Inducing Neural Collapse to a Fixed Hierarchy-Aware Frame for Reducing Mistake Severity

Tong Liang, Jim Davis
Ohio State University
Columbus, Ohio 43210
{liang.693, davis.1719}@osu.edu

Abstract

There is a recently discovered and intriguing phenomenon called Neural Collapse: at the terminal phase of training a deep neural network for classification, the within-class penultimate feature means and the associated classifier vectors of all flat classes collapse to the vertices of a simplex Equiangular Tight Frame (ETF). Recent work has tried to exploit this phenomenon by fixing the related classifier weights to a pre-computed ETF to induce neural collapse and maximize the separation of the learned features when training with imbalanced data. In this work, we propose to fix the linear classifier of a deep neural network to a Hierarchy-Aware Frame (HAFrame), instead of an ETF, and use a cosine similarity-based auxiliary loss to learn hierarchy-aware penultimate features that collapse to the HAFrame. We demonstrate that our approach reduces the mistake severity of the model's predictions while maintaining its top-1 accuracy on several datasets of varying scales with hierarchies of heights ranging from 3 to 12. Code: https://github.com/ltong1130ztr/HAFrame.

1. Introduction

A recent study [32] has unveiled a phenomenon termed neural collapse. It empirically revealed that the penultimate features of the same class tend to collapse to their within-class mean. The within-class means of all classes and their respective classifier weights tend to collapse to the vertices of a simplex Equiangular Tight Frame (ETF). A simplex ETF is a geometric structure that maximally separates the pair-wise angles of the K vectors in $\mathbb{R}^d$, $d \geq K$, and the respective maximal pair-wise cosine similarity of these K vectors is $\frac{-1}{K-1}$. As illustrated in Fig. 1(a), when $K = 4$, the simplex ETF reduces to a tetrahedron, and we can visualize this tetrahedron in 3D space via PCA projection since its geometry is 3D. One can view such ETF as an embedding of a hierarchy of four classes sharing the same root node as their parent. This hierarchy is visualized in Fig. 1(c), where the four classes are equally separated regarding their hierarchical distance from each other.

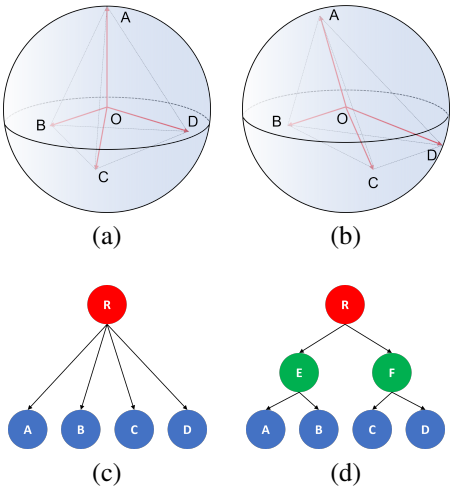


Figure 1. Illustration of a hierarchy-agnostic ETF (a) and a hierarchy-aware HAFrame (b) of four leaf classes and their hierarchies (c) and (d), respectively. All leaf classes in ETF have the same hierarchical distance.

Intuitively, the neural collapse phenomenon makes sense considering an ETF separates all classes equally and maximally from each other. However, such a structure may not emerge when trained with imbalanced data. Features of minor classes may collapse to the same vector (minority collapse) [13]. Therefore, some studies encourage the features to form an ETF structure by fixing the classifier weights at a pre-computed ETF [42] or employing additional regularizers to induce neural collapse [45, 29]. In the context of reducing mistake severity, this raises another concern: when the ETF classifier makes a mistake, it is mainly random due to its equiangular nature. Similarly, conventional neural networks are trained mainly with cross-entropy and one-hot labels, ignoring any underlying hierarchical label relationships. The associated performance evaluations also

focus on the top-1 accuracy of the predictions, treating all mistakes equally. In real-world application scenarios, some classification mistakes would have a much worse impact than others, e.g., mistaking a human for a tree in autonomous driving. Hence, it is critical to incorporate mistake severity into the performance evaluations and develop methods to reduce the mistake severity of the model predictions. One off-the-shelf way to define the severity of mistakes is by leveraging the hierarchical label relationship between incorrect predictions and their ground truths.

To impose a preferred error structure, we propose to fix the classifier vectors to a Hierarchy-Aware Frame (HAFrame) instead of an ETF to ensure that the classifier vectors of certain classes are "closer" than others. Consequently, when a mistake occurs, it is more likely to fall onto a "closer" class in the HAFrame, resulting in less mistake severity. An example of such a HAFrame is shown in Fig. 1(b). Compared to the ETF for four classes, the HAFrame captures the pair-wise hierarchical distances across the four classes from a hierarchy shown in Fig. 1(d), with class A closer to B, and A equally distant to C and D.

There are several contributions of our work:

- Our approach is easy to adapt to different hierarchies as we only require a minimal change of the classifier and no architectural change of the backbone network.
- Our approach provides an analytical solution to embed the hierarchical relationship of classes into the respective classifier.
- Our approach offers a new route to reduce mistake severity and the average hierarchical distance of predictions from the perspective of neural collapse.

2. Related Works

In recent years, multiple works have incorporated the semantic relationship of labels derived from text data or given by an explicit hierarchy to improve the classification of images or text. In this paper, we mainly discuss works closely related to incorporating hierarchical label relationships for image classification and reduction of mistake severity.

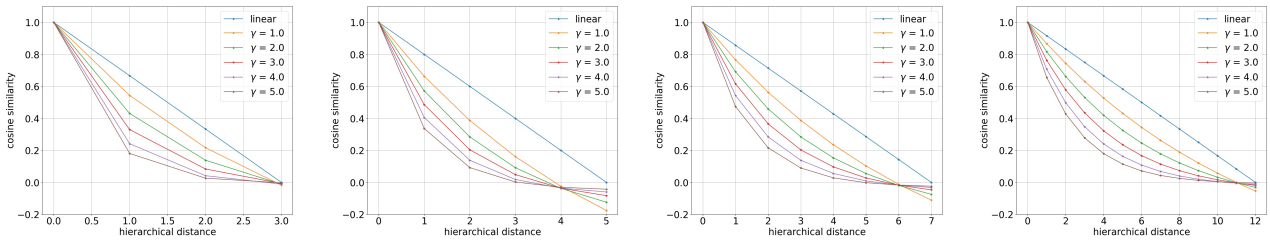
Hierarchy-aware label methods. These methods often utilize label embeddings to incorporate hierarchical label relationships into the model. In [4], soft-label embeddings derived from the hierarchical distances are proposed, and the KL-divergence from softmax scores to the soft-labels is minimized. The label embeddings capturing pair-wise similarities of the classes are fixed on a unit hypersphere in [3], or learned as hyperbolic embeddings in [43], then the image features are induced to align with the label embeddings. Aside from deriving the label embeddings from an explicit hierarchy, there are also works [14, 31, 28, 35] that model hierarchical label relationships implicitly via learning the associated semantic embeddings of the labels from text data. These methods maximize the similarity between the visual embeddings learned from images and the corresponding semantic embeddings learned from text data.

