Stationary Representations: Optimally Approximating Compatibility and Implications for Improved Model Replacements

Supplementary Material

A. Stationarity-Compatibility Theorem

Before proceeding to the main theorem, a key Lemma is established. This Lemma, concerning the probability of a random point on a surface cap of a hypersphere, plays an essential role in the subsequent discussion.

Lemma 1 *Let $\mathbf{w}_i \in \mathbb{R}^d$ for $i = 1, \ldots, n$ be i.i.d. vectors from the uniform distribution on the unit hypersphere. Then the probability $P_{n,d}$ of a random vector on a hypersphere cap around $\mathbf{w}_i$ is given by:*

$$P_{n,d} = \frac{1}{\sqrt{\pi}} \cdot \sin\left(\theta_{n,d}\right)^{d-2} \cdot \frac{\Gamma\left(\frac{d}{2}\right)}{\Gamma\left(\frac{d}{2} - \frac{1}{2}\right)} \tag{1}$$

where $\theta_{n,d}$ is the expected angle from a vector $\mathbf{w}_i$ to its nearest neighbor.

Proof. We begin by noting that the probability P of a random point on a hypersphere cap around prototype $\mathbf{w}_i$ is given by the ratio of the cap surface to the hypersphere's surface area. This can be approximated as $P = \frac{A_{\text{disc}}}{A}$ where A_{disc} is the area of the disc locally approximating the cap around the prototype $\mathbf{w}_i$. The surface area A of a hypersphere in d dimensions is given by

$$A = 2\pi^{d/2} \frac{R^{d-1}}{\Gamma(d/2)}$$

and the hyperarea A_{disc} of the disc is

$$A_{\text{disc}} = 2\pi^{(d-1)/2} \frac{r^{d-2}}{\Gamma((d-1)/2)}.$$

This leads to the simplified expression for the probability P of a random point on a disc on a hypersphere:

$$P = \frac{r^{d-2} \cdot R^{1-d} \cdot \Gamma\left(\frac{d}{2}\right)}{\sqrt{\pi} \cdot \Gamma\left(\frac{d}{2} - \frac{1}{2}\right)}. \tag{2}$$

Where r is the radius of the surface disc (locally approximating the cap), R is the radius of the hypersphere, d is the number of dimensions, and Γ is the gamma function. Using spherical coordinates, the relationship between R, r, and the polar angle θ is $r = R\sin(\theta)$. We use $\theta_{n,d}$ as described in [1] and [2] to denote the dependencies on n and d:

$$\theta_{n,d} = n^{-\frac{2}{d-1}} \Gamma\left(1 + \frac{1}{d-1}\right) \left(\frac{\Gamma\left(\frac{d}{2}\right)}{2\sqrt{\pi}(d-1)\Gamma\left(\frac{d-1}{2}\right)}\right)^{-\frac{1}{d-1}}. \tag{3}$$

Substituting $r = R\sin(\theta_{n,d})$ into the probability P of Eq. 2 and, considering the unit hypersphere $R = 1$, we get Eq. 1. This highlights the dependencies of the probability on both the number of prototypes n and their dimension d. □

Lemma 1 is used to demonstrate Theorem 1 that is reported in the following for better comprehension. It is noteworthy that a disc in high dimensional space can be considered a hyperball when referring to its filled volume.

Theorem 1 (Stationarity $\implies$ Compatibility) *Let $\mathbf{W} = [\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_K]$ be the $d \times K$ matrix of a d-Simplex fixed classifier. Given two tasks, $\mathcal{T}_k$ and $\mathcal{T}_t$. The task $\mathcal{T}_t$ is derived from $\mathcal{T}_k$ by incorporating an additional training set $\Delta\mathcal{T}$, such that $\mathcal{T}_t = \mathcal{T}_k \cup \Delta\mathcal{T}$. The combined task, $\mathcal{T}_t$, comprises a set of classes each denoted by y, where $y \in \{1, 2, \ldots, K_t\}$ and $K_t < K$. Under the assumption that learning the new task $\mathcal{T}_t$ causes the hyperball $\mathcal{B}_k(\mathbf{w}_y)$ with radius r_k^y to shrink into a smaller hyperball $\mathcal{B}_t(\mathbf{w}_y)$, i.e., $r_t^y \leq r_k^y$ for all y in the set $\{1, 2, \ldots, K_k\}$, then it necessarily follows that ϕ_t and ϕ_k optimally approximate the compatibility inequality constraints as defined in Def. 1 in expectation.*

