Learning Intra-class Multimodal Distributions with Orthonormal Matrices

Jumpei Goto, Yohei Nakata, Kiyofumi Abe, Yasunori Ishii
Panasonic Holdings Corporation, Japan
{goto.jumpei, nakata.yohei, abe.kiyo,
ishii.yasunori}@jp.panasonic.com

Takayoshi Yamashita
Chubu University, Japan
takayoshi@isc.chubu.ac.jp

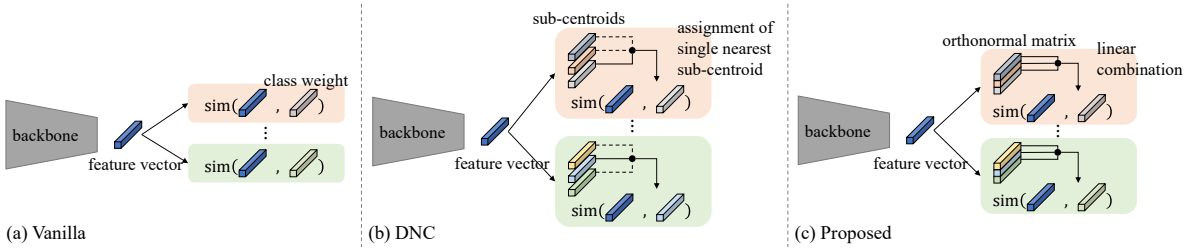


Figure 1. Overview of the proposed method. (a) A vanilla model performs classification by comparing the similarity $\mathrm{sim}(\cdot,\cdot)$ between a feature vector extracted from a backbone and a class-wise weight. (b) The deep nearest centroids method [43] prepares several sub-centroids for each class and assigns the single nearest sub-centroid to a feature vector, which ignores the relationship between the feature vector and the other sub-centroids. (c) The proposed method employs an orthonormal matrix and assigns the matrix rows to a feature vector in a linear combination, which fully utilizes the relationship between the feature vector and each matrix component.

Abstract

In this paper, we address the challenges of representing feature distributions which have multimodality within a class in deep neural networks. Existing online clustering methods employ sub-centroids to capture intra-class variations. However, conducting online clustering faces some limitations, i.e.*, online clustering assigns only a single sub-centroid to a feature vector extracted from a backbone and ignores the relationship between the other sub-centroids and the feature vector, and updating sub-centroids in an online clustering manner incurs significant storage costs. To address these limitations, we propose a novel method utilizing orthonormal matrices instead of sub-centroids for relaxing discrete assignments into continuous assignments. We update the orthonormal matrices using a gradient-based method, which eliminates the need for online clustering or additional storage. Experimental results on the CIFAR and ImageNet datasets exhibit that the proposed method outperforms current online clustering techniques in classification accuracy, sub-category discovery, and transferability, providing an efficient solution to the challenges posed by complex recognition targets.*

1. Introduction

In deep neural networks (DNNs), image classification is a fundamental task that is applied in image recognition [28], object detection [37], and image segmentation [34]. Generally, a DNN classification model comprises a backbone network, which extracts features from images, and a classifier, which infers each category from the extracted features. As shown in Fig. 1(a), the classifier involves a single weight vector for each class, and classes are identified by comparing scores obtained from the inner product between the weight vector and the features. Recently, data variation and amount of data have increased significantly to improve the performance of DNN. However, the increasing amount of data causes the feature distribution to become complex and multimodal, which makes it difficult to represent the feature distribution of each class using only a single weight vector [22].

The k-nearest neighbor (k-NN) classifier [13] can represent such multimodal distributions. The k-NN classifier considers all training features as exemplars, and captures multimodal distributions using these exemplars instead of a single weight vector. Wu *et al*. [44] propose scalable neighborhood component analysis (SNCA), which combines DNNs and neighborhood component analysis (NCA) [20] and utilizes the k-NN classifier. They demonstrate that the representations learned by the SNCA method work effectively in several tasks *e.g*. sub-category discovery. However, k-NN classifiers involve huge costs to store all of the training features for both training and inference phases. In addition, comparing with all training features in-

curs significant computational costs.

In response to these challenges, Wang *et al*. [43] propose the deep nearest centroids (DNC) method. Rather than storing all training features, DNC obtains several sub-centroids for each class via online clustering, as shown in Fig. 1(b). In the inference phase, DNC classifies by finding the nearest sub-centroid from an input feature. However, the online clustering method has several drawbacks. First, only one sub-centroid is assigned to an input feature, even though the input feature is similar to two or more sub-centroids, and the other sub-centroids cannot affect the backbone updates. Second, the update schedulers for backbone parameters and sub-centroids with online clustering are different, making it difficult to maintain balanced updates. Third, online clustering incurs huge memory costs during the training phase. The authors of DNC report that more than 26 GB memory per GPU is necessary for training on ImageNet [38] with only four sub-centroids for each class.

To address these drawbacks, we propose a method to represent multimodal distributions using orthonormal matrices. Our method utilizes sub-centroids as weight matrices and updates the weights using gradient-based approach, which eliminates the need for online clustering or additional storage. As shown in Fig. 1(c), the proposed method calculates the similarity between an input feature vector and the linear combination of the rows in the weight matrices. Therefore, our method fully utilizes the relationship between the input feature and each matrix component. By imposing orthonormal constraints on the class-wise weight matrices, the proposed method can relax the discrete assignment of the sub-centroids in a continuous manner [17]. In addition, the orthonormal constraints enable us to capture more interpretable features than other online clustering methods because uncorrelated features within a class are captured by the orthonormal components of the weight matrices in an unsupervised setting. These orthonormal constraints can be satisfied using Riemannian gradient descent methods as in online principal component analysis (PCA) [1].

Our contributions are summarized as follows:

- We propose a method that learns intra-class multimodal distributions using orthonormal matrices. The method relaxes clustering assignments in online clustering methods to continuous values and obtains interpretable representations without online clustering.

- We show that the classification results are superior due to finer embeddings. With ResNet-50, the proposed method improves the top-1 accuracy on the ImageNet dataset by 0.7 points when compared to DNC [43].

- The experimental results demonstrate that the proposed method exhibits better transferability than conventional methods. In coarse-to-fine experiments, the proposed method exhibits strong transferability on the CIFAR-100 and ImageNet datasets. Strong transferability is also exhibited in transfer learning experiments on the CUB and CityScapes datasets.