Hierarchy-aware loss methods. A Hierarchy and Exclusion (HXE) graph is proposed in [10] to model label relationships with a probabilistic classification model on the HXE graph capturing the semantic relationships (mutual exclusion, overlap, and subsumption) between any two labels. In [4], a hierarchical cross-entropy loss is proposed for the label tree. To integrate the knowledge of label relationships in a directed acyclic graph into the deep neural network, the probability of each label occurring is modeled independently to allow multi-label scenarios in [5]. In [18], the authors cast hierarchical classification as a discrete optimal transportation problem with an associated optimal transport loss.

Hierarchy-aware cost methods. This line of research derives a cost measurement for misclassifications from a given label hierarchy. The cost is then applied to amend flat predictions at inference time. In both [24] and [9], the cost is defined as the height of the lowest common ancestor between the ground truth and incorrect prediction, i.e., the semantic level at which the misclassification occurs. The associated classification problem is then formulated as a conditional risk minimization (CRM) problem. Similarly, the cost is also formulated as the loss of label specificity in [12], which optimizes the accuracy-specificity trade-offs in hierarchical classification for a given lower bound accuracy requirement.

Hierarchy-aware architecture methods. These methods require architectural changes to the network. A dynamic-structured network with a unifying hierarchy for classes of different datasets is proposed in [27]. During training, the sub-graph of the related classes is dynamically activated, incorporating data from different datasets. In [41], the authors proposed sharing the parameters of the non-leaf tree node classifiers and calibrating the posterior probability distribution of labels with a stochastic tree sampling method during training. The classifiers corresponding to nodes in a label hierarchy are constrained on hierarchically connected sphere manifolds in [37] to regularize the model performance. In [26], the classifier for each non-leaf node in the hierarchy is equipped with a virtual novel class to perform top-down classification with novelty detection.

In [40], the authors use independent classification heads for classes at different levels of the label hierarchy. All levels of classifiers share the same penultimate features, and the cross-entropy losses of all levels are optimized jointly. Similarly, a multi-classification head network is proposed in [1] to incorporate hierarchical label relationships, but each classification head is staged at a different depth of the backbone network. In [7] (Flamingo), the authors proposed to use multiple classification heads for different levels of



(a) FGVC-Aircraft [30] (b) CIFAR-100 [25] (c) iNaturalist2019 [23] (d) tiered-ImageNet-H [36]

Figure 2. Visualization of the exponential mapping functions given in Eqn. 2. The x-axis is hierarchical distance, and the y-axis is mapped cosine similarity. The linear mapping used in [3] is also plotted. The associated hierarchies used for each dataset are introduced in Sect. 4.

classes in the hierarchy, where the penultimate features are decoupled into different segments for the respective coarse and fine-grained classifiers. The multi-classification head setting is also adopted in [17] (HAFeature). The classifiers of all levels share the same penultimate features from the backbone network. Additional geometric constraints are placed on the parent and child classifiers in the hierarchy. The hierarchical relationships of labels in adjacent levels are also enforced by minimizing the Jensen-Shannon Divergence [15] between predictions of the coarse-level classifier and the soft-labels reconstructed from predictions of the next fine-level classifier.

3. Method

Inspired by prior works which fix the classifier to polytope [34, 33], Hadamard matrix [22], and simplex ETF [42], we propose to fix the linear classifier to a HAFrame with the hierarchical relationship between the leaf classes embedded into their pair-wise cosine similarities. During training, we employ a weighted loss consisting of the cross-entropy loss and the proposed cosine similarity-based auxiliary loss to induce penultimate features collapsing onto the associated classifier vectors (HAFrame) to achieve the desired reduction of mistake severity. The hierarchies required in our work are constrained to label trees, same as recent works [4, 24, 17].

3.1. Pair-wise Cosine Similarity

We use the height of the lowest common ancestor (LCA) of two leaf classes y_i and y_j in the given hierarchy as the measurement of their hierarchical distance, as used in previous works [3, 4, 24]:

$$d_{ij} = height(LCA(y_i, y_j)) \tag{1}$$

where $i, j \in \{1, 2, ..., K\}$ and K is the number of leaf classes in the hierarchy. We propose to map the pairwise hierarchical distance d_{ij} to pairwise cosine similarity S_{ij} between leaf classes y_i and y_j by an exponential mapping function:

$$S_{ij} = (1 - s_{min}) \cdot e^{-\gamma \cdot \frac{d_{ij}}{d_{max}}} + s_{min} \tag{2}$$

where d_{max} is the height of the hierarchy, $\gamma > 0$ is a hyper-parameter controlling the "spacing" between hierarchically adjacent classes, and s_{min} is a lower bound for the mapped cosine similarity: $S_{min} < S_{ij} \leq 1$.

We can construct a real-valued symmetric cosine similarity matrix $\boldsymbol{S}$, where $\boldsymbol{S}_{i,j} = \boldsymbol{S}_{j,i} = S_{ij}$. For a given γ and a set of hierarchical distances $d_{ij}, \forall i, j \in \{1, ..., K\}$, we search the minimum s_{min} between -1.0 and 1.0 with a step size of 0.02 such that the resulting $\boldsymbol{S}$ from our mapping (Eqn. 2) is *positive definite*. Other mapping functions can also be used here as long as the associated similarity matrix $\boldsymbol{S}$ is guaranteed to be positive definite.

We plotted the linear mapping function $S_{ij} = 1 - d_{ij}/d_{max}$ used in [3] and our mapping function with $\gamma = 1, 2, ..., 5$ in Fig. 2 over four datasets with increasing heights of the associated hierarchies for comparison. Our exponential mapping function is more flexible than the linear mapping function. The γ parameter allows for a trade-off between separating hierarchically close and distant classes. In terms of the resulting cosine similarities, a larger γ stretches hierarchically close classes further away while compressing hierarchically distant classes closer to each other.

3.2. Hierarchy-Aware Frame

In this section, we introduce the proposed HAFrame and how to solve it from the positive definite pair-wise cosine similarity matrix $\boldsymbol{S}$ derived from a given hierarchy.

Definition 1 (Hierarchy-Aware Frame)**.** Let a set of vectors $\{\boldsymbol{w}_i\}_{i=1}^K$ in $\mathbb{R}^K$ with $||\boldsymbol{w}_i||_2 = 1$, $i = 1, 2, ..., K$, and their pair-wise cosine similarities satisfy the following equation:

$$cos\angle(\boldsymbol{w}_i, \boldsymbol{w}_j) = \boldsymbol{w}_i^T \boldsymbol{w}_j = S_{ij}, \forall 1 \leq i \leq j \leq K \tag{3}$$

where S_{ij} is the cosine similarity between classes i and j given in Eqn. 2. Next, let $\boldsymbol{W} = [\boldsymbol{w}_1\ \boldsymbol{w}_2\ ...\ \boldsymbol{w}_K]$ be the proposed hierarchy-aware frame, $\boldsymbol{W}$ is in $\mathbb{R}^{K \times K}$ and satisfies