Proof. Let $\phi_t(\mathbf{x})$ and $\phi_k(\mathbf{x})$ be random variables representing the learned representations up to the t-th and the k-th task, respectively. We assume that these variables are distributed within hyperballs denoted as $\mathcal{B}_t(\mathbf{w}_y)$ and $\mathcal{B}_k(\mathbf{w}_y)$, where y is a generic class label, according to the joint probability density function $f_{\phi_t(\mathbf{x}),\phi_k(\mathbf{x})}$. Hyperballs are centered at the d-Simplex classifier prototype $\mathbf{w}_y$ and are defined as:

$$\mathcal{B}_t(\mathbf{w}_y) = \{\phi_t(\mathbf{x}) \in \mathbb{R}^d : ||\phi_t(\mathbf{x}) - \mathbf{w}_y||_2 \leq r_t^y\}, \tag{4}$$

$$\mathcal{B}_k(\mathbf{w}_y) = \{\phi_k(\mathbf{x}) \in \mathbb{R}^d : ||\phi_k(\mathbf{x}) - \mathbf{w}_y||_2 \leq r_k^y\} \tag{5}$$

being r_t^y and r_k^y the radii of $\mathcal{B}_t(\mathbf{w}_y)$ and $\mathcal{B}_k(\mathbf{w}_y)$, respectively. The distance between the two random variables $\phi_t(\mathbf{x}_a)$ and $\phi_k(\mathbf{x}_b)$ is a new random variable:

$$D_{k,t} = ||\phi_t(\mathbf{x}_a) - \phi_k(\mathbf{x}_b)||. \tag{6}$$

Verification in expectation of the compatibility definition of Def. 1 requires the evaluation of $D_{k,t}$, i.e, $\mathbb{E}[||\phi_k(\mathbf{x}_a) - \phi_t(\mathbf{x}_b)||]$, and compare it with the expected value of $D_{k,k}$, i.e., $\mathbb{E}[||\phi_k(\mathbf{x}_a) - \phi_k(\mathbf{x}_b)||]$. Defining the function g as:

$$g\left(x_a, x_b\right) = ||x_a - x_b||,$$

the expected value $\mathbb{E}[D_{k,t}]$ of Eq. 6 is given by:

$$\mathbb{E}[D_{k,t}] = \int_{\mathcal{B}_k^{y_i}} \int_{\mathcal{B}_t^{y_j}} g\left(x_a, x_b\right) f_{\phi_k,\phi_t}\left(x_a, x_b\right) dV(x_a) dV(x_b) \tag{7}$$

where y_i and y_j denote the classes associated with x_a and x_a, respectively, and $\mathcal{B}_k^{y_i}$, $\mathcal{B}_t^{y_j}$, and f_{ϕ_t,ϕ_k} are simplified notations for $\mathcal{B}_k(\mathbf{w}_{y_i})$, $\mathcal{B}_t(\mathbf{w}_{y_j})$, and $f_{\phi_t(\mathbf{x}),\phi_k(\mathbf{x})}$, respectively.

Eq. 7 is evaluated under the following assumptions: (1) UFM [3], which allows features of a model to be considered independent. (2) The hypothesis of a d-Simplex fixed classifier. This assumption allows focusing on a single pairwise class interaction, as interactions with all other classes are symmetrically similar and fixed. (3) Since $\phi_t(\mathbf{x})$ and $\phi_k(\mathbf{x})$ are derived from training two separate models, they are treated as independent random variables, each distributed according to $f_{\phi_t(\mathbf{x})}$ and $f_{\phi_k(\mathbf{x})}$, respectively. As a consequence, the joint probability density function can be substituted by the product of the probability density functions of $\phi_k(\mathbf{x})$ and $\phi_t(\mathbf{x})$, i.e., $f_{\phi_k(\mathbf{x}),\phi_t(\mathbf{x})}\left(x_a, x_b\right) = f_{\phi_k(\mathbf{x})}\left(x_a\right) f_{\phi_t(\mathbf{x})}\left(x_b\right)$ and integral of Eq. 7 reduces to:

$$\mathbb{E}[D_{k,t}] = \int_{\mathcal{B}_k^{y_j}} \int_{\mathcal{B}_t^{y_i}} ||x_a - x_b|| f_{\phi_k}\left(x_a\right) f_{\phi_t}\left(x_b\right) dV(x_a) dV(x_b). \tag{8}$$