2. Related work

Network interpretability. Distance-based methods can enhance the network interpretability of DNNs. The k-NN [13] method is one of the most common distance-based classifiers. Wu *et al*. [44] employ the NCA concept [20] to the DNN architecture, where they demonstrate that the k-NN classifier with a ResNet backbone network is effective in classification tasks and sub-category discovery. However, k-NN classifiers incur huge memory costs during both the training and inference phases because they use all available training features as exemplars. The nearest centroids technique [7] is another well-known distance-based classifier that stores class centers as exemplars. Guerriero *et al*. [21] combine the idea of nearest centroids with DNNs. However, they only utilize a single center for each class as an exemplar, thereby assuming unimodality and failing to capture intra-class multimodal variations. Wang *et al*. [43] prepare several sub-centroids for each class using an online clustering technique to represent the multimodal distributions within classes. These sub-centroids can capture intra-class multimodality. However, online clustering methods still incur huge storage costs during training. In addition, online clustering forces the assignment of an input feature vector to only a single sub-centroid, thereby ignoring data architecture. In contrast, the proposed method employs orthonormal weight matrices rather than sub-centroids, assigns an input feature to the continuous linear combination of the weight matrix, and obtains finer intra-class representations through the matrix components. Rather than using online clustering techniques, in our method, the weight matrices are updated using gradient-based methods, which do not incur extremely huge memory costs.

Prototype learning methods have been proposed based on the concept of distance-based methods. Li *et al*. [30] introduce the prototype layer, which measures distances from prototypes, and the model prediction is interpreted using the prototypes. Angelov and Soares [3] extract typical features as prototypes from networks pretrained on ImageNet, and Chen *et al*. [8] propose the ProtoPNet, which also utilizes features from pretrained backbone networks. ProtoPNet extracts prototypes from patches of feature maps and obtains interpretable activation maps. These prototype learning methods must insert prototype layers and fully-connected layers, which incurs additional computational costs. In addition, ProtoPNet utilizes feature maps extracted from the layers before the pooling layers, thereby incurring further computational costs. In contrast, the proposed method does not implement such additional layers, *e.g*., prototype layers,

and does not use pretrained weights.

Low-rank matrix factorization is used to enhance network interpretability. For example, non-negative matrix factorization (NMF) [29] decomposes an input matrix into two non-negative low-rank matrices. These decomposed matrices capture typical patterns contained in the input matrix. Fel *et al*. [18] leverage NMF to give post-hoc interpretability to models pretrained on ImageNet. The proposed method utilizes the components of orthonormal weight matrices as exemplars to obtain interpretability as in low-rank matrix factorization.

Orthogonality. Introducing orthogonality into DNN weights can improve accuracy and robustness to stabilize training. Xie *et al*. [45] confirm that weight orthogonality in linear and convolutional layers reduces degradation in plain CNNs. Cisse *et al*. [10] demonstrate that orthogonal weights in linear and convolutional layers improve robustness against adversarial attacks. Wang *et al*. [42] introduce orthogonality into the weights of convolutional layers represented in the doubly block-Toeplitz form. These methods impose an orthogonal constraint by adding the soft regularization term $\frac{\lambda}{2}\|\mathbf{W}^\top\mathbf{W} - I\|_F^2$ to the loss function, where $\lambda > 0$ is a hyperparameter. In contrast, the method proposed by Li *et al*. [31] strictly satisfies orthogonality via singular value decomposition. While the above methods impose orthogonality constraints to the entire networks, our method only applies orthogonality to the classifier. In the proposed method, Riemannian optimization [2] on a Stiefel manifold is employed to avoid the extra hyperparameter λ and strictly satisfy orthogonality.

Orthogonality is also applied in matrix decomposition and reconstruction. In PCA, an input matrix is decomposed into non-correlated components by utilizing orthonormality and obtaining interpretability. In addition, a discrete clustering problem can be relaxed into a continuous optimization problem by adopting orthogonal matrix decomposition [16, 17]. Zhang *et al*. [47] leverage the orthogonal matrix reconstruction concept to improve accuracy and explainability via salient activation tracking. In the proposed method, an objective function similar to PCA is implemented to enhance model interpretability.

Orthogonality also facilitates the separation of class features. This property is utilized in few-shot learning, where a new class weight is appended orthogonally to existing class weights in classification [24] and segmentation [32] tasks. This separability of orthogonality helps the proposed method capture finer representations.

3. Intra-class multimodal distributions with orthonormal matrix

In the following subsections, we introduce the general classification problem settings, outline DNC [43], and describe its limitations. Finally, we describe the proposed method in detail.

3.1. Problem formulation

Here, let $\mathcal{X}$ and $\mathcal{Y} = \{1, 2, \dots, C\}$ be an image space and a set of semantic classes, respectively. The goal of classification problems is to fit a model $h : \mathcal{X} \to \mathcal{Y}$ using a given training dataset $\mathcal{T} = \{(\mathbf{x}_i, y_i) \in \mathcal{X} \times \mathcal{Y}\}_{i=1}^N$ to predict a semantic class of a new image $\mathbf{x} \in \mathcal{X}$. A common DNN model can be decomposed as $h = g \circ f_\theta$, where $f_\theta : \mathcal{X} \to \mathbb{R}^d$ is a feature extractor (or backbone), θ is a trainable parameter vector, and $g : \mathbb{R}^d \to \mathcal{Y}$ is a classifier. The widely used linear classifier can be written as

$$g(\mathbf{z}) = \arg\max_{c \in \mathcal{Y}} \ell_c(\mathbf{z}), \tag{1}$$

$$\ell_c(\mathbf{z}) = {\mathbf{w}^c}^\top \mathbf{z} + b_c, \tag{2}$$

where $\mathbf{z} \in \mathbb{R}^d$ is a feature vector, $\mathbf{w}^c \in \mathbb{R}^d$ and $b_c \in \mathbb{R}$ are a class-wise weight vector and a bias, respectively, and $\ell : \mathbb{R}^d \to \mathbb{R}$ is a logit function.

In Eq. (2), the inner product ${\mathbf{w}^c}^\top \mathbf{z}$ calculates the similarity between the weight vector $\mathbf{w}^c$ and feature vector $\mathbf{z}$. Thus, letting $\mathrm{sim} : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ be a similarity function, the logit function ℓ_c can be represented as

$$\ell_c(\mathbf{z}) = \mathrm{sim}(\mathbf{w}^c, \mathbf{z}). \tag{3}$$

Several functions can be used for $\mathrm{sim}(\cdot, \cdot)$ such as the cosine similarity [36] and ℓ_2-norm [6].

3.2. Deep nearest centroids

A classification model implementing Eq. (3) is trained such that all feature vectors belonging to the semantic class c are attracted to only a single weight vector $\mathbf{w}^c$. Thus, as shown in Fig. 2(a), the learned features of each class tend to be distributed unimodally. However, in practice, intra-class feature distributions are diverse, and a single weight vector cannot represent intra-class variations. Wang *et al*. [43] tackle this intra-class representation problem by utilizing the nearest centroids [7] concept. They propose DNC, which represents each class c by K sub-centroids $\{\mathbf{w}_i^c\}_{i=1}^K$ and updates sub-centroids in an online clustering manner. As shown in Fig. 2(b), DNC captures the multimodal distribution of feature vectors via these sub-centroids.