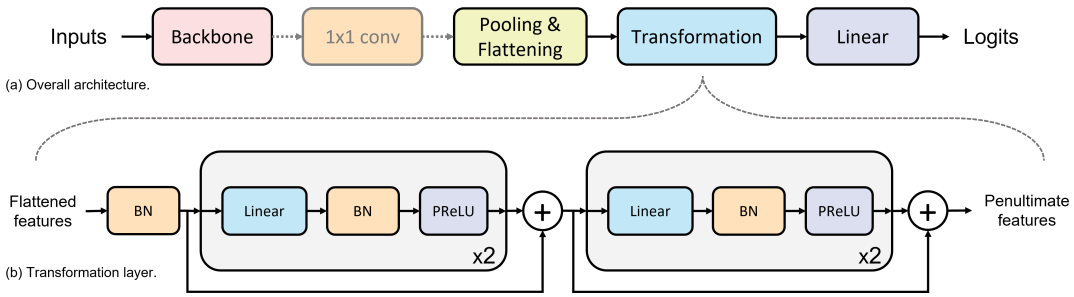


Figure 3. Illustration of the customized network architectures. (a) Top: overall network architecture of our approach, the 1x1 convolutional layer is only used in our type-II models. (b) Bottom: the proposed transformation layer, where BN is 1D batch norm layer.

$$\boldsymbol{S} = \boldsymbol{W}^T\boldsymbol{W} \tag{4}$$

Since $\boldsymbol{S}$ is guaranteed to be positive definite by our search for the proper s_{min} in the mapping function (Eqn. 2), we can find $\boldsymbol{W}$ by the following matrix factorization:

$$\boldsymbol{S} = \boldsymbol{Q}\boldsymbol{D}\boldsymbol{Q}^T = (\boldsymbol{Q}\boldsymbol{D}^{\frac{1}{2}}\boldsymbol{U}^T)(\boldsymbol{U}\boldsymbol{D}^{\frac{1}{2}}\boldsymbol{Q}^T) = \boldsymbol{W}^T\boldsymbol{W} \tag{5}$$

where $\boldsymbol{Q} \in \mathbb{R}^{K\times K}$ and $\boldsymbol{D} \in \mathbb{R}^{K\times K}$ are acquired from eigenvalue decomposition of $\boldsymbol{S}$, and $\boldsymbol{U} \in \mathbb{R}^{K\times K}$ is an orthonormal matrix obtained from QR-decomposition of a randomly sampled matrix in $\mathbb{R}^{K\times K}$ that allows an arbitrary rotation and satisfies $\boldsymbol{U}^T\boldsymbol{U} = \boldsymbol{U}\boldsymbol{U}^T = \boldsymbol{I}_K$. Therefore, the proposed hierarchy-aware frame is given by:

$$\boldsymbol{W} = \boldsymbol{U}\boldsymbol{D}^{\frac{1}{2}}\boldsymbol{Q}^T \tag{6}$$

We prove that $\boldsymbol{W}$ (i.e., our HAFrame) satisfies the *frame condition* [6] as long as the similarity matrix $\boldsymbol{S}$ is positive definite in the supplementary material.

3.3. Additional Transformation Layer

Since we are encouraging the penultimate features $\boldsymbol{h}$, instead of the centered features $\tilde{\boldsymbol{h}}$ ($\tilde{\boldsymbol{h}} = \boldsymbol{h} - \boldsymbol{h}_G$, where $\boldsymbol{h}_G$ is the average of all penultimate features) described in [32], to collapse onto their respective classifier vectors, we propose to use an additional transformation layer before the final classification layer to facilitate such collapse. The overall architecture of our model is shown in Fig. 3(a).

Our transformation layer needs to learn a mapping of the features to classifiers potentially residing in different quadrants (including the negative quadrant) of the Euclidean space $\mathbb{R}^K$. The entries of penultimate feature from most common convolutional neural networks [19, 39, 38] in recent years have an inductive bias towards non-negative values due to the use of nonlinear activation functions (e.g., ReLU [16], GELU [21], etc.). Our proposed transformation layer uses parametric ReLU (PReLU) [20], which learns the slope of the rectified linear function for negative inputs to mitigate the aforementioned bias, therefore enabling the transformed features to have negative entries approximating the respective fixed classifier vectors of $\boldsymbol{W}$. We also employ residual connections to improve information flow during training. The architecture of the transformation module is shown in Fig. 3(b). All four linear layers in the transformation module have K (number of classes) hidden units as the classifier vectors $\{\boldsymbol{w}_i\}_{i=1}^K$ of HAFrame are in $\mathbb{R}^K$. To accommodate this requirement, we add a 1x1 convolutional layer between the backbone and the pooling layer to reduce the number of channels in the backbone features to K.

3.4. Cosine Similarity Based Auxiliary Loss

Once we solve the hierarchy-aware frame $\boldsymbol{W}$ for a given pair-wise similarity matrix $\boldsymbol{S}$, the associated column vectors in $\boldsymbol{W}$ are used as weight vectors in the linear classification layer of the network. The corresponding bias term for the linear layer is removed. The prediction $\hat{y}$ is given by:

$$\hat{y} = \arg\max_i \boldsymbol{W}^T\boldsymbol{h} = \arg\max_i cos\angle(\boldsymbol{w}_i, \boldsymbol{h}) \tag{7}$$

where $\boldsymbol{h} \in \mathbb{R}^K$ is the penultimate feature vector of an input example produced by the transformation layer. The logits of an example are given by $\boldsymbol{W}^T\boldsymbol{h}$, consistent with a regular linear classifier omitting the bias term. In addition to cross-entropy loss, we propose to use a cosine similarity-based auxiliary loss to facilitate the collapse of penultimate features onto the respective classifier weights:

$$\mathcal{L}_{COS} = \sum_{i=1}^{K}(cos\angle(\boldsymbol{w}_i, \boldsymbol{h}) - cos\angle(\boldsymbol{w}_i, \boldsymbol{w}_y))^2 \tag{8}$$

where y is the ground truth of $\boldsymbol{h}$ and $\boldsymbol{w}_y$ is the fixed classifier vector corresponding to class y. This auxiliary loss reduces to zero when $cos\angle(\boldsymbol{h}, \boldsymbol{w}_y) = 1$, i.e., the penultimate feature vector $\boldsymbol{h}$ is aligned to the same direction of its classifier $\boldsymbol{w}_y$.

The overall training loss is a mixture of cross-entropy loss ($\mathcal{L}_{CE}$) and the proposed auxiliary loss with a hyperparameter $\alpha \in [0, 1]$ controlling the mixing ratio:

$$\mathcal{L} = (1-\alpha)\mathcal{L}_{CE} + \alpha\mathcal{L}_{COS} \tag{9}$$

This loss is averaged across all training examples in a mini-batch.

4. Experiments

We compare our approach with a baseline model trained with cross-entropy and recent competitive methods of CRM [24], Flamingo [7], and HAFeature [17] introduced in Sect. 2, we omit comparison with earlier approaches [14, 3, 4] as more recent works [24, 17] have demonstrated that the results of earlier methods are suboptimal. Since CRM is an inference-time approach to modifying flat predictions, we apply CRM to the predictions of the baseline model.

Datasets. Following the previous works [4, 24], we conduct comparative experiments on the tieredImageNet-H [36, 4] and iNaturalist2019 [23] datasets. We also include comparison results on CIFAR-100 [25] and FGVC-Aircraft [30]. The four datasets used in our experiments scale from 100 classes to 1010 classes, and their respective hierarchy's height ranges from 3 to 12. For FGVC-Aircraft, we adopt the original hierarchy provided by the dataset. For CIFAR-100, we use the hierarchy provided by the Flamingo approach [7]. As for iNaturalist2019 and tieredImageNet-H, we adopt their respective hierarchies from [4]. The statistics of these datasets are summarized in Table 1.

