{i\neq j} \| \frac{\mathbf{w}_i}{\|\mathbf{w}_i\|} - \frac{\mathbf{w}_i}{\|\mathbf{w}_i\|} \|^{-2}$,
- Ang2: $\sum_{i\neq j} \arccos\left(\frac{\mathbf{w}_i \cdot \mathbf{w}_j}{\|\mathbf{w}_i\|\|\mathbf{w}_j\|}\right)^{-2}$,
- Cos: $\sum_{i\neq j} \frac{\mathbf{w}_i \cdot \mathbf{w}_j}{\|\mathbf{w}_i\|\|\mathbf{w}_j\|}$.

The Hamming distance-based angular metric described in Eq. 9 is defined as a discrete angular distance with the following formulations:



(a) Supervised (b) Self-Supervised

FIGURE 5. Grouped images of CIFAR-100 dataset are shown. Each row shows images from an individual superclass which contains minimum three subclasses (13 superclasses are shown). (a) Supervised categorization over 20 superclasses in total. (b) The proposed Self-supervised categorization over 48 superclasses are found where their test accuracy is 70.72 % shown. We rearrange an order of rows with similar images to be located in the same row. Each individual image consists of 32 × 32 pixels.

- $\alpha_{\mathsf{D}_2} = \frac{1}{\pi} \arccos \mathsf{D}_2$,
- $\alpha_{\mathsf{D}_3} = \frac{1}{\pi} \arccos \mathsf{D}_3$,

where D_2 is a binary discrete product with $\{-1, 1\}$, and D_3 is a ternary discrete product with $\{-1, 0, 1\}$. For simplicity, we use the notations D_2 and D_3 to represent the discrete angular metric, rather than α_{D_2} and α_{D_3}.

For both 'baseline and 'ours', we apply two types of regularization: weight decay (l^2) and energy minimization (E) [4]. Weight decay is equivalent to an l^2-norm constraint in the SGD setting [32], represented as $\lambda_f \sum_k \|\mathbf{w}_k\|$, where $\mathbf{w}_k \in \mathbf{W}$ and $\lambda_f > 0$. Energy minimization (E) [4] imposes a constraint that maximizes angular distance [1], [4] (denoted as 'Energy' minimization), expressed as $\lambda_e \sum_{i \neq j} d(\mathbf{w}_i, \mathbf{w}_j)$, where $d(\cdot, \cdot)$ is a pairwise distance, and $\lambda_e > 0$. The performance differences between cases with E regularization are detailed in the Appendix.

B. RESULTS

In this section, we present experimental results for last-layer regularization using various metrics, including continuous (U-Euc2, Ang2, and Cos), discrete (D_2 and D_3), and proposed blended metrics (A, G, and H). Due to space limit, we show some results including a test using ResNet-50 in Appendix.

1) FINE-GRAINED CATEGORIZATION

In this experiment, we used fine-grained image datasets. One is with birds (CUB200) and another is with cars (Cars) that focus on single species of objects. A hierarchy (superclass) of CUB200 is defined by the academic expert on birds (70 superclasses over 200 subclasses in total).

As shown in Table 1, the proposed self-supervised hierarchical regularization significantly improved the test

accuracy over all metrics on ResNet-18. This is comparable with the proposed hierarchical regularization using the supervised superclass label. Compared to CUB200, as shown in Table 2, the improvement of the proposed method is not that significant using Cars. Cars dataset includes unclear superclass categorization (e.g. Sedan, Coupe, Wagon, and so on). Moreover, even though the discriminative appearance to categorize is shown on rearmost part mostly, the images show random parts of cars. Our proposed blended metric-based methods showed better performance than that of other cases.

2) OBJECT CLASSIFICATION

We present the test accuracy (%) of the compared methods across different metrics for regularization on the last layer using CIFAR-100 in Table 3 and TinyImageNet in Table 4, respectively. As CIFAR-100 dataset provides the given superclass labels (20 superclasses over 100 subclasses in total), we compare the performance of the networks using those supervised (grondtruth) labels and our proposed self-supervised labels.