Training DNC is accomplished by alternately updating the parameters θ of the feature extractor and the sub-centroids $\{\mathbf{w}_i^c\}_{i=1}^K$. Parameters θ are learned by minimizing the softmax cross-entropy loss using the following logit function as

$$\ell_c(\mathbf{z}) = \max_{i \in \{1,2,\dots,K\}} \mathrm{sim}(\mathbf{w}_i^c, \mathbf{z}), \tag{4}$$

where both $\mathbf{w}_i^c$ and $\mathbf{z}$ are ℓ_2-normalized and $\mathrm{sim}(\cdot, \cdot)$ is cosine similarity. Here, let $\mathbf{W}^c := [\mathbf{w}_1^c, \mathbf{w}_2^c, \dots, \mathbf{w}_K^c]^\top \in \mathbb{R}^{K \times d}$ for each class c and $\mathbf{a}^c \in \{0, 1\}^K$ be a vector that

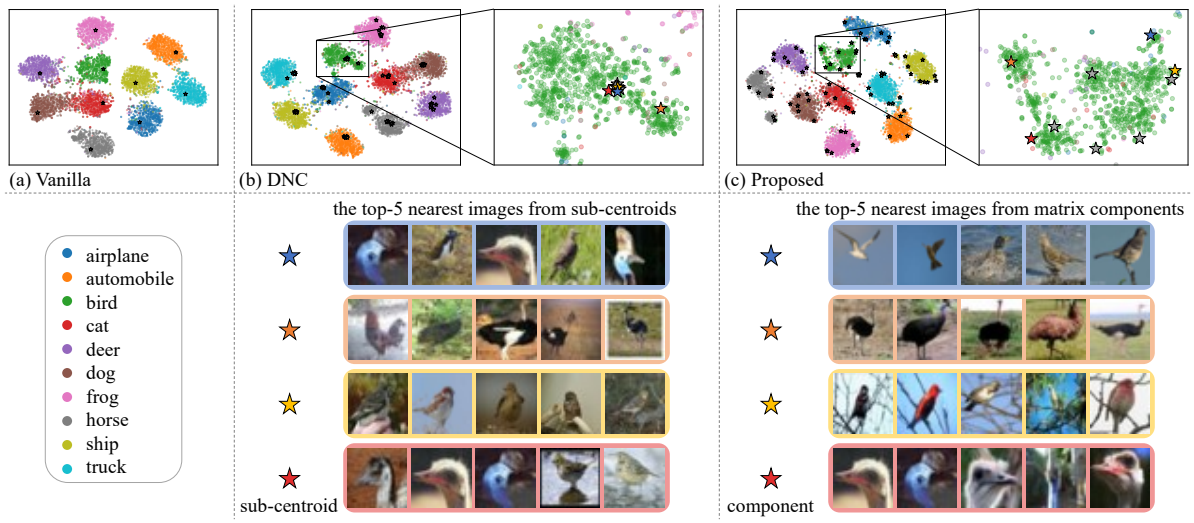


Figure 2. t-SNE [40] visualizations of the extracted features of CIFAR-10 [27] (top row), and the top-5 nearest images from the sub-centroid for DNC or the matrix component for our method (second row). The zoomed green scatter plots show the feature distribution of "bird" class, and star plots represent the weight vectors for the vanilla model (a), the sub-centroids for DNC (b), and the matrix components for the proposed method (c).

assigns a single sub-centroid to the feature vector $\mathbf{z}$ as

$$a_i^c = \begin{cases} 1, & \text{if } i = \arg\max_{j \in \{1,2,\dots,K\}} \text{sim}(\mathbf{w}_j^c, \mathbf{z}), \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

Then, Eq. (4) can be rewritten as

$$\ell_c(\mathbf{z}) = \text{sim}\left(\mathbf{W}^{c\top}\mathbf{a}^c, \mathbf{z}\right). \tag{6}$$

In the update of sub-centroids, DNC adopts online clustering using a momentum update technique. During each training iteration, DNC conducts class-wise clustering on a batch. Here, let $\bar{\mathbf{z}}_i^c$ be a mean vector of the features assigned to i-th cluster of class c in the current batch. Then, the update of sub-centroid $\mathbf{w}_i^c$ is

$$\mathbf{w}_i^c \leftarrow \mu \mathbf{w}_i^c + (1-\mu)\bar{\mathbf{z}}_i^c, \tag{7}$$

where $\mu \in [0, 1]$ is a momentum coefficient. To assign a feature vector $\mathbf{z}$ to an appropriate sub-centroid $\mathbf{w}_i^c$, they employ Sinkhorn-Knopp iteration-based clustering [15] which solves the optimal transport problem with entropy regularization. However, due to the application of online clustering, DNC assigns a feature vector $\mathbf{z}$ to only a single sub-centroid $\mathbf{w}_i^c$, which implies that even if $\mathbf{z}$ is similar to two sub-centroids $\mathbf{w}_i^c$ and $\mathbf{w}_j^c$, if $\text{sim}(\mathbf{w}_i^c, \mathbf{z})$ is slightly larger than $\text{sim}(\mathbf{w}_j^c, \mathbf{z})$, the feature extractor is trained to make $\mathbf{z}$ similar to $\mathbf{w}_i^c$. Unfortunately, in such a case, the property that $\mathbf{z}$ resembles $\mathbf{w}_j^c$ is ignored.

In addition, online clustering makes it difficult to maintain an effective balance between updating backbone parameters and sub-centroids. DNC updates the backbone parameters using SGD with a learning rate scheduled by a step policy. On the other hand, sub-centroids are updated with a constant step size in Eq. (7). As a result, the updates of parameters and sub-centroids become unbalanced.

Furthermore, online clustering requires external memory to store past features. For example, consider training on ImageNet with 1K classes using a batch size of 256. Here, the batch size is obviously smaller than the number of classes. Therefore, DNC must store several previous batches in external memory to conduct online clustering. However, this prevents the larger number of sub-centroids, and the authors of DNC set $K = 4$ due to memory limitations [43].

3.3. Proposed method

3.3.1 Relaxing the assignment vector

Equation (6) implies that the matrix $\mathbf{W}^c$ can potentially capture intra-class variations, and considering $\mathbf{W}^c$ as a weight matrix allows us to avoid the aforementioned problems associated with online clustering. This motivates us to simply replace the sub-centroids $\{\mathbf{w}_i^c\}_{i=1}^K$ with a weight matrix $\mathbf{W}^c \in \mathbb{R}^{n \times d}$ with $n < d$. By adapting a weight matrix, we update $\mathbf{W}^c$ using gradient descent methods rather than online clustering techniques. To apply gradient-based methods, we modify the logit function in Eq. (6) by relaxing the discrete assignment vector $\mathbf{a}^c \in \{0, 1\}^K$ with a continuous vector $\mathbf{a}^c \in \mathbb{R}_+^n$. This relaxation allows all feature vectors to affect the updates of weight matrices for all classes through the cross-entropy loss.