Dataset	Height	Classes	Train	Val	Test
FGVC-Aircraft	3	100	3,334	3,333	3,333
CIFAR-100	5	100	45,000	5,000	10,000
iNaturalist2019	7	1010	187,385	40,121	40,737
tieredImageNet-H	12	608	425,600	15,200	15,200

Table 1. Statistics of the four datasets used in our experiments.

Training Configurations. We use ResNet-50 [19] as the backbone network to evaluate all methods on FGVC-Aircraft, iNaturalist2019, and tieredImageNet-H. For CIFAR-100, we adopt a WideResNet-28 [44] backbone for all methods, following experimental settings in HAFeature [17]. We initialize all models on FGVC-Aircraft and iNaturalist2019 with ImageNet-1K [11] pretrained weights. The WideResNet models are trained on CIFAR-100 without pretrained weights. The previous works [4, 24, 17] all used pretrained weights from ImageNet-1K, a superset of tieredImageNet-H, to initialize models on tieredImageNet-H. We instead use pretrained weights from the PASS dataset [2] (i.e., an ImageNet replacement for self-supervised pretraining without humans) for all models on tieredImageNet-H to avoid refitting the model to the same data already seen by its pretrained weights.

We evaluate all methods with two penultimate feature extraction models for fair and sufficient comparison. The first model (type-I) follows the Flamingo/HAFeature [7, 17] settings by adding an extra transformation layer before the classification layer, similar to our approach shown in Fig. 3(a) but without the 1x1 convolutional layer before the pooling layer. This extra transformation layer consists of a linear layer with 600 hidden units and two batch norm layers before and after the linear layer. The output is rectified with ELU [8] activation. The second model (type-II, our architecture) adds a 1x1 convolutional layer before the pooling layer and replaces the type-I model's transformation layer with the proposed transformation layer introduced in Sect. 3.3.

The models across all datasets are trained for 100 epochs. The type-I models follow the training strategy of the Flamingo approach [7], i.e., using SGD optimizer with 0.01 as the initial learning rate (LR) for the backbone network and 0.1 as the initial LR for both transformation and the classification layers. The type-II models (using the proposed transformation layer) use 0.01 as the initial LR for both the backbone and transformation layer and 0.1 as the initial LR for the classification layers. We make such changes to type-II models as this yields better results. Both type-I and type-II models are trained with a cosine annealing learning rate scheduler used by the Flamingo approach [7]. The batch size of all models trained on iNaturalist2019 and tieredImageNet-H is 256. We find that a smaller batch size of 64 yields overall better results for models on CIFAR-100 and FGVC-Aircraft. During training, we select the model with the highest validation accuracy for subsequent evaluation.

Evaluation Metrics. We adopt the evaluation metrics used in previous works [4, 24, 17]: (1) top-1 accuracy; (2) average mistake severity, i.e., the sum of the heights of LCA between incorrect predictions and the respective ground truth labels averaged across all incorrect predictions; (3) average hierarchical distance at k, for $k = 1, 5, 20$, i.e., HierDist@1, HierDist@5, HierDist@20, respectively. For each test example, the average height of LCA between the ground truth label and k most likely predictions are evaluated, and the resulting average height for each test example is further averaged across the entire test set. We train five models for each method and report every evaluation metric's mean and 95% confidence interval derived from the t-distribution with four degrees of freedom.

Hyperparameter Search. We conducted a search for the proper γ in the mapping function (Eqn. 2) and the mixing ratio α between cross-entropy and our cosine similarity auxiliary loss in Eqn. 9 using the validation examples of each dataset. For FGVC-Aircraft and CIFAR-100, we conducted a grid search of γ and α on $\{1.0, 2.0, 3.0, 4.0, 5.0\} \times \{0.3, 0.4, 0.5, 0.6\}$. For iNaturalist2019 and tieredImageNet, we searched on a smaller grid $\{3.0, 4.0, 5.0\} \times \{0.3, 0.4, 0.5\}$. Each configuration of γ and α is evaluated on the validation set with three runs. We select the configuration that yields the best average hierarchical distance at 1 (HierDist@1) averaged across three runs. The resulting configurations (α, γ) for FGVC-Aircraft, CIFAR-100, iNaturalist2019, and tieredImageNet-H are (0.5, 1.0), (0.4, 2.0), (0.4, 5.0), and (0.5, 3.0), respectively.

Model	Method	Top-1 Accuracy ↑	Mistake Severity ↓	HierDist@1 ↓	HierDist@5 ↓	Hierdist@20 ↓
Type-I	cross-entropy	79.18 +/- 0.5511	2.12 +/- 0.0240	0.44 +/- 0.0097	2.10 +/- 0.0033	2.67 +/- 0.0040
	CRM [24]	79.30 +/- 0.5250	2.08 +/- 0.0201	0.43 +/- 0.0091	1.74 +/- 0.0040	2.44 +/- 0.0015
	Flamingo [7]	81.00 +/- 0.5873	2.04 +/- 0.0343	0.39 +/- 0.0072	2.06 +/- 0.0041	2.65 +/- 0.0018
	HAFeature [17]	73.23 +/- 0.6085	2.48 +/- 0.0937	0.66 +/- 0.0152	2.10 +/- 0.0126	2.61 +/- 0.0078
Type-II	cross-entropy	79.58 +/- 0.2727	2.15 +/- 0.0159	0.44 +/- 0.0067	2.11 +/- 0.0055	2.67 +/- 0.0034
	CRM [24]	79.62 +/- 0.2953	2.13 +/- 0.0109	0.43 +/- 0.0058	1.75 +/- 0.0043	2.45 +/- 0.0022
	Flamingo [7]	80.02 +/- 0.7886	2.10 +/- 0.0373	0.42 +/- 0.0168	2.08 +/- 0.0058	2.66 +/- 0.0031
	HAFeature [17]	74.39 +/- 0.7813	2.53 +/- 0.0334	0.65 +/- 0.0226	2.10 +/- 0.0064	2.61 +/- 0.0041
	HAFrame (ours)	80.49 +/- 0.4692	2.02 +/- 0.0381	0.39 +/- 0.0039	1.74 +/- 0.0027	2.45 +/- 0.0024

Table 2. Experiment results on FGVC-Aircraft dataset. The details of type-I and type-II models are included in the training config.