As shown in Table 3, the regularization with the proposed metrics (A, G, and H) outperforms other cases on the CIFAR-100 dataset. Compared to the baseline, our approach, when utilizing certain continuous metric-based methods and supervised label-based methods, exhibits degraded performance. The use of supervised labels does not seem to contribute significantly to the learning of the classification objective on this dataset. Interestingly, classification performance improved when employing self-supervised labels compared to supervised labels. This observation suggests that the hierarchy of objects may be based on factors such as function rather than the appearance of the object, which is directly associated with visual classification. The discrete angular metric (D_2 and D_3) and blended based regularization (A, G, H) based regularization demonstrates enhanced generalization performance in terms of test accuracy compared to other continuous metrics such as U-Euc2, Ang2, and Cos.

As shown in Table 4, the compared methods show a similar trend to that observed in CIFAR-100. Notably, the proposed hierarchical regularization, based on the blended metrics, demonstrates a relatively substantial improvement in performance compared to the other cases.

3) ABLATION STUDY: SUPERCLASS-SUBCLASS PAIRS IDENTIFIED BY THE PROPOSED SELF-SUPERVISED METHOD

Since the generalization performances using the supervised labels and the labels found by the proposed self-supervised method are comparable (as shown in Table 1 and Table 3), we qualitatively examined images with pairs of superclass and subclass. In Fig. 4 and Fig. 5, each row displays a set of subclass images from the same superclass based on (a) supervised labels and (b) self-supervised labels. To facilitate easy comparison, we arranged the rows containing subclass images found by the self-supervised method near rows from supervised labels where images are visually similar. Notably, our self-supervised grouping method identified a set of superclass-subclass pairs that showed similar images, particularly on the CIFAR-100 dataset.

VI. CONCLUSION

In this paper, we proposed a novel diversity promoting regularization method aimed at maximizing pairwise distances between parameters while preserving their hierarchical structure. Within a deep learning framework, we redefined the topology space by employing a hierarchical parameterization using multiple hyperspheres. This hierarchical structure was established in a self-supervised manner. On the hypersphere, each projection was parameterized by a surface parameter and a center parameter. To accommodate the multiple separated spaces, we devised a discrete metric combined with a continuous metric, aligning with their isolated property. The projection parameters were enforced to be evenly distributed on individual hyperspheres with equidistance constraints.

Through extensive experiments on publicly available datasets, including CUB200-2011, Stanford-Cars, CIFAR-100, and TinyImageNet, our proposed method demonstrated notable improvements in classification performance for deep neural networks. As a potential avenue for future exploration, our approach could be integrated with hierarchical representation learning techniques, such as hyperbolic (or Poincaré) embeddings [17], [18]. By combining these methods, we anticipate further enhancements and novel insights into hierarchical model representations.

APPENDIX

A. DATASET ACQUISITION DETAILS

CIFAR-100 and CUB200 datasets include a given pair of superclass and subclass labels. For CUB200, labels of a superclass can be extracted from their filenames following [27]. Table 5 shows statistics of benchmark datasets used in the experimental sections.

B. DEEP NEURAL NETWORK MODELS AND TRAINING DETAILS

We used ResNet which consists of the basic blocks or the bottleneck blocks with output channels [64, 128, 256, 512] in Conv. layers. A dimensionality of an input vector to the FC layer is 512.

Parameters in our proposed method using ResNet are optimized using the SGD with several settings: we fixed 1) the weight initialization with *Random-Seed* number '0' in pytorch, 2) learning rate schedule [0.1, 0.01, 0.001], 3) with momentum 0.9, 4) regularization: weight decay with $\lambda_f = 0.0005$, energy $\lambda_e = 0.1$, and hierarchical $\lambda_l = 0.1$, and 5) grouping loss with $\lambda_g = 0.1$. A bias term in the FC layer is not used. The images (CUB200, Cars, and TinyImageNet) in training and test sets are resized to 256×256 size. Then, the image is cropped with 224×224 size at random location in training and at center location in test. Horizontal flipping

TABLE 5. Statistics of benchmark datasets.