Suppose the similarity function is ℓ_2-norm with a scaling parameter $\kappa > 0$

$$\text{sim}(\mathbf{u}, \mathbf{v}) = -\kappa \|\mathbf{u} - \mathbf{v}\|_2^2. \tag{8}$$

Then, Eq. (6) can be rewritten as

$$\ell_c(\mathbf{z}; \kappa) = -\kappa \left\| \mathbf{z} - \mathbf{W}^{c\top}\mathbf{a}^c \right\|_2^2. \tag{9}$$

Here, two issues must be considered when performing gradient descent with softmax cross-entropy minimization with this logit. First, a weight matrix $\mathbf{W}^c$ can be non-full-rank, and the intra-class variation of class c may not be represented with the non-full-rank matrix. For example, assume $n = 2$. Let $\mathbf{W}_1^c$ and $\mathbf{W}_2^c$ be the first and second rows of $\mathbf{W}^c$, respectively. If $\mathbf{W}^c$ is not full rank, then there exists $\alpha \in \mathbb{R}$ satisfying $\mathbf{W}_2^c = \alpha \mathbf{W}_1^c$. By letting $\mathbf{a}^c = (a_1^c, a_2^c)^\top$, the equation ${\mathbf{W}^c}^\top \mathbf{a}^c = (a_1^c + a_2^c \alpha)\mathbf{W}_1^c$ holds. This implies that all feature vectors belonging to c are attracted to $\mathbf{W}_1^c$, thereby resulting in a unimodal distribution. The second issue is determining how to assign the vectors. As an extreme case, assume that $\mathbf{a}^c = [1, 0, \ldots, 0]^\top$ holds for all feature vectors. Then, ${\mathbf{W}^c}^\top \mathbf{a}^c = \mathbf{W}_1^c$ holds, which means that using ${\mathbf{W}^c}^\top \mathbf{a}^c$ in Eq. (6) is equivalent to using $\mathbf{W}_1^c$ in Eq. (3), thereby failing to represent multimodal distribution.

3.3.2 Orthonormal constraint

In the following, we demonstrate that introducing orthonormality to the weight matrices resolves the above shortcomings. Here, let $\mathbf{W}^c$ be an orthonormal matrix, *i.e.*, $\mathbf{W}^c$ satisfies the constraint $\mathbf{W}^c {\mathbf{W}^c}^\top = \mathbf{I}_n$ where $\mathbf{I}_d$ is n-dimensional identity matrix, which implies that $\mathbf{W}^c$ is always full rank. In addition, by utilizing the orthonormality, we can determine the non-negative assignment vector $\mathbf{a}^c = \mathrm{ReLU}\left(\mathbf{W}^c \mathbf{z}\right)$ to minimize the ℓ_2-norm $\left\| \mathbf{z} - {\mathbf{W}^c}^\top \mathbf{a}^c \right\|_2^2$. Using this assignment vector, Eq. (9) can be rewritten as

$$\ell_c(\mathbf{z}; \kappa) = -\kappa \left\| \mathbf{z} - {\mathbf{W}^c}^\top \mathrm{ReLU}\left(\mathbf{W}^c \mathbf{z}\right) \right\|_2^2 . \tag{10}$$

Here, we apply ℓ_2-normalization to the feature vectors as DNC does, and we omit constant terms. Then, the logit function can be expressed as

$$\ell_c(\mathbf{z}; \kappa) = \kappa \left\| \mathrm{ReLU}\left(\mathbf{W}^c \mathbf{z}\right) \right\|_2^2 . \tag{11}$$

Note that $\mathbf{a}^c = \mathrm{ReLU}\left(\mathbf{W}^c \mathbf{z}\right)$ satisfies $0 \leq a_i^c \leq 1$ with ℓ_2-normalized features, which can be considered as continual relaxation of the assignment for DNC expressed in Eq. (5).

In summary, we can describe the softmax cross-entropy minimization with $\ell_c(\mathbf{z}; \kappa)$ as

$$\begin{aligned} \underset{\theta, \mathbf{W}^1, \ldots, \mathbf{W}^C}{\text{minimize}} \quad & -\frac{1}{N} \sum_{(\mathbf{x}_i, y_i) \in \mathcal{T}} \log \left(\frac{\exp\left(\ell_{y_i}(f_\theta(\mathbf{x}_i); \kappa)\right)}{\sum_{k=1}^C \exp\left(\ell_k(f_\theta(\mathbf{x}_i); \kappa)\right)} \right), \\ \text{subject to} \quad & \mathbf{W}^c {\mathbf{W}^c}^\top = \mathbf{I}_n, \quad c = 1, 2, \ldots, C. \end{aligned} \tag{12}$$

This problem can be considered as an orthonormal constrained optimization problem. To solve an orthonormal constrained optimization problem, the constraint can be relaxed using Lagrange multiplier $\lambda > 0$, and adding the regularization term $\lambda \left\| \mathbf{W}^c {\mathbf{W}^c}^\top - \mathbf{I}_n \right\|$ described in [10]. This method is efficient in terms of computational cost if the number of constraints is large. However, we must tune the penalty parameter λ properly to balance the losses. In our case, the number of orthonormal constraints is sufficiently small to adopt Riemannian optimization. We can consider orthonormal constrained optimization as an unconstrained optimization problem on a Stiefel manifold as

$$\mathrm{St}(d, n) \coloneqq \left\{ \mathbf{W} \in \mathbb{R}^{n \times d} \mid \mathbf{W}\mathbf{W}^\top = \mathbf{I}_n \right\} . \tag{13}$$

A Stiefel manifold is a type of Riemannian manifold. Thus, we can apply Riemannian optimization, *e.g.*, Riemannian SGD [5] and Riemannian Adam [4], to the optimization problem expressed in Eq. (12). In the Riemannian optimization process, a map (referred to as a retraction) is utilized to update variables. In the proposed method, we use a QR retraction [2] as a retraction on a Stiefel manifold. Note that the weight decay for $\mathbf{W}^c$ is set to zero because $\|\mathbf{W}^c\|_F^2 = n$ holds due to the orthonormal constraint.

3.4. Difference between DNC and proposed method

In DNC, sub-centroids represent a mean vector of the features. Thus, each feature vector tends to distribute around the sub-centroids as shown in Fig. 2(b). In contrast, the proposed method learns to represent the feature vector $\mathbf{z}$ via a linear combination $\sum_{i=1}^n a_i^c \mathbf{W}_i^c$. The assignment vector $\mathbf{a}^c$ satisfies $0 \leq a_i^c \leq 1$ and $\|\mathbf{a}^c\|_2 \leq 1$. Therefore, the weight matrices $\{\mathbf{W}_1^c, \mathbf{W}_2^c, \ldots, \mathbf{W}_n^c\}$ represent the edge of the feature distribution, as shown in Fig. 2(c). This enables our method to capture finer representations such as "a bird on a blue sea or a blue sky" and "a bird perching on a branch," which cannot be obtained using DNC.