Model	Method	Top-1 Accuracy ↑	Mistake Severity ↓	HierDist@1 ↓	HierDist@5 ↓	Hierdist@20 ↓
Type-I	cross-entropy	77.65 +/- 0.2635	2.34 +/- 0.0271	0.52 +/- 0.0102	2.25 +/- 0.0084	3.19 +/- 0.0045
	CRM [24]	77.63 +/- 0.2800	2.30 +/- 0.0255	0.51 +/- 0.0093	1.11 +/- 0.0077	2.18 +/- 0.0028
	Flamingo [7]	77.91 +/- 0.5733	2.31 +/- 0.0179	0.51 +/- 0.0137	2.07 +/- 0.0198	3.08 +/- 0.0094
	HAFeature [17]	77.49 +/- 0.3391	2.24 +/- 0.0158	0.51 +/- 0.0084	1.43 +/- 0.0108	2.64 +/- 0.0105
Type-II	cross-entropy	76.45 +/- 0.2207	2.43 +/- 0.0235	0.57 +/- 0.0106	2.35 +/- 0.0049	3.30 +/- 0.0030
	CRM [24]	76.48 +/- 0.2278	2.38 +/- 0.0175	0.56 +/- 0.0095	1.15 +/- 0.0074	2.20 +/- 0.0029
	Flamingo [7]	75.19 +/- 0.3188	2.31 +/- 0.0270	0.57 +/- 0.0043	2.42 +/- 0.0161	3.29 +/- 0.0105
	HAFeature [17]	76.44 +/- 0.1560	2.26 +/- 0.0290	0.53 +/- 0.0055	1.71 +/- 0.0130	2.84 +/- 0.0143
	HAFrame (ours)	77.71 +/- 0.2319	2.21 +/- 0.0108	0.49 +/- 0.0066	1.11 +/- 0.0018	2.18 +/- 0.0013

Table 3. Experiment results on CIFAR-100 dataset. The details of type-I and type-II models are included in the training config.

4.1. Results

The experiment results on FGVC-Aircraft, CIFAR-100, iNaturalist2019, and tieredImageNet-H are shown in Tables 2, 3, 4, and 5, respectively. The rows in the tables highlighted with light purple are competitive methods. The method with the smallest (best) average mistake severity is selected first. Other methods with an average mistake severity not greater than the smallest mistake severity of 0.05 are also deemed competitive. Among these competitive methods, we highlight the best-performing entry for each metric with purple.

Our approach (HAFrame) has reached the best average mistake severity across all four datasets and the best top-1 accuracy on three datasets with a $\sim$0.51% (80.49%) drop of top-1 accuracy compared to the Flamingo approach (81.00%) on FGVC-Aircraft (Table 2). Our approach also performs best on the average hierarchical distance among the competitive methods on three datasets. On tieredImageNet-H (Table 5), our HierDist@5 and HierDist@20 are worse than, yet close to, CRM but still outperform Flamingo and HAFeature by a large margin. It is worth noting that CRM only reaches competitive average mistake severity on tieredImageNet-H (Table 5), but its associated average hierarchical distances at $k = 5$ and $k = 20$ remain competitive or best on all datasets. The HAFeature approach improves average mistake severity better than CRM on three datasets but does not perform well on FGVC-Aircraft (Table 2) with a shallow hierarchy of 4 levels (including the root) and does not rank predictions of less likely classes well, i.e., its HierDist@5 and HierDist@20 do not perform as good as CRM or our approach. The average mistake severity of the Flamingo approach reaches competitive results on FGVC-Aircraft and tieredImageNet-H, but its average hierarchical distances exhibit suboptimal performance similar to the HAFeature approach.

4.2. Ablation Study

In this section, we examine the effectiveness of the proposed (1) transformation layer $\mathcal{T}(.)$; (2) the fixed HAFrame classifiers (dubbed as HAF); and (3) the cosine similarity-based auxiliary loss $\mathcal{L}_{COS}$ in a cumulative fashion with the type-I model as the baseline. All variants examined in this section are trained with the same settings introduced in the previous section, including the configuration of α and γ for models with fixed HAFrame classifiers. The results are shown in Table 6. Our study shows that adding the transformation layer alone does not necessarily improve the model performance. However, fixing the corresponding classifier weights to a HAFrame improves the average mistake severity of three datasets. This also improves HierDist@5 and HierDist@20 for all four datasets. On top of these two changes, adding the auxiliary loss further facilitates the penultimate features to collapse on the HAFrame and reaches the best performance on all metrics among the variants examined except for a top-1 accuracy drop of 0.25% on iNaturalist2019.

4.3. Neural Collapse on Hierarchy-Aware Frame

In this section, we briefly introduce the metrics employed to visualize the neural collapse on an ETF employed in [32] and our extension to these metrics to visualize the collapse on a HAFrame. We compare our type-II models of fixed HAFrame classifiers with type-I baseline models of learnable classifiers from the perspective of HAFrame collapse and ETF collapse, respectively. We checkpoint the

Model	Method	Top-1 Accuracy ↑	Mistake Severity ↓	HierDist@1 ↓	HierDist@5 ↓	Hierdist@20 ↓
Type-I	cross-entropy	70.68 +/- 0.2097	2.22 +/- 0.0103	0.65 +/- 0.0068	1.95 +/- 0.0043	3.37 +/- 0.0040
	CRM [24]	70.67 +/- 0.2095	2.16 +/- 0.0045	0.63 +/- 0.0057	1.17 +/- 0.0042	1.75 +/- 0.0033
	Flamingo [7]	70.11 +/- 0.1119	2.13 +/- 0.0063	0.64 +/- 0.0014	1.79 +/- 0.0126	3.28 +/- 0.0114
	HAFeature [17]	70.57 +/- 0.1645	2.13 +/- 0.0192	0.63 +/- 0.0045	1.55 +/- 0.2188	2.68 +/- 0.4208
Type-II	cross-entropy	70.44 +/- 0.1576	2.26 +/- 0.0071	0.67 +/- 0.0036	1.97 +/- 0.0060	3.40 +/- 0.0070
	CRM [24]	70.47 +/- 0.1363	2.21 +/- 0.0099	0.65 +/- 0.0029	1.18 +/- 0.0020	1.76 +/- 0.0016
	Flamingo [7]	70.13 +/- 0.1499	2.15 +/- 0.0061	0.64 +/- 0.0045	1.76 +/- 0.0037	3.31 +/- 0.0071
	HAFeature [17]	68.46 +/- 4.6278	2.21 +/- 0.1298	0.70 +/- 0.1501	1.50 +/- 0.1235	2.49 +/- 0.0842
	HAFrame (ours)	70.89 +/- 0.1213	2.04 +/- 0.0107	0.59 +/- 0.0033	1.14 +/- 0.0033	1.73 +/- 0.0023

Table 4. Experiment results on iNaturalist2019 dataset. The details of type-I and type-II models are included in the training config.

Model	Method	Top-1 Accuracy ↑	Mistake Severity ↓	HierDist@1 ↓	HierDist@5 ↓	Hierdist@20 ↓
Type-I	cross-entropy	73.63 +/- 0.1165	6.94 +/- 0.0208	1.83 +/- 0.0117	5.70 +/- 0.0192	7.34 +/- 0.0291
	CRM [24]	73.54 +/- 0.1495	6.89 +/- 0.0272	1.82 +/- 0.0155	4.82 +/- 0.0062	6.03 +/- 0.0041
	Flamingo [7]	72.34 +/- 0.1488	6.93 +/- 0.0391	1.92 +/- 0.0135	5.75 +/- 0.0130	7.41 +/- 0.0098
	HAFeature [17]	73.52 +/- 0.1613	6.89 +/- 0.0281	1.82 +/- 0.0125	5.52 +/- 0.0176	6.95 +/- 0.0120
Type-II	cross-entropy	72.51 +/- 0.4317	6.95 +/- 0.0298	1.91 +/- 0.0338	5.69 +/- 0.0085	7.28 +/- 0.0082
	CRM [24]	72.45 +/- 0.4077	6.90 +/- 0.0274	1.90 +/- 0.0308	4.85 +/- 0.0090	6.05 +/- 0.0057
	Flamingo [7]	66.46 +/- 1.1572	7.05 +/- 0.0319	2.36 +/- 0.0921	5.77 +/- 0.0220	7.31 +/- 0.0120
	HAFeature [17]	68.32 +/- 0.9225	7.04 +/- 0.0356	2.23 +/- 0.0741	5.62 +/- 0.0223	6.97 +/- 0.0105
	HAFrame (ours)	74.00 +/- 0.3549	6.89 +/- 0.0251	1.79 +/- 0.0216	4.94 +/- 0.0118	6.15 +/- 0.0065

Table 5. Experiment results on tieredImageNet-H dataset. The details of type-I and type-II models are included in the training config.

baseline and our models every five epochs during training of 100 epochs with the same settings as our previous experiments.

