Hypernym Bias: Unraveling Deep Classifier Training Dynamics through the Lens of Class Hierarchy

Roman Malashin **Valeria Yachnaya** **Alexander Mullin**
Saint-Petersburg State University of Aerospace Instrumentation
Pavlov Institute of Physiology, RAS

Abstract

We investigate the training dynamics of deep classifiers by examining how hierarchical relationships between classes evolve during training. Through extensive experiments, we argue that the learning process in classification problems can be understood through the lens of label clustering. Specifically, we observe that networks tend to distinguish higher-level (hypernym) categories in the early stages of training, and learn more specific (hyponym) categories later. We introduce a novel framework to track the evolution of the feature manifold during training, revealing how the hierarchy of class relations emerges and refines across the network layers. Our analysis demonstrates that the learned representations closely align with the semantic structure of the dataset, providing a quantitative description of the clustering process. Notably, we show that in the hypernym label space, certain properties of neural collapse appear earlier than in the hyponym label space, helping to bridge the gap between the initial and terminal phases of learning. We believe our findings offer new insights into the mechanisms driving hierarchical learning in deep networks, paving the way for future advancements in understanding deep learning dynamics.

1 INTRODUCTION

Deep neural networks have demonstrated remarkable capabilities in learning complex functions, yet the underlying reasons for their success remain an open question. In this work, we aim to shed light on one crucial aspect of this puzzle: **What patterns do neural networks tend to learn first, and which ones do they learn last?** The question has been studied for a long time, but valuable interpretations continue to emerge. One important contribution is the concept of simplicity bias proposed by Arpit et al. (2017), which suggests that neural networks tend to learn simple (frequent) patterns followed by more complex or noisy patterns. Neural collapse is another phenomenon that characterizes the terminal phase of classifier training (Papyan et al., 2020). During this phase variance of penultimate features within the same class converges to zero, while the class means are arranged in the feature space according to Equiangular Tight Frame (ETF) structure.

In this work we study dynamics of the classifier learning process through the lens of evolving hierarchy of object categories. According to our experiments we argue that in the classification task the network tend to learn relations between high-level categories (hypernyms) on early stages, while more specific (hyponyms) are learned in the later training epochs; terminal phase is characterized by removing hypernymy relations from the features of the layer. By analogue we call this tendency a hypernym bias, as it can be nicely interpreted as manifestation of simplicity bias: classes of same hypernym share frequent (simple) features, therefore neurons responsible for their detection are activated more often and trained faster. The neural collapse state can be interpreted as final state of the label clustering process in the specific layer: every cluster contains only a single label, and information about labels' relations is removed from features. The basic intuition about the phenomenon we study in this paper can be grasped from Figure 1.[1]

From Figure 1b training can be interpreted as top-to-bottom hierarchical label clustering that starts by in-

Proceedings of the 28th International Conference on Artificial Intelligence and Statistics (AISTATS) 2025, Mai Khao, Thailand. PMLR: Volume 258.

[1]Video of UMAP embeddings evolution is available here.

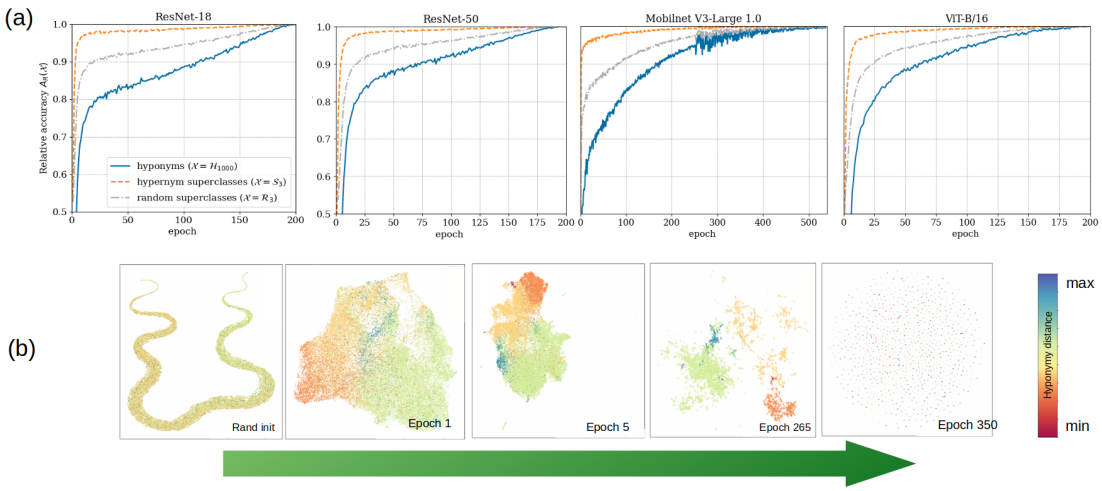

Figure 1: Hypernym bias. (a) Relative accuracy during training for ResNet-18, ResNet-50, ViT-B/16, and MobileNet-V3. Hypernyms reach near maximum recognition accuracy on early epochs. (b) UMAP embeddings of ResNet-152 penultimate layer features in neural collapse settings. Color shows hyponymy distance relative to anchor class (<tabby cat>). The dynamics can be interpreted as top-to-bottom hierarchical label clustering.

creasing distance between large hypernyms[2] and neural collapse at the end of training, which indicates the formation of singleton clusters containing only a single label, and thus removing hyponymy relations from the feature space.