Lemma 1 allows for the case-by-case evaluation of Equation 8 in the case of assessing the alignment and compatibility of class prototypes in trainable and non-trainable classifiers. From the Lemma it follows that when retraining a model from scratch in which the classifier is trainable, the probability of class prototypes falling, according to Nearest Neighbor rule, within their corresponding hyperballs of a previously trained model decreases exponentially as both dimensionality and the number of classes for training increases (Fig. 1). Following the definition of Eq. 1a, the conditions for optimal compatibility between prototypes of corresponding classes in both models are realized when their distance reaches its minimum value. This occurs when they are perfectly aligned. In this case, classes will not manifest randomly and the probability of them falling within the same regions does not decrease exponentially.

Eq. 3 in Lemma 1, also indicates that the introduction of new classes results in a decrease in the angles between them, a phenomenon also shown in [2]. Assuming two perfectly aligned models, the introduction of new classes in one of the models results in two effects: a decrease in intraclass and interclass distances between features. Such reductions in distance indicate a deviation from the concentric arrangement between of corresponding class hyperballs in the two models, leading to a compromise of the conditions for optimal compatibility. While one might consider pre-allocating a large number of classes to leverage a broader representation space

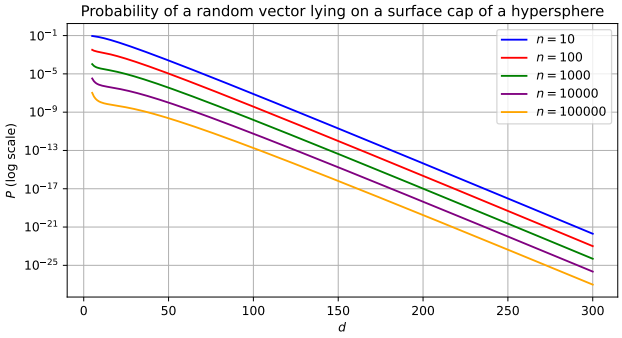


Figure 1. The probability P of Eq. 1 of a point lying within a disc on a hypersphere's surface. Different curves (logarithmic scale) correspond to varying numbers of points sampled (n), across a dimension range (d). The plot shows that as the dimension and the number of points increases, the probability decreases significantly, reflecting the curse of dimensionality.

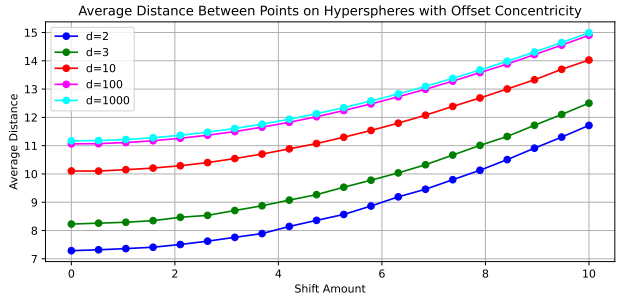
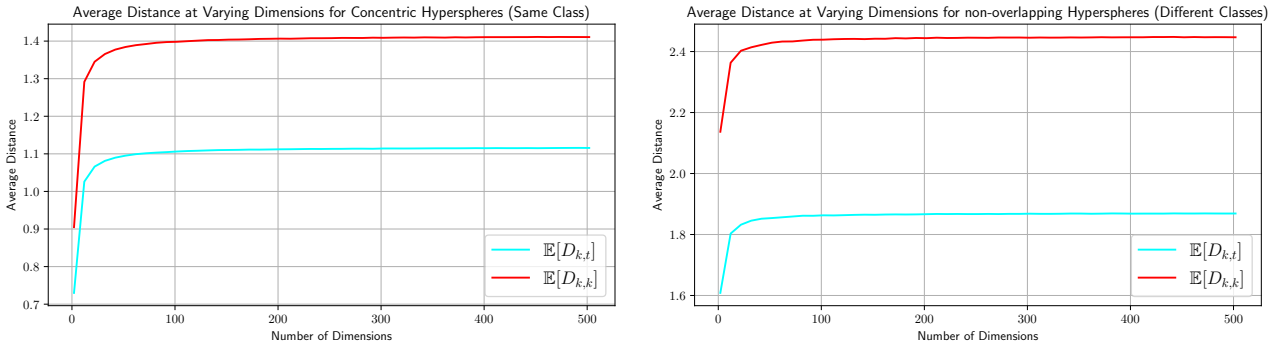


Figure 2. Expected distance of Eq. 8 between points on two closely aligned (or nearly concentric) hyperballs. Distance increases by shifting one of the hyperballs showing that optimality (i.e. less distance variation) is when hyperball are concentric.

for future classes to prevent the reduction of class angles, this strategy is found to be suboptimal in trainable classifiers. In fact, without supervision, the pre-allocated prototypes for future classes tend to collapse onto each other, as evidenced by [4, 5]. This tendency illustrates the inherent limitations of this approach in achieving optimal compatibility with trainable classifiers.