Angular collapse. The classifiers and class means of training features should approach the ideal pair-wise angle during the neural collapse. Therefore, the respective pair-wise cosine similarities should approach the ideal cosine similarity $\hat{S}_{ij}$. We monitor the average of pair-wise cosine similarities during the training process to check if they are reaching the expected angle:

$$Avg_{1\leq i<j\leq K}(|cos\angle(\boldsymbol{x}_i, \boldsymbol{x}_j) - \hat{S}_{ij}|) \tag{10}$$

where $\boldsymbol{x}_i$, for $i = 1, ..., K$, are the set of K vectors collapsing to the respective frame. They can either be the classifier weights or class means of the penultimate features (centered $\tilde{\boldsymbol{h}}$ for ETF, not centered $\boldsymbol{h}$ for HAFrame). We also track the standard deviation of pair-wise cosine similarities during the training process:

$$Std_{1\leq i<j\leq K}(|cos\angle(\boldsymbol{x}_i, \boldsymbol{x}_j) - \hat{S}_{ij}|) \tag{11}$$

For ETF collapse, the ideal cosine similarity between any pair of classes is equal to $\frac{-1}{K-1}$, i.e., $\hat{S}_{ij} = \frac{-1}{K-1}$ for $\forall i, j \in \{1, ..., K\}$ and $i \neq j$ (equiangularity of ETF). The proposed HAFrame encodes the hierarchical relationship between classes into the pair-wise cosine similarities of the classes. Therefore, the ideal cosine similarity in Eqn. 10 and Eqn. 11 is given by our mapping function in Eqn. 2, and we have $\hat{S}_{ij} = S_{ij}$ for HAFrame collapse. Both the mean and standard deviation in Eqn. 10 and Eqn. 11 are approaching zero if neural collapse on an ETF or HAFrame occurs.

The visualization results are shown in the top two rows of Fig. 4. Since we fixed our classifier to the pre-computed HAFrame, the associated means and standard deviations are always zeros. The features extracted from our HAFrame models reach smaller means (top row of Fig. 4) and standard deviations (middle row of Fig. 4) on all four datasets compared to the baseline, demonstrating the effectiveness of our approach.

Self-Duality. During training, the class feature means of training examples should converge to the respective classifier vectors. Therefore, the means and classifiers become self-dual. Following [32], we visualize self-duality by measuring the Frobenius Norm of the difference between classifiers and class feature means:

$$\left\| \frac{\boldsymbol{W}}{||\boldsymbol{W}||_F} - \frac{\boldsymbol{H}}{||\boldsymbol{H}||_F} \right\|_F \tag{12}$$

where $\boldsymbol{W}$ is the classifier weight matrix, and $\boldsymbol{H} = [\boldsymbol{h}_1, ..., \boldsymbol{h}_K]$ is a matrix of class feature means of training examples. This norm approaches zero as the features collapse to the associated classifiers.

The visualization results are shown in the bottom row of Fig. 4. Both the baseline and our model reached self-duality on FGVC-Aircraft and CIFAR-100, and they are still approaching self-duality on the larger two datasets. We observed that more training epochs (e.g., 200 or 350 epochs) lead to better self-duality on the larger datasets with minor or no performance improvements.

5. Conclusion

Our proposed approach maps the pair-wise hierarchical distances of the flat classes into their associated cosine similarities and provides an analytical solution to the proposed hierarchy-aware frame for a given similarity matrix. The

dataset	model	$\mathcal{L}_{CE}$	$\mathcal{T}(.)$	HAF	$\mathcal{L}_{COS}$	Top-1 Acc ↑	Mistake Severity ↓	HieDist@1 ↓	HieDist@5 ↓	Hiedist@20 ↓
FGVC-Aircraft	ResNet50	✓	✗	✗	✗	79.18 +/- 0.5511	2.12 +/- 0.0240	0.44 +/- 0.0097	2.10 +/- 0.0033	2.67 +/- 0.0040
	ResNet50*	✓	✓	✗	✗	79.58 +/- 0.2727	2.15 +/- 0.0159	0.44 +/- 0.0067	2.11 +/- 0.0055	2.67 +/- 0.0034
	ResNet50*	✓	✓	✓	✗	79.18 +/- 0.5347	2.08 +/- 0.0299	0.43 +/- 0.0145	1.90 +/- 0.0056	2.55 +/- 0.0101
	ours	✓	✓	✓	✓	**80.49 +/- 0.4692**	**2.02 +/- 0.0381**	**0.39 +/- 0.0039**	**1.74 +/- 0.0027**	**2.45 +/- 0.0024**
CIFAR-100	WideResNet28	✓	✗	✗	✗	77.65 +/- 0.2635	2.34 +/- 0.0271	0.52 +/- 0.0102	2.25 +/- 0.0084	3.19 +/- 0.0045
	WideResNet28*	✓	✓	✗	✗	76.45 +/- 0.2207	2.43 +/- 0.0235	0.57 +/- 0.0106	2.35 +/- 0.0049	3.30 +/- 0.0030
	WideResNet28*	✓	✓	✓	✗	77.30 +/- 0.3798	2.37 +/- 0.0131	0.54 +/- 0.0111	1.59 +/- 0.0108	2.71 +/- 0.0147
	ours	✓	✓	✓	✓	**77.71 +/- 0.2319**	**2.21 +/- 0.0108**	**0.49 +/- 0.0066**	**1.11 +/- 0.0018**	**2.18 +/- 0.0013**
iNatura-list2019	ResNet50	✓	✗	✗	✗	70.68 +/- 0.2097	2.22 +/- 0.0103	0.65 +/- 0.0068	1.95 +/- 0.0043	3.37 +/- 0.0040
	ResNet50*	✓	✓	✗	✗	70.44 +/- 0.1576	2.26 +/- 0.0071	0.67 +/- 0.0036	1.97 +/- 0.0060	3.40 +/- 0.0070
	ResNet50*	✓	✓	✓	✗	**71.14 +/- 0.2245**	2.19 +/- 0.0132	0.63 +/- 0.0025	1.39 +/- 0.0021	2.20 +/- 0.0048
	ours	✓	✓	✓	✓	70.89 +/- 0.1213	**2.04 +/- 0.0107**	**0.59 +/- 0.0033**	**1.14 +/- 0.0033**	**1.73 +/- 0.0023**
tiered-ImageNet -H	ResNet50	✓	✗	✗	✗	73.63 +/- 0.1165	6.94 +/- 0.0208	1.83 +/- 0.0117	5.70 +/- 0.0192	7.34 +/- 0.0291
	ResNet50*	✓	✓	✗	✗	72.51 +/- 0.4317	6.95 +/- 0.0298	1.91 +/- 0.0338	5.69 +/- 0.0085	7.28 +/- 0.0082
	ResNet50*	✓	✓	✓	✗	73.54 +/- 0.2328	6.93 +/- 0.0274	1.83 +/- 0.0117	5.45 +/- 0.0026	6.82 +/- 0.0048
	ours	✓	✓	✓	✓	**74.00 +/- 0.3549**	**6.89 +/- 0.0251**	**1.79 +/- 0.0216**	**4.94 +/- 0.0118**	**6.15 +/- 0.0065**

Table 6. The ablation study results for FGVC-Aircraft (1st row), and CIFAR-100 (2nd row), iNaturalist2019 (3rd row), tieredImageNet-H (4th row). Each row in the table corresponds to the average results of 5 runs with a 95% confidence interval. Both ResNet50 and WideResNet28 are customized type-I models. The ResNet50*, WideResNet28*, and ours are customized type-II models.