In this work:

- We provide empirical evidence that on early epochs of training classifiers tend to prioritize learning decision boundaries that distinguish superclasses of labels that unite large hypernyms, and thus learning can be seen from the perspective of hierarchical clustering of labels.
- We assess the evolution of the class hierarchical structure of the manifold. By comparing within-manifold and hyponymy distances, we show that learned clusters of labels align with the semantic structure of the dataset during the early stages of training.
- We show that in the hypernym label spaces some aspects of neural collapse manifest earlier than in the original (hyponym) label space.

2 RELATED WORKS

Hierarchical learning in neural networks. Hierarchical learning phenomenon - where network first acquire broader categories - has been explored by Rogers and McClelland (2004), who used small backpropagation network to model how children learn. Saxe et al. (2019) provided mathematical explanation why this dynamic emerges in linear networks, and Pinson et al. (2023) extended these findings to convolutional linear networks. Additionally, Decelle et al. (2023) used RBM to uncover hierarchical relationships, demonstrating that the phenomenon generalizes beyond backpropagation-based networks. *Our work extends these insights to a practical setting, bridging the gap between theory and real-world training scenarios.*

Hierarchical classification. Errors in trained classifiers tend to occur more frequently between closely related classes, such as different species of animals or various types of vehicles. This pattern is evident in the block-diagonal structure of the confusion matrix, provided that the classes are properly sorted. Initially, the ImageNet dataset was organized according to the lexical database of semantic relations WordNet (Fellbaum, 2010) under assumption that it will help to build new efficient classification algorithms (Russakovsky et al., 2015). Since then lots of works tried to incorporate hierarchy bias into the training process of deep neural networks: either in architecture (Hinton et al., 2015; Malashin, 2016; Xiao et al., 2014) or in loss function (Redmon and Farhadi, 2017; Brust and Denzler, 2019). Some authors reported promising results. However these findings are often derived in nonoptimal training settings (Brust and Denzler, 2019) or from closed datasets, which complicates fair comparison (e.g. (Hinton et al., 2015)). It is also well-

[2]For example, on the 1st epoch features associated with close to anchor labels (red, orange and yellow dots) are close but not separated from each other

known that, in few-shot leaning settings removing hierarchically related classes from the pretraining process significantly impacts the performance on these classes (Vinyals et al., 2016). This suggests that features of the network inherently capture hierarchical relationships. *We believe our result provide insight into why hierarchical structure is not used in state-of-the-art solutions.*

Frequency Principle. The Frequency Principle (FP) or Spectral bias is a phenomenon in neural network training, independently discovered in (Rahaman et al., 2019) and (Xu et al., 2019), which states that the low-frequency components of the target function are learned first by the neural network. The principle is formulated for the multidimensional spectrum of data (where the number of dimensions equals the number of elements in the input feature vector). However, its interpretation is often associated with one-dimensional (time sequences) or two-dimensional (images) spaces (Xu et al., 2024).

Theoretical justification of FP remains challenging (Xu et al., 2024) and its applicability may be limited in some cases (Zhang et al., 2020; Wang et al., 2023a). Nevertheless, FP serves as a valuable tool to think about training dynamics, providing intuitive interpretations for existing results. For example, deep image prior (DIP) (Ulyanov et al., 2018) leverages randomly initialized network to restore image. It can be explained more generally by FP: by inherently learning low-frequency components before high-frequency details, neural networks used in DIP naturally reconstruct the essential structures of images first. Beyazit et al. (2024) use FP to explain the poorer performance of neural networks on tabular data compared to other methods (such as decision trees) (Shwartz-Ziv and Armon, 2022). Benjdiraa et al. (2023) suggest to use high-pass filter before calculating the similarity metric between the reconstructed and reference images to improve reconstruction.

Interpreting hypernym bias as a manifestation of Frequency Principle is complicated by two factors: a) the need of Fourier Transform computation of high-dimensional data, which suffers from the curse of dimensionality and b) challenges in applying the principle to naturally unordered datasets (which is true for image classification).