4. Experiment

In this section, we discuss experiments conducted to evaluate the performance and effectiveness of the proposed method. We report the mean accuracy and the standard deviation error bars of three experiments conducted with different initialization seeds.

4.1. Classification

We evaluate the classification performance of the proposed method on the CIFAR-10/100 [27] and ImageNet [38] datasets. We compare the proposed method with the vanilla model, DeepNCM [21], SNCA [44], and DNC [43]. ResNet [23] and Swin [33] are used for comparison between CNN and transformer-based architectures.

Learning details. We use MMClassification [11] and follow the default settings in terms of the batch size, number of epochs, and optimizers. For CIFAR-10/100, we train ResNet for 200 epochs with a batch size of 128 (32 batches per GPU) and a resolution of 32×32. For ImageNet, we set the resolution to 224×224, and train ResNet for 100 epochs with a batch size of 256 (32 batches per GPU) and Swin for 300 epochs with a batch size of 1,024 (64 batches per GPU). To train the ResNet model, we use the Riemannian SGD optimizer with an initial learning rate of 0.1, momen-

tum of 0.9, and weight decay of 0.0001. A step scheduler is adopted, where the learning rate decays by 0.1 in {100, 150} epochs for CIFAR-10, 0.2 in {60, 120, 160} epochs for CIFAR-100, and 0.1 in {30, 60, 90} epochs for ImageNet. To facilitate stable training, the learning rate for the orthonormal weight matrices of the classifiers in the proposed method is multiplied by 0.1. To train Swin, we use the Riemannian AdamW optimizer [35] to minimize the label smoothing loss [39] and set the initial learning rate to 0.001 and the weight decay to 0.05. The learning rate is scheduled using a cosine annealing policy. Standard data augmentations are applied, including flipping and cropping during both the ResNet and Swin training phases. The RandAugment [14], Random Erasing [49], MixUp [48], and CutMix [46] augmentations are also applied in Swin training. Riemannian optimization is implemented based on geoopt [26]. The hyperparameters of the proposed method are set to $n = 10$ and $\kappa = 50 \times d/2048$, where $d = 2048$ for ResNet, $d = 768$ for Swin-S, and $d = 1024$ for Swin-B. To reproduce DNC results, we set the number of sub-centroids $K = 4$. In addition, we use $K = 10$ for fair comparison.

Main results. Tables 1 to 3 show GPU memory usage in the training phase and the top-1/5 accuracy results on CIFAR-10/100 and ImageNet, respectively. In Tab. 3, N/A indicates that we cannot measure GPU memory usage on NVIDIA A100 40GB due to the memory limitation. Experiments using DNC with $K = 10$ on ImageNet cannot also be conducted due to the memory limitation. We confirm that the proposed method does not require huge additional storage which DNC has need of. On the CIFAR-10 dataset, the top-1 accuracy of the proposed method is slightly less than that of the vanilla method. However, as shown in Fig. 2, the proposed method captures more interpretable feature representations, *e.g.*, "a bird perching on a branch," which cannot be acquired by the vanilla method. On the CIFAR-100 dataset, the proposed method outperforms the other baselines by 0.45 points in terms of the top-1 accuracy and 0.83 points in terms of the top-5 accuracy. On the ImageNet dataset, the proposed method outperforms the baselines by 0.47 points in terms of the top-1 accuracy with ResNet. SNCA is trained on ImageNet for 130 epochs. However, we find that the proposed method obtains higher score with only 100 epochs training. With Swin, the proposed method outperforms the vanilla method by 0.07 points (top-1 accuracy). However, the accuracy of the proposed method is less than that of DNC. The bottom two lines in Tab. 3 show the Swin-S experimental results using the cross-entropy loss instead of the label smoothing loss. Using the cross-entropy loss on Swin-S, the proposed method exceeds the vanilla model by 0.32 points in terms of the top-1 accuracy, which is much higher than 0.07 points improvement with the label smoothing loss. These results on Swin-S imply that the proposed method is incompatible with the label smoothing loss. As shown in Tabs. 1 to 3, the proposed method especially improves the top-5 accuracy. This means that the proposed method embeds similar inputs close together in the feature space. and captures inter-class relationship. In contrast, the label smoothing loss treats all classes other than the ground truth class equally by assigning a constant value to these classes, which ignores inter-class relationship. These properties contradict each other and incur lower accuracy than DNC.

Table 1. Top-1 and top-5 accuracy (%) on CIFAR-10 [27] validation set and training GPU memory usage (GB per GPU). The accuracy results of DeepNCM are reported in the literature [43].

Method	Backbone	memory	top-1	top-5
Vanilla	ResNet-50	1.13	**95.52** ± 0.08	**99.87**
DeepNCM [21]	ResNet-50	-	93.67	-
DNC$_{K=4}$ [43]	ResNet-50	1.55	95.17 ± 0.11	99.74
DNC$_{K=10}$ [43]	ResNet-50	1.56	95.19 ± 0.10	99.73
Proposed	ResNet-50	1.14	95.39 ± 0.10	99.87
Vanilla	ResNet-101	1.82	**95.46** ± 0.16	99.87
DNC$_{K=4}$ [43]	ResNet-101	2.24	95.34 ± 0.08	99.81
DNC$_{K=10}$ [43]	ResNet-101	2.25	95.41 ± 0.04	99.74
Proposed	ResNet-101	1.82	95.42 ± 0.06	**99.88**

Table 2. Top-1 and top-5 accuracy (%) on CIFAR-100 [27] validation set and training GPU memory usage (GB per GPU). The accuracy results of DeepNCM are reported in the literature [43].

Method	Backbone	memory	top-1	top-5
Vanilla	ResNet-50	1.14	80.07 ± 0.17	94.99
DeepNCM [21]	ResNet-50	-	72.25	-
DNC$_{K=4}$ [43]	ResNet-50	1.60	78.88 ± 0.43	92.92
DNC$_{K=10}$ [43]	ResNet-50	1.72	79.31 ± 0.14	93.48
Proposed	ResNet-50	1.17	**80.53** ± 0.30	**95.82**
Vanilla	ResNet-101	1.82	80.31 ± 0.10	94.94
DNC$_{K=4}$ [43]	ResNet-101	2.29	79.71 ± 0.07	93.31
DNC$_{K=10}$ [43]	ResNet-101	2.41	79.28 ± 0.29	93.04
Proposed	ResNet-101	1.85	**80.76** ± 0.20	**95.83**

Table 3. Top-1 and top-5 accuracy (%) on ImageNet [38] validation set and training GPU memory usage (GB per GPU). The accuracy results for the vanilla [43], DNC [43] and SNCA [44] methods are reported in the corresponding literature.