Dataset	#classes	#train	#test	input size (cropped)	#superclasses
CIFAR-100	100	50,000	10,000	32×32	20
CUB200	200	5,994	5,794	224×224	70
Cars	196	8,144	8,041	224×224	n.a.
TinyImageNet	200	100,000	10,000	224×224	n.a.

TABLE 6. Test accuracy (%) along with different metrics for regularization on CUB200 using ResNet-50. Plain (Baseline) without additional regularization shows 78.28 accuracy. The top three accuracy values are presented in bold.

	Metrics on the last layer							
	U-Euc2	Ang2	Cos	D_2	D_3	A	G	H
Plain (Baseline)	77.73	77.23	77.23	77.96	77.96	78.20	78.27	77.89
Supervised Hierarchy (Ours)	82.48	82.11	82.20	82.18	82.63	82.63	82.72	82.63
Self-Supervised Hierarchy (Ours)	82.70	82.18	82.68	82.63	**82.91**	**82.93**	**82.91**	82.82

TABLE 7. Test accuracy (%) along with different metrics for regularization on Cars using ResNet-50. Plain (Baseline) without additional regularization shows 88.88 accuracy. The top three accuracy values are presented in bold.

	Metrics on the last layer							
	U-Euc2	Ang2	Cos	D_2	D_3	A	G	H
Plain (Baseline)	88.72	88.18	88.57	88.62	88.62	88.77	88.96	88.58
Self-Supervised Hierarchy (Ours)	89.08	88.73	**89.40**	88.67	88.73	**89.16**	**89.16**	**89.18**

TABLE 8. Test accuracy (%) along with different metrics for regularization on CUB200 using ResNet-18. Plain (Baseline) without additional regularization shows 71.95 % accuracy. The top three accuracy values are presented in bold.

		Metrics on the last layer							
	$\{l^2, E\}$	U-Euc2	Ang2	Cos	D_2	D_3	A	G	H
Supervised Hierarchy (Ours)	$\{l^2\}$	76.70	76.45	76.63	76.49	76.44	**77.06**	**77.45**	76.92
Supervised Hierarchy (Ours)	$\{l^2, E\}$	75.85	76.23	76.23	76.66	76.78	76.73	**77.06**	**77.06**
Self-Supervised Hierarchy (Ours)	$\{l^2\}$	76.09	75.95	76.56	76.33	76.33	76.73	76.40	76.92
Self-Supervised Hierarchy (Ours)	$\{l^2, E\}$	76.59	76.73	76.01	76.64	76.64	**77.02**	76.68	76.68

TABLE 9. Test accuracy (%) along with different metrics for regularization on Cars using ResNet-18. Plain (Baseline) without additional regularization shows 87.05 % accuracy. The top three accuracy values are presented in bold.

		Metrics on the last layer							
	$\{l^2, E\}$	U-Euc2	Ang2	Cos	D_2	D_3	A	G	H
Self-Supervised Hierarchy (Ours)	$\{l^2\}$	88.13	88.17	88.16	88.08	88.08	**88.54**	88.37	88.44
Self-Supervised Hierarchy (Ours)	$\{l^2, E\}$	88.06	87.91	88.12	88.18	88.48	**88.58**	88.45	**88.57**

TABLE 10. Test accuracy (%) along with different metrics for regularization on CIFAR-100 using ResNet-18 (without pre-training). Plain (Baseline) without additional regularization shows 70% accuracy. The top three accuracy values are presented in bold.