In contrast, stationary features of models learned through a pre-allocated d-Simplex fixed classifier are concentric and do not suffer from class collapse due to pre-allocation. Using this result and the three previously established assumptions, the verification of optimality can be achieved. This is done by computing the expected distance according to Eq. 8, particularly within the hyperballs of two models corresponding to a single class. Expected distance is computed according to Eq. 8 by shifting one of the hyperballs and assuming a uniform distribution. Given the symmetry of a hyperball, shifting in any single direction is adequate for the evaluation. Since no closed form solution of Eq. 8 exists Monte Carlo



(a) Same class

(b) Different classes

Figure 3. Comparison of expected distances between feature points from two learning phases, characterized by indices k (before learning) and t (after learning), across different dimensions of the representation space. Both $\mathbb{E}[D_{k,t}]$ and $\mathbb{E}[D_{k,k}]$ are examined. *(a)*: Expected distance in the case of same class, the value of $\mathbb{E}[D_{k,t}]$ remains less than $\mathbb{E}[D_{k,k}]$, satisfying on average the condition of Eq. 1a. *(b)*: In the case of two different classes, the expected distance, does not satisfy the condition of Eq. 1b.

integration is employed. Fig. 2 illustrates optimality for a corresponding class in two stationary models. It shows that as the amount of shift increases, there is a corresponding increase in the expected distance, a phenomenon observed across various dimensional spaces.

The same evaluation is used to verify the definition of compatibility in Eq. 1a and Eq. 1b:

$$\mathbb{E}[D_{k,t}] \leq \mathbb{E}[D_{k,k}] \tag{9}$$

(in the case of the same class) and if

$$\mathbb{E}[D_{k,t}] \geq \mathbb{E}[D_{k,k}] \tag{10}$$

(in the case of different classes) hold. In Fig. 3a and Fig. 3b, we show plots of $\mathbb{E}[D_{k,t}]$ and $\mathbb{E}[D_{k,k}]$ with varying feature dimension from 2 to 500. Without loss of generality, the hyperball radius starts at 1 and is reduced to 0.5 (further radius reductions follow the same principle and are not shown). The plots show that as the radius is reduced (i.e., more knowledge is assimilated) in the case of the same class the expected distance $\mathbb{E}[D_{k,t}]$ is always below $\mathbb{E}[D_{k,k}]$ at any feature dimensions (Fig. 3a). Differently, as shown in Fig. 3b, the expected distance evaluation for the case of different classes results in $\mathbb{E}[D_{k,t}] < \mathbb{E}[D_{k,k}]$ therefore not satisfying Eq. 1b. To satisfy Eq. 1b, the hyperball $\mathcal{B}_t(\mathbf{w}_{y_i})$ from Eq. 4 should be placed away from the hyperball $\mathcal{B}_k(\mathbf{w}_{y_j})$ of the other class (Eq. 5). Such repositioning changes the concentric arrangement of the hyperballs of the same class, which negatively affects the optimality.

The optimal approximation to compatibility directly follows from: (1) the fact that hyperballs centered at the vertices of a regular d-Simplex, are at their pairwise maximum distance, and (2) the addition of more classes does not alter this distance because their corresponding representation space is pre-allocated and remains unchanged (i.e., stationary). □

In the proof above, it emerges that the satisfaction of both compatibility constraints of Def. 1 cannot be achieved. In the following corollary, we provide the explicit statement outside the proof above for a clearer and more focused exposition of this result, as it has a general validity beyond the specific assumption of a d-Simplex fixed classifier.