Method	Backbone	memory	top-1	top-5
Vanilla	ResNet-50	2.99	76.20 ± 0.10	93.01
SNCA [44]	ResNet-50	-	76.67	92.84
DNC$_{K=4}$ [43]	ResNet-50	24.58	76.49 ± 0.09	93.08
Proposed	ResNet-50	3.32	**77.19** ± 0.15	**93.55**
Vanilla	ResNet-101	4.50	77.52 ± 0.11	93.06
DNC$_{K=4}$ [43]	ResNet-101	26.09	77.80 ± 0.10	93.85
Proposed	ResNet-101	4.83	**78.27** ± 0.06	**94.17**
Vanilla	Swin-S	11.17	83.02 ± 0.14	96.29
DNC$_{K=4}$ [43]	Swin-S	N/A	**83.26** ± 0.13	**96.40**
Proposed	Swin-S	11.30	83.09 ± 0.06	96.37
Vanilla	Swin-B	15.25	83.36 ± 0.12	96.44
DNC$_{K=4}$ [43]	Swin-B	N/A	**83.68** ± 0.12	**97.02**
Proposed	Swin-B	15.42	83.43 ± 0.01	96.54
Vanilla w/ CE	Swin-S	11.17	82.57 ± 0.07	96.11
Proposed w/ CE	Swin-S	11.30	**82.89** ± 0.02	**96.29**

Table 4. Top-1 accuracy (%) with coarse-grained and fine-grained labels on CIFAR-100 [27] validation set using models trained on CIFAR-20. The SNCA results are reported in the literature [44].

Method	Backbone	20 classes	100 classes
Vanilla	ResNet-50	87.04 ± 0.21	53.04 ± 0.26
SNCA [44]	ResNet-50	81.42	62.32
$DNC_{K=4}$ [43]	ResNet-50	86.60 ± 0.12	68.70 ± 0.48
Proposed	ResNet-50	**87.43** ± 0.16	**70.94** ± 0.07
Vanilla	ResNet-101	87.05 ± 0.25	48.96 ± 0.62
$DNC_{K=4}$ [43]	ResNet-101	86.63 ± 0.41	67.48 ± 0.59
Proposed	ResNet-101	**87.57** ± 0.14	**71.17** ± 0.11

Table 5. Top-1 accuracy (%) with coarse-grained and fine-grained labels on ImageNet [38] validation set using models trained on ImageNet-127. The results for the vanilla [43], DNC [43] and SNCA [44] methods are reported in the corresponding literature.

Method	Backbone	127 classes	1000 classes
Vanilla	ResNet-50	84.29	43.23
SNCA [44]	ResNet-50	81.62	52.75
$DNC_{K=4}$ [43]	ResNet-50	84.39	52.21
Proposed	ResNet-50	**85.81** ± 0.12	**56.08** ± 0.11
Vanilla	ResNet-101	85.88	45.31
$DNC_{K=4}$ [43]	ResNet-101	85.91	54.60
Proposed	ResNet-101	**86.44** ± 0.07	**56.94** ± 0.24

Table 6. Top-1 and top-5 accuracy (%) on CUB [41] test set. The results for the vanilla and DNC methods are reported in the literature [43].

Method	Backbone	top-1	top-5
Vanilla	ResNet-50	84.48	96.31
DNC [43]	ResNet-50	85.21	96.70
Proposed	ResNet-50	**85.44** ± 0.25	**96.91**

Table 7. Segmentation mIoU score (%) on CityScapes [12] validation set. The results for the vanilla and DNC methods are reported in the literature [43].

Pretrain method	Backbone	mIoU
Vanilla	ResNet-101	78.1 ± 0.12
DNC [43]	ResNet-101	78.7 ± 0.13
Proposed	ResNet-101	**79.0** ± 0.09

4.2. Coarse-to-fine experiment

As the classes become coarser, the intra-class distributions become increasingly complex. Thus, the feature distributions are multimodal. Coarse-to-fine experiments, which train the model using coarse-grained labels and evaluate with fine-grained labels, can be conducted to measure the performance on the multimodality.

We follow the experimental setup described in the literature [25] using CIFAR-100 and ImageNet. CIFAR-100 has 100 fine-grained categories and 20 coarse-grained categories (CIFAR-20). With the ImageNet dataset, as described in the literature [25], we can acquire 127 coarse-grained labels (ImageNet-127) from 1,000 fine-grained labels via top-down clustering on WordNet tree [19]. We train the ResNet under the conditions described in Sec. 4.1 using coarse-grained labels. To evaluate performance with the fine-grained labels, we use the top-1 nearest neighbor accuracy with the cosine similarity using the features extracted by the backbone network.

Table 8. Ablation experiment on CIFAR-100/20 [27] validation set for the number of rows of orthonormal weight matrices (n).

n	CIFAR-100 top-1	CIFAR-100 top-5	CIFAR-20 20 classes	CIFAR-20 100 classes
4	79.92 ± 0.13	95.56	87.20 ± 0.12	70.12 ± 0.10
7	80.31 ± 0.14	95.56	87.31 ± 0.26	70.74 ± 0.32
10	**80.53** ± 0.30	**95.82**	**87.43** ± 0.16	70.94 ± 0.07
13	80.08 ± 0.22	95.55	87.27 ± 0.13	**71.13** ± 0.32

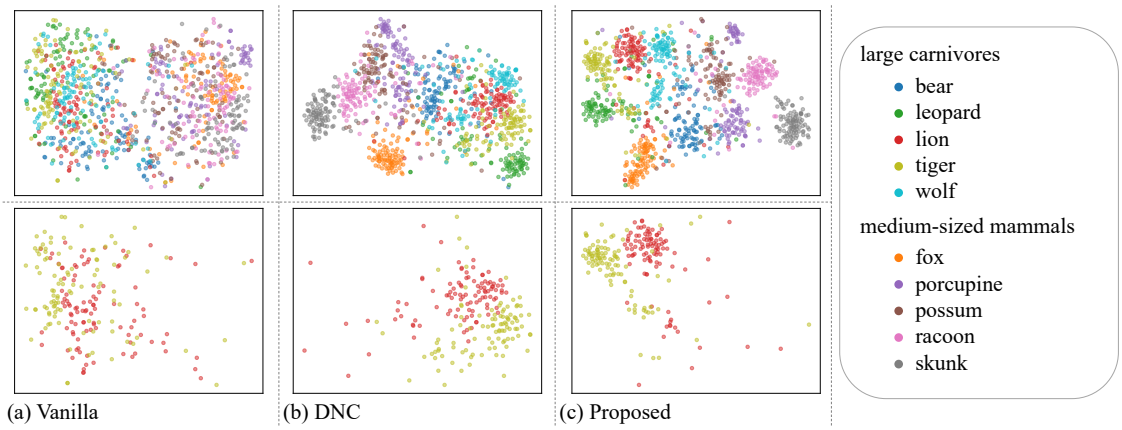
Table 9. Ablation experiment on CIFAR-100/20 [27] validation set for scaling parameter (κ).