		Metrics on the last layer							
	$\{l^2, E\}$	U-Euc2	Ang2	Cos	D_2	D_3	A	G	H
Supervised Hierarchy (Ours)	$\{l^2\}$	69.01	69.65	68.77	69.19	69.74	70.03	69.96	69.89
Supervised Hierarchy (Ours)	$\{l^2, E\}$	70.58	**70.77**	69.88	69.80	69.74	70.64	70.67	70.07
Self-Supervised Hierarchy (Ours)	$\{l^2\}$	70.18	70.17	70.33	70.15	70.70	70.53	70.71	70.28
Self-Supervised Hierarchy (Ours)	$\{l^2, E\}$	69.96	70.24	69.92	70.08	70.35	**70.90**	**70.97**	70.39

TABLE 11. Test accuracy (%) along with different metrics for regularization on TinyImageNet using ResNet-18. Plain (Baseline) without additional regularization shows 64.54 % accuracy. The top three accuracy values are presented in bold.

		Metrics on the last layer							
	$\{l^2, E\}$	U-Euc2	Ang2	Cos	D_2	D_3	A	G	H
Self-Supervised Hierarchy (Ours)	$\{l^2\}$	64.52	65.57	64.56	64.34	64.52	64.71	65.01	64.91
Self-Supervised Hierarchy (Ours)	$\{l^2, E\}$	64.34	64.29	64.96	65.38	65.35	**65.86**	**65.77**	**65.89**

TABLE 12. Test accuracy (%) along with different metrics for regularization on CUB200 using ResNet-50. Plain (Baseline) without additional regularization shows 78.28 % accuracy. The top three accuracy values are presented in bold.

		Metrics on the last layer							
	$\{l^2, E\}$	U-Euc2	Ang2	Cos	D_2	D_3	A	G	H
Supervised Hierarchy (Ours)	$\{l^2\}$	82.68	82.41	82.29	82.58	**82.96**	82.81	82.30	82.74
Supervised Hierarchy (Ours)	$\{l^2, E\}$	82.48	82.11	82.20	82.18	82.63	82.63	82.72	82.63
Self-Supervised Hierarchy (Ours)	$\{l^2\}$	82.81	82.34	**82.93**	82.75	82.86	82.75	**82.96**	**82.91**
Self-Supervised Hierarchy (Ours)	$\{l^2, E\}$	82.70	82.18	82.68	82.63	**82.91**	**82.93**	**82.91**	82.82

TABLE 13. Test accuracy (%) along with different metrics for regularization on Cars using ResNet-50. Plain (Baseline) without additional regularization shows 88.88 % accuracy. The top three accuracy values are presented in bold.

		Metrics on the last layer							
	$\{l^2, E\}$	U-Euc2	Ang2	Cos	D_2	D_3	A	G	H
Self-Supervised Hierarchy (Ours)	$\{l^2\}$	88.70	88.59	**89.18**	88.68	88.60	88.91	89.15	89.01
Self-Supervised Hierarchy (Ours)	$\{l^2, E\}$	88.06	87.91	88.12	88.18	88.48	**88.58**	88.45	**88.57**

is applied in training. While ResNet model for CIFAR-100 is trained from scratch without the pretrained weights for 300 epochs, other cases are trained using pretrained model provided by pytorch library[6] with 100 epochs. The learning rate decay by 0.1 at [150, 225] epochs from an initial value of 0.1 if without the pretrained weights, and at [30, 60] epochs from an initial value of 0.01 otherwise. The experiments are conduced using GPU "NVIDIA TESLA P40". We used one GPU for ResNet-18, and four GPUs for ResNet-50.

C. ADDITIONAL RESULTS ON RESNET-50

We show test accuracy (%) of the compared methods along different metrics using ResNet-50 further using fine-grained image datasets (CUB200 and Cars). In the SGD optimizer, we used 128 of a minibatch size in four GPUs.

As shown in Table 6, the proposed self-supervised hierarchical regularization significantly improved the test accuracy over all metrics on ResNet-50 similar to that of ResNet-18. As shown in Table 7, our proposed self-supervised regularization showed improved generalization performance compared to the baseline method without hierarchical regularization.