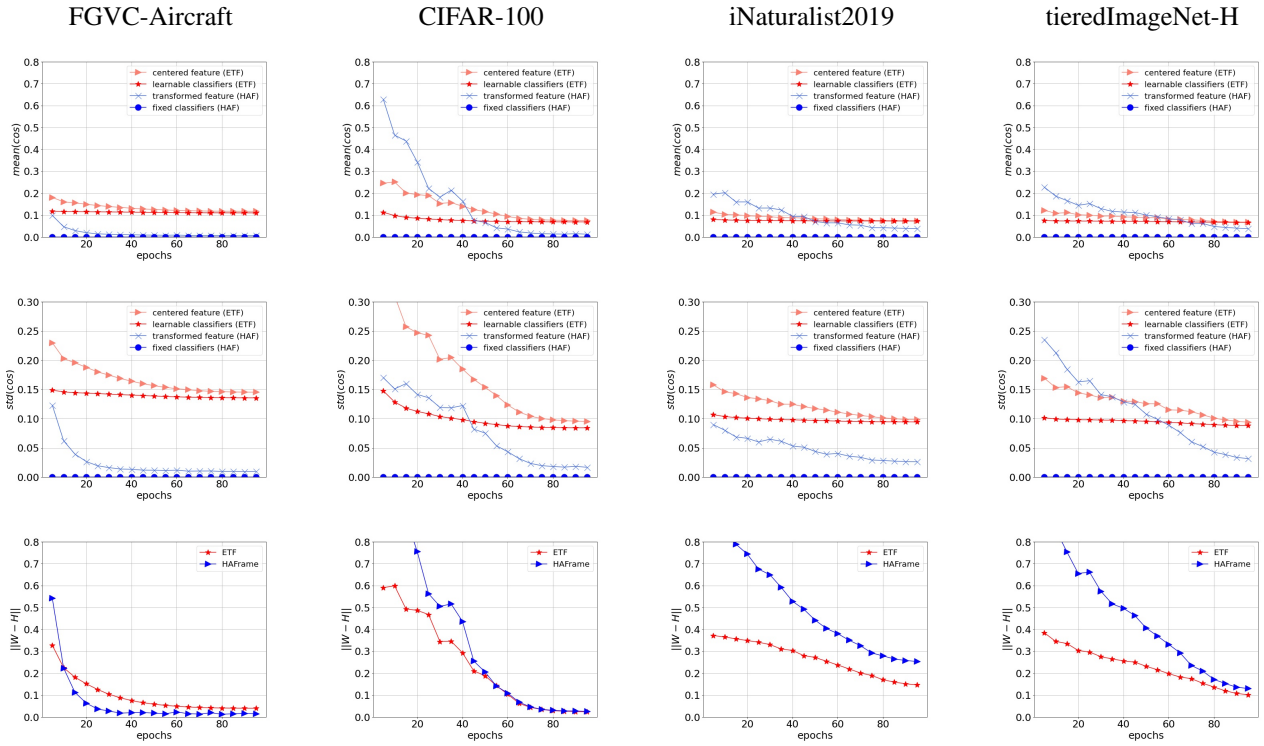


Figure 4. Neural collapse visualization: HAFrame (blue) vs. ETF (red) on all four datasets. The x-axis is the epoch number, and the y-axis is either the mean (top row), the standard deviation (middle row), or the self-duality (bottom row) introduced in Sect. 4.3. The number of hidden units in the transformation layer of type-I models is increased to 1010 and 608 for the respective models of iNaturalist2019 and tieredImageNet-H to meet the penultimate feature dimension's requirement ($d \geq K$) for ETF collapse.

proposed approach is easy to implement as it only requires an extra 1x1 conv layer, a plug-in transformation layer, and freezing of the respective classifier to a HAFrame. Therefore, it is also easy to adapt to different hierarchies. Our approach offers a new route to reduce the mistake severity of model predictions from the neural collapse point of view. We examined the proposed approach through extensive experiments and our approach consistently reaches the best mistake severity while maintaining competitive classification accuracy (best on 3/4 datasets) and average hierarchical distances (best on 3/4 datasets). Future work may seek loss functions that better facilitate the collapse of penultimate features on the HAFrame or further optimize the architecture of the transformation layer to improve performance.

References

[1] Bilal Alsallakh, Amin Jourabloo, Mao Ye, Xiaoming Liu, and Liu Ren. Do Convolutional Neural Networks Learn Class Hierarchy? *IEEE Transactions on Visualization and Computer Graphics*, 24:152–162, 2017.

[2] Yuki M. Asano, Christian Rupprecht, Andrew Zisserman, and Andrea Vedaldi. PASS: An ImageNet Replacement for Self-supervised Pretraining Without Humans. *NeurIPS Track on Datasets and Benchmarks*, 2021.

[3] Björn Barz and Joachim Denzler. Hierarchy-Based Image Embeddings for Semantic Image Retrieval. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 638–647, 2019.

[4] Luca Bertinetto, Romain Mueller, Konstantinos Tertikas, Sina Samangooei, and Nicholas A. Lord. Making better mistakes: Leveraging class hierarchies with deep networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[5] Clemens-Alexander Brust and Joachim Denzler. Integrating Domain Knowledge: Using Hierarchies to Improve Deep Classifiers. In *Asian Conference on Pattern Recognition (ACPR)*, 2019.

[6] Peter G. Casazza, Gitta Kutyniok, and Friedrich Philipp. *Introduction to Finite Frame Theory*, pages 1–53. Birkhäuser Boston, Boston, 2013.

[7] Dongliang Chang, Kaiyue Pang, Yixiao Zheng, Zhanyu Ma, Yi-Zhe Song, and Jun Guo. Your "Flamingo" is My "Bird": Fine-Grained, or Not. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[8] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs). In *International Conference on Learning Representations (ICLR)*, 2016.

[9] Jia Deng, Alexander Berg, Kai Li, and Fei-Fei Li. What Does Classifying More Than 10, 000 Image Categories Tell Us? In *European Conference on Computer Vision (ECCV)*, volume 5, pages 71–84, 12 2010.

[10] Jia Deng, Nan Ding, Yangqing Jia, Andrea Frome, Kevin Murphy, Samy Bengio, Yuan Li, Hartmut Neven, and Hartwig Adam. Large-Scale Object Classification Using Label Relation Graphs. In *European Conference on Computer Vision (ECCV)*, 2014.