κ	CIFAR-100 top-1	CIFAR-100 top-5	CIFAR-20 20 classes	CIFAR-20 100 classes
10	79.30 ± 0.24	93.74	86.56 ± 0.26	55.33 ± 0.29
50	**80.53** ± 0.30	**95.82**	**87.43** ± 0.16	**70.94** ± 0.07
100	78.29 ± 0.23	94.93	87.08 ± 0.16	69.33 ± 0.30

Main results. Tables 4 and 5 show the accuracy results obtained with the coarse- and fine-grained labels on CIFAR-20 and ImageNet-127, respectively. As can be seen, on CIFAR-20, the proposed method outperforms the compared methods by 2.24 points in accuracy with the fine-grained labels, and it exceeded 2.34 points in terms of fine-grained accuracy on ImageNet-127. Figure 3 shows t-SNE visualization results on CIFAR-20 for the "large carnivores" and "medium-sized mammals" classes, and Fig. 4 shows the nearest neighbor examples of the CIFAR-20 training images from the query test images. By comparing results of the vanilla method in Fig. 3(a) with those of DNC in Fig. 3(b), we see that DNC separates some classes, *e.g.*, "leopard," "fox," and "skunk" from the other classes. However, the boundaries among the similar fine-grained classes, *e.g.*, "lion" and "tiger", are not clear with DNC. In contrast, as shown in Fig. 3(c), the proposed method obtains a larger margin among the fine-grained categories without ground-truth, fine-grained labels. This margin helps the proposed method discriminate between "lion" and "tiger," as shown in the first and second rows in Fig. 4.

4.3. Transferability

We evaluate the transfer learning performance of the proposed method by training pretrained ImageNet models on Caltech-UCSD Birds-200-2011 (CUB) [41] and CityScapes [12]. We follow the experimental setup described in the DNC literature [43]. For CUB, we train the ResNet-50 model for 100 epochs with a resolution of 448×448 and a batch size of 64. We use the Riemannian SGD optimizer with an initial learning rate of 0.01, momentum of 0.9, and weight decay of 0.0005. The learning rate is scheduled using a cosine annealing policy. For CityScapes,



(a) Vanilla (b) DNC (c) Proposed

Figure 3. t-SNE [40] visualizations of features extracted from CIFAR-20 [27] images of "large carnivores" and "medium-sized mammals" classes (first row). The second row extracts two sub-categories, *i.e.*, "lion" and "tiger," from the scatters plotted in the first row for better visualization. The proposed method (c) constructed more compact clusters than the vanilla model (a) and DNC (b).

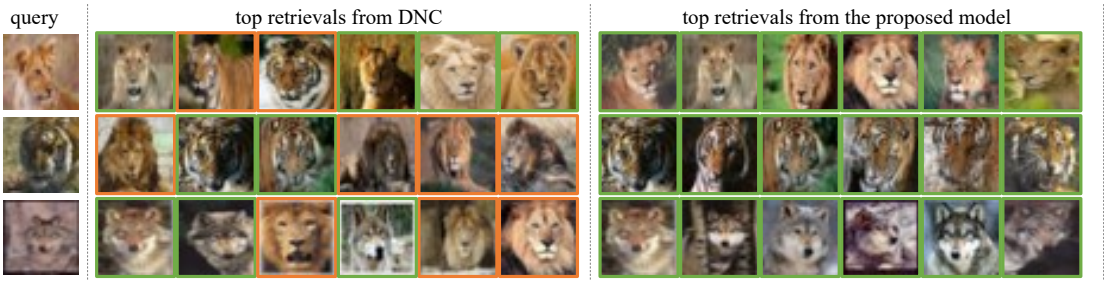


Figure 4. Nearest neighbor images from the models trained on CIFAR-20 [27] and evaluated on the fine-grained labels. Correct and incorrect retrievals are framed with green and orange grids, respectively. The query images are from "lion", "tiger", and "wolf" categories.

we train DeepLab$_{\text{V3}}$ [9] with ResNet-101 for 160K iterations with a resolution of 769 $\times$ 769 and a batch size of 8. We use the Riemannian SGD optimizer with an initial learning rate of 0.01, momentum of 0.9, and weight decay of 0.0005. The learning rate is scheduled using a polynomial annealing policy.

Main results. Tables 6 and 7 show the experimental results on CUB and CityScapes, respectively. As can be seen, the proposed method outperforms both the vanilla method and DNC, by 0.23 points (top-1 accuracy) and 0.21 points (top-5 accuracy) on CUB, and by 0.3 points (mIoU) on CityScapes, which demonstrates that the proposed method can realize higher transferability on both classification and semantic segmentation tasks.

4.4. Ablation study

We conduct classification experiments on CIFAR-100 and coarse-to-fine experiments on CIFAR-20 with ResNet-50 as ablation studies.

Orthonormal weight matrices' dimension. Table 8 shows the effect of the number of rows of the orthonormal weight matrices (n). We find that increasing n from 4 to 10 yielded accuracy improvements, but the accuracy is reduced with $n = 13$. Although increasing n promotes representation ability for each class, setting a large n value causes a conflict among the class subspaces spanned by the weight matrices due to the finite embedding dimension d.

Scaling parameter. Table 9 studies the impact of the scaling parameter κ. As shown in Eq. (11), κ controls the similarity between the input feature and the linear combination of the rows of the weight matrix. The accuracy obtained with $\kappa = 50$ is more than 1.0 points higher than that with $\kappa = 10$ and $\kappa = 100$. These results imply that κ has a substantial influence on the accuracy of the model.

5. Conclusion

This paper proposed a classification method that considers the intra-class multimodal properties of data. The proposed method utilized orthonormal matrices to relax cluster assignments in the existing methods to continuous values. This technique enabled us to train models using gradient-based methods without the need for online clustering which often leads to significant storage costs. Additionally, the utilization of orthonormal matrices enabled us to capture more refined representations through orthonormal components. The proposed method was evaluated in an extensive set of experiments, and results confirmed that the proposed method improves both model accuracy and interpretability.

Acknowledgement: We would like to thank Denis Gudovskiy for carefully proofreading the manuscript.