D. ABLATION STUDY: IMPACT OF ANGULAR DISTANCE-BASED (ENERGY MINIMIZATION) REGULARIZATION

In line with the experiments detailed in the main text, we consistently employed angular distance-based regularization [4] in conjunction with weight decay (l^2). This section specifically explores the effects of angular distance-based regularization, referred to as energy minimization (E) regularization, and compares its impact to the simpler case utilizing l^2-norm-based regularization (i.e., weight decay). Our evaluation encompasses two scenarios: one with only weight decay (l^2) and another with weight decay combined with energy minimization regularization (l^2, E), applied to parameters across all layers except the last layer.

As depicted in the tables (Table 8, Table 9, Table 10, Table 11, Table 12, and Table 13), the inclusion of additional regularization (E) applied to all layers except the last tends to improve test accuracy. Notably, when our proposed metrics are implemented in the last layer, they consistently outperform the other cases. Accuracy values with gray color indicate the values shown in Table 1, Table 2, Table 3, Table 4, Table 6, and Table 7, respectively.

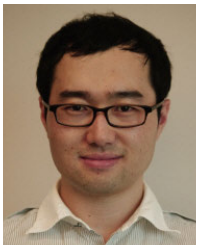
[6] from https://download.pytorch.org/models/resnet18-5c106cde.pth

REFERENCES

[1] P. Xie, W. Wu, Y. Zhu, and E. P. Xing, "Orthogonality-promoting distance metric learning: Convex relaxation and theoretical analysis," in *Proc. 35th Int. Conf. Mach. Learn. (ICML)*. Stockholm, Sweden: Stockholmsmässan, Jul. 2018, pp. 5399–5408.

[2] P. Xie, Y. Deng, Y. Zhou, A. Kumar, Y. Yu, J. Zou, and E. P. Xing, "Learning latent space models with angular constraints," in *Proc. 34th Int. Conf. Mach. Learn. (ICML)*, Sydney, NSW, Australia, Aug. 2017, pp. 3799–3810.