[11] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A Large-scale Hierarchical Image Database. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 248–255. Ieee, 2009.

[12] Jia Deng, J. Krause, A.C. Berg, and Li Fei-Fei. Hedging your bets: Optimizing accuracy-specificity trade-offs in large scale visual recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[13] Cong Fang, Hangfeng He, Qi Long, and Weijie J. Su. Exploring deep neural networks via layer-peeled model: Minority collapse in imbalanced training. *Proceedings of the National Academy of Sciences of the United States of America (PNAS)*, 2021.

[14] Andrea Frome, Greg S Corrado, Jon Shlens, Samy Bengio, Jeff Dean, Marc' Aurelio Ranzato, and Tomas Mikolov. DeViSE: A Deep Visual-Semantic Embedding Model. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2013.

[15] B. Fuglede and F. Topsoe. Jensen-Shannon Divergence and Hilbert Space Embedding. In *International Symposium on-Information Theory, 2004. ISIT 2004. Proceedings.*, pages 31–, 2004.

[16] Kunihiko Fukushima. Cognitron: A self-organizing multilayered neural network. *Biological Cybernetics*, 20:121–136, 1975.

[17] Ashima Garg, Depanshu Sani, and Saket Anand. Learning Hierarchy Aware Features for Reducing Mistake Severity. In *European Conference on Computer Vision (ECCV)*, 2022.

[18] Yubin Ge, Site Li, Xuyang Li, Fangfang Fan, Wanqing Xie, Jane Jia You, and Xiaofeng Liu. Embedding Semantic Hierarchy in Discrete Optimal Transport for Risk Minimization. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2835–2839, 2021.

[19] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[20] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. In *International Conference on Computer Vision (ICCV)*, 2015.

[21] Dan Hendrycks and Kevin Gimpel. Gaussian Error Linear Units (GELUs). *arXiv*, 2016.

[22] Elad Hoffer, Itay Hubara, and Daniel Soudry. Fix Your Classifier: The Marginal Value of Training The Last Weight Layer. In *International Conference on Learning Representations (ICLR)*, 2018.

[23] Grant Van Horn, Oisin Mac Aodha, Yang Song, Yin Cui, Chen Sun, Alexander Shepard, Hartwig Adam, Pietro Perona, and Serge J. Belongie. The iNaturalist Species Classification and Detection Dataset. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[24] Shyamgopal Karthik, Ameya Prabhu, Puneet K. Dokania, and Vineet Gandhi. No Cost Likelihood Manipulation at Test Time for Making Better Mistakes in Deep Networks. In *International Conference on Learning Representations (ICLR)*, 2021.

[25] Alex Krizhevsky. Learning Multiple Layers of Features from Tiny Images. *University of Toronto*, 05 2012.

[26] Kibok Lee, Kimin Lee, Kyle Min, Yuting Zhang, Jinwoo Shin, and Honglak Lee. Hierarchical Novelty Detection for Visual Object Recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[27] Xiaodan Liang, Hongfei Zhou, and Eric Xing. Dynamic-structured Semantic Propagation Network. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[28] Shaoteng Liu, Jingjing Chen, Liangming Pan, Chong-Wah Ngo, Tat-Seng Chua, and Yu-Gang Jiang. Hyperbolic Visual Embedding Learning for Zero-Shot Recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[29] Xuantong Liu, Jianfeng Zhang, Tianyang Hu, He Cao, Lujia Pan, and Yuan Yao. Inducing Neural Collapse in Deep Long-tailed Learning. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2023.

[30] S. Maji, J. Kannala, E. Rahtu, M. Blaschko, and A. Vedaldi. Fine-Grained Visual Classification of Aircraft. *arXiv*, 2013.

[31] Mohammad Norouzi, Tomas Mikolov, Samy Bengio, Yoram Singer, Jonathon Shlens, Andrea Frome, Gregory S. Corrado, and Jeffrey Dean. Zero-Shot Learning by Convex Combination of Semantic Embeddings. In *International Conference on Learning Representations (ICLR)*, 2013.

[32] Vardan Papyan, X.Y. Han, and David Donoho. Prevalence of Neural Collapse During the Terminal Phase of Deep Learning Training. *Proceedings of the National Academy of Sciences of the United States of America (PNAS)*, 117, 09 2020.

[33] Federico Pernici, Matteo Bruni, Claudio Baecchi, and A. Bimbo. Regular Polytope Networks. *IEEE Transactions on Neural Networks and Learning Systems*, 33:4373–4387, 2021.

[34] Federico Pernici, Matteo Bruni, Claudio Baecchi, and Alberto Del Bimbo. Maximally Compact and Separated Features with Regular Polytope Networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2019.

[35] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning Transferable Visual Models From Natural Language Supervision. *arXiv*, 2021.

[36] Mengye Ren, Eleni Triantafillou, Sachin Ravi, Jake Snell, Kevin Swersky, Joshua B. Tenenbaum, Hugo Larochelle, and Richard S. Zemel. Meta-Learning for Semi-Supervised Few-Shot Classification. In *International Conference on Learning Representations (ICLR)*, 2018.

[37] Damien Scieur and Youngsung Kim. Connecting Sphere Manifolds Hierarchically for Regularization. In *International Conference on Machine Learning (ICML)*, 2021.

[38] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the Inception Architecture for Computer Vision. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[39] Mingxing Tan and Quoc Le. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, 2019.

[40] Hui Wu, Michele Merler, Rosario Uceda-Sosa, and John R. Smith. Learning to Make Better Mistakes: Semantics-Aware Visual Food Recognition. In *ACM International Conference on Multimedia*, 2016.

[41] Tz-Ying Wu, Pedro Morgado, Pei Wang, Chih-Hui Ho, and Nuno Vasconcelos. Solving Long-tailed Recognition with Deep Realistic Taxonomic Classifier. In *European Conference on Computer Vision (ECCV)*, 07 2020.

[42] Yibo Yang, Liang Xie, Shixiang Chen, Xiangtai Li, Zhouchen Lin, and Dacheng Tao. Inducing Neural Collapse in Imbalanced Learning: Do We Really Need a Learnable Classifier at the End of Deep Neural Network? *Conference on Neural Information Processing Systems (NeurIPS)*, 2022.

[43] Zhen Yu, Toan Nguyen, Yaniv Gal, Lie Ju, Shekhar S. Chandra, Lei Zhang, Paul Bonnington, Victoria Mar, Zhiyong Wang, and Zongyuan Ge. Skin lesion recognition with class-hierarchy regularized hyperbolic embeddings. In Linwei Wang, Qi Dou, P. Thomas Fletcher, Stefanie Speidel, and Shuo Li, editors, *Medical Image Computing and Computer Assisted Intervention – MICCAI 2022*, pages 594–603, Cham, 2022. Springer Nature Switzerland.

[44] Sergey Zagoruyko and Nikos Komodakis. Wide Residual Networks. *arXiv*, 2016.

[45] Zhisheng Zhong, Jiequan Cui, Yibo Yang, Xiaoyang Wu, Xiaojuan Qi, Xiangyu Zhang, and Jiaya Jia. Understanding Imbalanced Semantic Segmentation Through Neural Collapse. *arXiv*, 2023.