Corollary 1 (Infeasibility) *The two compatibility inequalities in Def. 1 cannot be satisfied by the representation learned by a trainable classifier.*

Proof. The proof follows immediately from the arguments presented in the final part of the proof of Theorem 1. The discussion therein establishes that in order to satisfy Eq. 1b, a shift of the hyperball $\mathcal{B}_t$ in Eq. 4 away from the hyperball $\mathcal{B}_k$ in Eq. 5 is required. This results in a departure from the concentric configuration for the case of the same class, thereby negatively affecting the optimality of Eq. 1a. In the case in which the classifier can be trained, the introduction of additional classes alters the pairwise class distances, and as a result, a departure from the concentric configuration cannot be avoided. As a consequence the inequality constraints of compatibility cannot be satisfied. □

B. Implementation Details

In the following section, we provide more detailed information about the experimental settings described in Sec. 4.2. We pre-train ResNet18 models on ImageNet32 for 300 epochs. Pre-training was done using an SGD optimizer with a learning rate of 0.1, momentum 0.9, and weight decay $1 \cdot 10^{-4}$. Models are trained with a mini-batch size of 128, and the learning rate follows a cosine annealing schedule throughout the training process. For methods based on the d-Simplex fixed classifier [6], we pre-allocate $K = 1024$ classes (features vectors are then of size $d = 1023$) and training is

performed according to the cross-entropy loss of Eq. 2. The other methods utilize a trainable classifier, wherein the feature size corresponds to that of the ResNet18 architecture, namely 512.

Models were fine-tuned on CIFAR100R for 70 epochs. Fine-tuning was performed using the SGD optimizer with learning rate of 0.001, momentum 0.9, weight decay 10^{-4} and with mini-batch size of 128. The learning rate is decreased according to a linear scheduling with a reduction factor of 0.1 at epochs 50 and 65.

C. Ablation Studies

In this section, we present ablation studies of d-Simplex-HOC using CIFAR100R/10. These studies involved fine-tuning the model for 31 tasks, with two model replacements as is the experiment of Fig. 4a.

C.1. Hyperparamters

The training of d-Simplex-HOC is influenced by the hyper-parameters λ and τ, as used in Eq. 3 and Eq. 5, respectively. Tab. 1 shows the AC metric for different values of λ and τ. The results show that using $\lambda = 0.1$ and $\tau = 10$ yields the highest performance in terms of AC. A lower value of λ suggests a greater emphasis on the contrastive loss relative to the cross-entropy loss, prioritizing the higher-order component over the first-order one offered solely by the cross-entropy. The value of τ yielding the highest AC in our study closely aligns with that reported in [7]. This similarity suggests a consistent τ effect across various contexts of representation learning.

λ \ τ	1	5	8	10 (♠)	15	20
0.05	0.10	0.55	0.63	0.64	0.35	0.23
0.1 (♠)	0.10	0.58	0.64	**0.65**	0.36	0.23
0.25	0.06	0.30	0.43	0.42	0.34	0.21
0.5	0.09	0.23	0.19	0.20	0.18	0.21
0.75	0.17	0.19	0.16	0.13	0.12	0.10

Table 1. Ablation study for d-Simplex-HOC in 31 tasks using CIFAR100R/10 with two model replacements of λ (Eq. 3) and τ (Eq. 5). The evaluation is performed with respect to the AC metric. Values used in our implementation are marked with the "(♠)" symbol.

C.2. Learning Rate

Learning a new task without affecting the existing model's representation requires a proper selection of the learning rate. Tab. 2a reports the metrics AC and AA_{31}, obtained for different learning rate values η. A higher η enables the model to adapt more quickly to new tasks; however, this results in a noticeable decline in performance with respect to both AA_{31} and AC. This decline is primarily due to significant changes in the model's representation before and after the updates. In contrast, a lower learning rate allows the model to transition more gradually from its current state, leading to improved compatibility. This approach, while improving compatibility, results in a slight reduction in the model's ability to assimilate new knowledge from the task. Considering this trade-off, we opted for a learning rate of 0.001 in our implementation.

η	AC	AA_{31}
0.1	0.07	58.21
0.01	0.40	68.67
0.005	0.57	**68.94**
0.001 (♠)	**0.65**	67.40
0.0005	0.57	66.31
0.0001	0.32	63.44
0.00001	0.30	63.32

(a)

#imgs	AC	AA_{31}
500	**0.69**	**67.97**
300 (♠)	0.65	67.40
200	0.55	66.95
100	0.42	65.83
50	0.32	65.01
10	0.25	62.05
5	0.22	61.77

(b)

Table 2. Ablation for d-Simplex-HOC in 31 tasks using CIFAR100R/10 with two model replacements of learning rate η (a) and of the number of images (#imgs) per class in CIFAR100R (b). Values used in our implementation are marked with the "(♠)" symbol.