[3] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, 1995.
[4] W. Liu, R. Lin, Z. Liu, L. Liu, Z. Yu, B. Dai, and L. Song, "Learning towards minimum hyperspherical energy," in *Proc. 32nd Int. Conf. Neural Inf. Process. Syst. (NIPS)*. Red Hook, NY, USA: Curran Associates, 2018, pp. 6225–6236.
[5] P. Xie, A. Singh, and E. P. Xing, "Uncorrelation and evenness: A new diversity-promoting regularizer," in *Proc. 34th Int. Conf. Mach. Learn. (ICML)*, vol. 70, 2017, pp. 3811–3820.
[6] M. Cogswell, F. Ahmed, R. B. Girshick, L. Zitnick, and D. Batra, "Reducing overfitting in deep networks by decorrelating representations," in *Proc. Int. Conf. Learn. Represent.*, 2016.
[7] R. Kurzweil, *How to Create a Mind: The Secret of Human Thought Revealed*. New York, NY, USA: Penguin Books, 2013.
[8] N. Verma, D. Mahajan, S. Sellamanickam, and V. Nair, "Learning hierarchical similarity metrics," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 2280–2287.
[9] L. Du, Z. Lu, Y. Wang, G. Song, Y. Wang, and W. Chen, "Galaxy network embedding: A hierarchical community structure preserving approach," in *Proc. 27th Int. Joint Conf. Artif. Intell. (IJCAI)*, Jul. 2018, pp. 2079–2085.
[10] L. Tu, *An Introduction to Manifolds*. New York, NY, USA: Springer, 2010.
[11] Z. Gong, P. Zhong, and W. Hu, "Diversity in machine learning," *IEEE Access*, vol. 7, pp. 64323–64350, 2019.
[12] D. Yang, S. Hong, Y. Jang, T. Zhao, and H. Lee, "Diversity-sensitive conditional generative adversarial networks," in *Proc. Int. Conf. Learn. Represent.*, 2019.
[13] N. Li, Y. Yu, and Z.-H. Zhou, "Diversity regularized ensemble pruning," in *Proc. 2012th Eur. Conf. Mach. Learn. Knowl. Discovery Databases (ECMLPKDD)*, 2012, pp. 330–345.
[14] N. Ratzlaff and L. Fuxin, "HyperGAN: A generative model for diverse, performant neural networks," in *Proc. 36th Int. Conf. Mach. Learn.*, vol. 97, Jun. 2019, pp. 5361–5369.
[15] T. Zhou, S. Wang, and J. A. Bilmes, "Diverse ensemble evolution: Curriculum data-model marriage," in *Advances in Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates, 2018, pp. 5905–5916.
[16] W. Liu, Y.-M. Zhang, X. Li, Z. Yu, B. Dai, T. Zhao, and L. Song, "Deep hyperspherical learning," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., 2017, pp. 3950–3960.
[17] M. Nickel and D. Kiela, "Poincaré embeddings for learning hierarchical representations," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 6338–6347.
[18] O. Ganea, G. Becigneul, and T. Hofmann, "Hyperbolic neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 31, 2018, pp. 5345–5355.
[19] M. Norouzi, D. J. Fleet, and R. R. Salakhutdinov, "Hamming distance metric learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 25, 2012, pp. 1061–1069.
[20] J. T. Hoe, K. W. Ng, T. Zhang, C. S. Chan, Y.-Z. Song, and T. Xiang, "One loss for all: Deep hashing with a single cosine similarity based learning objective," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021, pp. 24286–24298.
[21] J. Lee, *Introduction to Topological Manifolds* (Graduate Texts in Mathematics), Springer, 2000.
[22] M. E. Müller, "A note on a method for generating points uniformly on n-dimensional spheres," *Commun. ACM*, vol. 2, no. 4, pp. 19–20, Apr. 1959.
[23] R. Harman and V. Lacko, "On decompositional algorithms for uniform sampling from n-spheres and n-balls," *J. Multivariate Anal.*, vol. 101, no. 10, pp. 2297–2304, Nov. 2010.
[24] M. Vladimirova, J. Verbeek, P. Mesejo, and J. Arbel, "Understanding priors in Bayesian neural networks at the unit level," in *Proc. 36th Int. Conf. Mach. Learn.*, in Proceedings of Machine Learning Research, K. Chaudhuri and R. Salakhutdinov, Eds., Long Beach, CA, USA, vol. 97, Jun. 2019, pp. 6458–6467.
[25] J. S. Brauchart and P. J. Grabner, "Distributing many points on spheres," *J. Complex.*, vol. 31, pp. 293–326, Jun. 2015.
[26] Y. Bengio, N. Léonard, and A. Courville, "Estimating or propagating gradients through stochastic neurons for conditional computation," Aug. 2013, *arXiv:1308.3432*.
[27] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, "The Caltech-UCSD birds-200-2011 dataset," California Inst. Technol., Pasadena, CA, USA, Tech. Rep. CNS-TR-2011-001, 2011. [Online]. Available: http://www.vision.caltech.edu/visipedia-data/CUB-200-2011/
[28] J. Krause, M. Stark, J. Deng, and L. Fei-Fei, "3D object representations for fine-grained categorization," in *Proc. 4th Int. IEEE Workshop 3D Represent. Recognit. (3dRR)*, Sydney, NSW, Australia, 2013, pp. 554–561. [Online]. Available: http://imagenet.stanford.edu/internal/car196/
[29] A. Krizhevsky, "Learning multiple layers of features from tiny images," M.S. thesis, Dept. Comput. Sci., Univ. Toronto, 2009.
[30] Y. Le and X. Yang, "Tiny imagenet visual recognition challenge," cs231n.stanford.edu, 2015.
[31] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Las Vegas, NV, USA, Jun. 2016, pp. 770–778.
[32] G. Zhang, C. Wang, B. Xu, and R. Grosse, "Three mechanisms of weight decay regularization," in *Proc. Int. Conf. Learn. Represent.*, 2019.