References

[1] Pierre Ablin, Simon Vary, Bin Gao, and P-A Absil. Infeasible deterministic, stochastic, and variance-reduction algorithms for optimization under orthogonality constraints. *arXiv preprint arXiv:2303.16510*, 2023. 2

[2] P-A Absil, Robert Mahony, and Rodolphe Sepulchre. *Optimization algorithms on matrix manifolds*. Princeton University Press, 2008. 3, 5

[3] Plamen Angelov and Eduardo Soares. Towards explainable deep neural networks (xdnn). *Neural Networks*, 130:185–194, 2020. 2

[4] Gary Becigneul and Octavian-Eugen Ganea. Riemannian adaptive optimization methods. In *ICLR*, 2019. 5

[5] Silvere Bonnabel. Stochastic gradient descent on Riemannian manifolds. *IEEE Trans. Aut. Contr.*, 58(9):2217–2229, 2013. 5

[6] Jun Cen, Peng Yun, Junhao Cai, Michael Yu Wang, and Ming Liu. Deep metric learning for open world semantic segmentation. In *ICCV*, pages 15333–15342, 2021. 3

[7] BB Chaudhuri. A new definition of neighborhood of a point in multi-dimensional space. *Pattern Recognition Letters*, 17(1):11–17, 1996. 2, 3

[8] Chaofan Chen, Oscar Li, Daniel Tao, Alina Barnett, Cynthia Rudin, and Jonathan K Su. This looks like that: Deep learning for interpretable image recognition. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *NeurIPS*, volume 32. Curran Associates, Inc., 2019. 2

[9] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017. 8

[10] Moustapha Cisse, Piotr Bojanowski, Edouard Grave, Yann Dauphin, and Nicolas Usunier. Parseval networks: Improving robustness to adversarial examples. pages 854–863. PMLR, 2017. 3, 5

[11] MMClassification Contributors. Openmmlab's image classification toolbox and benchmark. *URL: https://github.com/open-mmlab/mmclassification*, 2020. 5

[12] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016. 7

[13] Thomas Cover and Peter Hart. Nearest neighbor pattern classification. *IEEE Trans. Inf. Theory.*, 13(1):21–27, 1967. 1, 2

[14] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *CVPRW*, pages 702–703, 2020. 6

[15] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. *NeurIPS*, 26, 2013. 4

[16] Chris Ding, Tao Li, Wei Peng, and Haesun Park. Orthogonal nonnegative matrix t-factorizations for clustering. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 126–135, 2006. 3

[17] Petros Drineas, Alan Frieze, Ravi Kannan, Santosh Vempala, and Vishwanathan Vinay. Clustering large graphs via the singular value decomposition. *Machine learning*, 56:9–33, 2004. 2, 3

[18] Thomas Fel, Agustin Picard, Louis Bethune, Thibaut Boissin, David Vigouroux, Julien Colin, Rémi Cadène, and Thomas Serre. Craft: Concept recursive activation factorization for explainability. In *CVPR*, pages 2711–2721, 2023. 3

[19] Christiane Fellbaum. *WordNet: An electronic lexical database*. MIT press, 1998. 7

[20] Jacob Goldberger, Geoffrey E Hinton, Sam Roweis, and Russ R Salakhutdinov. Neighbourhood components analysis. *NeurIPS*, 17, 2004. 1, 2

[21] Samantha Guerriero, Barbara Caputo, and Thomas Mensink. Deepncm: Deep nearest class mean classifiers. 2018. 2, 5, 6

[22] Hideaki Hayashi and Seiichi Uchida. A discriminative gaussian mixture model with sparsity. In *ICLR*, 2021. 1

[23] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 5

[24] Michael Hersche, Geethan Karunaratne, Giovanni Cherubini, Luca Benini, Abu Sebastian, and Abbas Rahimi. Constrained few-shot class-incremental learning. In *CVPR*, pages 9057–9067, 2022. 3

[25] Minyoung Huh, Pulkit Agrawal, and Alexei A Efros. What makes imagenet good for transfer learning? *arXiv preprint arXiv:1608.08614*, 2016. 7

[26] Max Kochurov, Rasul Karimov, and Serge Kozlukov. Geoopt: Riemannian optimization in pytorch. *arXiv preprint arXiv:2005.02819*, 2020. 6

[27] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical Report 0, University of Toronto, Toronto, Ontario, 2009. 4, 5, 6, 7, 8

[28] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017. 1

[29] Daniel D Lee and H Sebastian Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999. 3

[30] Oscar Li, Hao Liu, Chaofan Chen, and Cynthia Rudin. Deep learning for case-based reasoning through prototypes: A neural network that explains its predictions. In *AAAI*, volume 32, 2018. 2

[31] Shuai Li, Kui Jia, Yuxin Wen, Tongliang Liu, and Dacheng Tao. Orthogonal deep neural networks. *IEEE Trans. Pattern Anal. Mach. Intell.*, 43(4):1352–1368, 2019. 3

[32] Sun-Ao Liu, Yiheng Zhang, Zhaofan Qiu, Hongtao Xie, Yongdong Zhang, and Ting Yao. Learning orthogonal prototypes for generalized few-shot semantic segmentation. In *CVPR*, pages 11319–11328, 2023. 3

[33] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, pages 10012–10022, 2021. 5

[34] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, pages 3431–3440, 2015. 1

[35] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019. 6

[36] Chunjie Luo, Jianfeng Zhan, Xiaohe Xue, Lei Wang, Rui Ren, and Qiang Yang. Cosine normalization: Using cosine similarity instead of dot product in neural networks. In *Artificial Neural Networks and Machine Learning–ICANN 2018: 27th International Conference on Artificial Neural Networks, Rhodes, Greece, October 4-7, 2018, Proceedings, Part I 27*, pages 382–391. Springer, 2018. 3

[37] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015. 1

[38] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, 115:211–252, 2015. 2, 5, 6, 7

[39] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, June 2016. 6

[40] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008. 4, 8

[41] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011. 7

[42] Jiayun Wang, Yubei Chen, Rudrasis Chakraborty, and Stella X Yu. Orthogonal convolutional neural networks. In *CVPR*, pages 11505–11515, 2020. 3

[43] Wenguan Wang, Cheng Han, Tianfei Zhou, and Dongfang Liu. Visual recognition with deep nearest centroids. In *ICLR*, 2023. 1, 2, 3, 4, 5, 6, 7

[44] Zhirong Wu, Alexei A Efros, and Stella X Yu. Improving generalization via scalable neighborhood component analysis. In *ECCV*, pages 685–701, 2018. 1, 2, 5, 6, 7

[45] 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. In *CVPR*, pages 6176–6185, 2017. 3

[46] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *CVPR*, pages 6023–6032, 2019. 6

[47] Borui Zhang, Wenzhao Zheng, Jie Zhou, and Jiwen Lu. Bort: Towards explainable neural networks with bounded orthogonal constraint. In *ICLR*, 2023. 3

[48] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *ICLR*, 2018. 6

[49] Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation. In *AAAI*, volume 34, pages 13001–13008, 2020. 6