C.3. Training-sets Relative Size

We aim to study the impact on performance of the relative size between the dataset used for pre-training the models, namely ImageNet32, and the CIFAR100R dataset used for fine-tuning them. To this end, we varied the number of images per class in the CIFAR100R dataset. Tab. 2b shows the values with 500 (all the images of CIFAR100 are used in CIFAR100R), 300, 200, 100, 50, 10, and 5 images per class. We observe that compatibility performance (AC) decreases as the number of images per class reduces. Conversely, the average accuracy exhibits a gradual decline. This highlights that achieving compatibility is a complex constraint requiring adequate data.

C.4. Episodic Memory Size

Fine-tuning is performed using data from the new task along with an episodic memory to mitigate potential forgetting [8]. Consequently, we assess how the number of images per class in the episodic memory impacts the model's performance. Fig. 4 shows AA_t curves for various numbers of images per class in the episodic memory. These plots illustrate scenarios ranging from the *rehearsal-free* case, where no images are retained, to the case where all images of each class are stored (300 images per class), and include intermediate scenarios as well. As expected, the more data are used in the memory, the more the accuracy increases. Remarkably, in the rehearsal-free case, there is a continuous improvement in accuracy. This case indicates that d-Simplex-HOC is capable

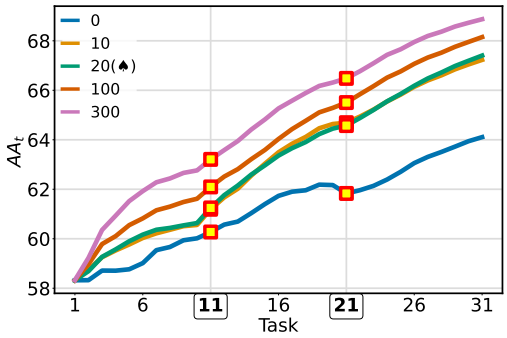


Figure 4. Ablation for d-Simplex-HOC in 31 tasks with CIFAR100R/10 with two model replacements of the number of images in the episodic memory (0 is *rehearsal-free*). Values used in our implementation are marked with the "(♠)" symbol.

of leveraging improvements from model replacement, even in the absence of episodic memory. This evidence may be relevant for future search/retrieval systems which evolve or enhance their performance over time.

D. d-Simplex fixed classifier PyTorch Code

We provide a GPU-based implementation to generate a d-Simplex classifier matrix $\mathbf{W}$ for a given number of pre-allocated classes K that offers faster computation compared to CPU-based implementations [6, 9, 10].

```
def dsimplex_fixed_classifier(K):
    W = torch.zeros((K, K-1))
    W[:-1,:] = torch.eye(K-1)
    W = W.cuda()
    c = torch.sqrt(1 + torch.Tensor([K-1]).cuda())
    W[-1,:] = W[-1,:] + (1 - c) / (K-1)
    W.add_(-torch.mean(W, dim=0))
    W.div_(torch.linalg.norm(W) + 1e-8)
    W.requires_grad = False
    return W
```

References

[1] Johann S Brauchart, Alexander B Reznikov, Edward B Saff, Ian H Sloan, Yu Guang Wang, and Robert S Womersley. Random point sets on the sphere—hole radii, covering, and separation. *Experimental Mathematics*, 27(1):62–81, 2018. 1

[2] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4690–4699, 2019. 1, 2

[3] Dustin G Mixon, Hans Parshall, and Jianzong Pi. Neural collapse with unconstrained features. *Sampling Theory, Signal Processing, and Data Analysis*, 20(2):1–13, 2022. 2

[4] 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*, 118(43):e2103091118, 2021. 2

[5] Yibo Yang, Shixiang Chen, Xiangtai Li, Liang Xie, 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? *Advances in Neural Information Processing Systems*, 35:37991–38002, 2022. 2

[6] Federico Pernici, Matteo Bruni, Claudio Baecchi, and Alberto Del Bimbo. Regular polytope networks. *IEEE Transactions on Neural Networks and Learning Systems*, 2021. 3, 5

[7] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020. 4

[8] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2001–2010, 2017. 4

[9] Federico Pernici, Matteo Bruni, Claudio Baecchi, and Alberto Del Bimbo. Maximally compact and separated features with regular polytope networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2019. 5

[10] Tejaswi Kasarla, Gertjan J Burghouts, Max van Spengler, Elise van der Pol, Rita Cucchiara, and Pascal Mettes. Maximum separation as inductive bias in one matrix. In *NeurIPS*, 2022. 5