YOUNGSUNG KIM received the B.S., M.S., and Ph.D. degrees in electrical and electronic engineering from Yonsei University, in 2006, 2008, and 2012, respectively. He is currently a Faculty Member with the Department of Artificial Intelligence and the Graduate School of Electrical and Computer Engineering, Inha University. He was a Research Staff Member with the Samsung Advanced Institute of Technology (SAIT), from 2014 to 2022, and a Postdoctoral Associate with the MIT SMART Center, from 2013 to 2014.

YOONSUK HYUN received the B.S. degree in mathematics from Seoul National University, Seoul, South Korea, in 2005, and the Ph.D. degree in mathematics from MIT, Cambridge, USA, in 2011. He was a Research Fellow with KIAS, from 2011 to 2015, and a Senior Researcher with the Samsung Advanced Institute of Technology (SAIT), from 2015 to 2020. Since 2020, he has been with the Department of Mathematics, Inha University, Incheon, South Korea, as an Assistant Professor. His current research interests include machine learning, deep learning, and computer vision.

JAE-JOON HAN (Member, IEEE) received the B.S. degree in electronic engineering from Yonsei University, South Korea, in 1997, the M.S. degree in electrical and computer engineering from the University of Southern California, Los Angeles, in 2001, and the Ph.D. degree in electrical and computer engineering from Purdue University, West Lafayette, IN, USA, in 2006. Then, he was a Postdoctoral Fellow with Purdue University, in 2007. Since 2007, he has been with the Samsung Advanced Institute of Technology, Gyeonggi-do, South Korea, where he was promoted to the Vice President of Technology leading computer vision research, in 2018. His research interests include computer vision algorithms and its applications, such as machine vision for semiconductor manufacturing, autonomous driving, learning-based image processing, and edge AI technologies.

EUNHO YANG (Member, IEEE) received the B.S. and M.S. degrees in computer science from Seoul National University, in 2004 and 2006, respectively, and the Ph.D. degree in computer science from The University of Texas at Austin, in 2014. From 2014 to 2016, he was a Research Staff Member with the IBM T. J. Watson Research Center. He is currently an Associate Professor with the Graduate School of Artificial Intelligence and the School of Computing, Korea Advanced Institute of Science and Technology (KAIST). His research interests include learning algorithms, such as effective learning methods in situations where there is insufficient data and applied research on vision, NLP, and speech.

SUNG JU HWANG received the Ph.D. degree in computer science from The University of Texas at Austin, under the supervision of Prof. Kristen Grauman. He is currently an Associate Professor with the Kim Jaechul School of Artificial Intelligence and the School of Computing, KAIST. Prior to working with KAIST, he was an Assistant Professor with the School of Electric and Computer Engineering, UNIST. Before that, he was a Postdoctoral Research Associate with Disney Research, working under the supervision of Prof. Leonid Sigal.

JINWOO SHIN received the B.S. degree in mathematics and in CS from Seoul National University, in 2001, and the Ph.D. degree in mathematics from the Massachusetts Institute of Technology, in 2010, with George M. Sprowls Award (for best MIT CS Ph.D. theses). He is currently a KAIST Endowed Chair Professor (jointly affiliated) with the Kim Jaechul Graduate School of AI and the School of Electrical Engineering, KAIST. He was a Postdoctoral Researcher with the Algorithms & Randomness Center, Georgia Institute of Technology, in 2010 and 2012, and the Business Analytics and Mathematical Sciences Department, IBM T. J. Watson Research, in 2012 and 2013. His early works are mostly on applied probability and theoretical computer science. After, he joined KAIST, in Fall 2013, he started to work on the algorithmic foundations of machine learning. He received the Rising Star Award, in 2015, from the Association for Computing Machinery (ACM) Special Interest Group for the Computer Systems Performance Evaluation Community (SIGMETRICS). He also received the Kenneth C. Sevcik Award at ACM SIGMETRICS/Performance, in 2009, the Best Publication Award from INFORMS Applied Probability Society, in 2013, the Best Paper Award at ACM MOBIHOC, in 2013, the Bloomberg Scientific Research Award, in 2015, and the ACM SIGMETRICS Test of Time Award, in 2019.