Simplicity bias. Arpit et al. (2017) have shown that simple patterns are learned first before the network transitions into the noise memorization phase. Simple patterns are associated with those that frequently occur across many examples. Arpit et al. (2017) compare learning dynamics on real data with dynamics on noise (where no patterns are present). They suggest that datasets contain a large number of simple examples with common patterns, which are learned during the first epoch, after which the network spends most of its time learning more complex examples. Hu et al. (2020) theoretically and practically demonstrate that during the initial phase of training, a neural network learns a linear function. In (Mangalam and Prabhu, 2019), a more general observation is experimentally demonstrated: during the early iterations, networks learn to correctly recognize examples that can be distinguished by shallow classifiers (such as SVMs or Random Forests). Zhang et al. (2021) emphasize that this property of neural networks helps prevent overfitting, even in conditions of significant overparameterization. Therefore, early stopping can be considered a method to prevent the learning of high-frequency noise components in individual signal examples. Simplicity bias sometimes is considered to be a pitfall (Shah et al., 2020; Pezeshki, 2022), because it allows learning spurious correlations; these features are often not useful for generalization and are referred as shortcuts (Geirhos et al., 2020; Wang et al., 2022). A separate research direction focuses on finding ways to reduce false correlations and studying learning dynamics in their presence (Pezeshki, 2022; Qiu et al., 2024).

The Frequency Principle and Simplicity Bias are closely related; the latter can be thought of as more general but less quantifiable of the former. Xu and Zhou (2021) demonstrate that, in regression tasks, low frequencies carry the majority of the information needed for signal reconstruction and are easier to learn.

In simplicity bias terminology frequent features are called simple, and simplicity bias should manifest in accelerated learning of hypernym features common among many classes. This is what we robustly observe in neural networks trained in practical settings.

Neural collapse. Neural collapse (NC) phenomenon was first described by Papyan et al. (2020) and then was further extended to broader settings in (Dang et al., 2024), (Fang et al., 2021) and other works. Neural collapse is a state of the last layer of the neural classifier, which can be observed in the terminal phase of training. It is characterized by zero within-class variability of features, when class means form vertices of the Equiangular Tight Frame (ETF), and classification decision is similar to prototypes (Snell et al., 2017). While the ETF configuration is optimal from the perspective of robustness to adversarial attacks (Papyan et al., 2020), its contribution to generalization is arguable, as NC is observed only when accuracy on a balanced train set reaches 100% (Hui et al., 2022). Nevertheless knowledge about dynamics of the last layer is helpful, for example, for designing few-shot learning algorithms, where often only last layer is

tuned (Yang et al., 2023).

Neural collapse of penultimate layer can be seen as deeper representation of perfectly diagonal confusion matrix, which doesn't posses any information about hyponymy relations. From the perspective of hypernym bias NC is the final stage of top-to-bottom hierarchical label clustering process.

3 METHODOLOGY

We base our conclusions from experiments with ImageNet dataset, which classes are organized according to lexical database of semantic relations WordNet (Fellbaum, 2010). We validate these conclusions on additional datasets in Section C.1.4. WordNet defines a graph where the nodes are synsets (sets of synonyms), and the edges represent the hypernymy-hyponymy relationships. Examples of synsets are: {car, automobile} and {small, compact}. Synset {dog} is a hypernym relative to the synsets {bulldog} and {spitz}, which are its hyponyms. It is important to note that the hyponymy relationship is not the only one; for instance, meronymy (part-whole, e.g., {wheel}-{car}), troponymy (one expressing an aspect of the other, e.g., {whisper}-{speak}), and others relationships are often considered. While it's arguably impossible to determine the relationships fully and consistently, if we view hypernym bias through the lens of simplicity bias (Arpit et al., 2017), the imperfect nature of these relationships should not negate the phenomenon. This is because hypernyms tend to share features across multiple hyponyms.

For first series of experiments we used simplified to a tree WordNet graph (Russakovsky et al., 2015). This allows us to interpret the hyponym classifier as unambiguous greedy classifier of hypernyms. For manifold analysis, we utilize the complete WordNet graph, which permits multiple parent relationships. To avoid confusion we refer to the simplified version as WordNet tree, while the full version is referred to as WordNet graph.

3.1 Greedy hypernym classification

Label spaces. Let $\mathcal{H}_{N_H} = \{h_i\}_{i=1}^{N_H}$ be the set of N_H hyponyms (classes), and $\mathcal{S}_{N_S} = \{s_j\}_{j=1}^{N_S}$ be the set of N_S hypernyms (superclasses), where each superclass $s_i = \{h_k\}_{k=1}^{N_{s_i}}$ is defined such that:

$$\bigcup_{s\in\mathcal{S}} s = \mathcal{H}_{N_H} \quad \text{and} \quad s_i \bigcap s_j = \emptyset \quad \forall s_i, s_j \in \mathcal{S}, \quad i \neq j. \tag{1}$$

In the case of ImageNet, we have $N_H = 1000$. For simplicity we omit subscript notations indicating the sizes of sets where possible. A standard classifier network $f(x;\theta)$ equipped with softmax, maps image x to a probability distribution over hyponyms (classes) $h \in \mathcal{H}$ based on the current weights θ. The training objective utilizes the cross-entropy loss function:

$$\mathcal{L}_{CE} = -\sum_{h\in\mathcal{H}} \log f_h(x;\theta) y_h(x), \tag{2}$$

where $y_h(x)$ represents true probability that x belongs to h (0 or 1 in one-hot encoding), and $f_h(x;\theta)$ is the estimated probability of the same.

We consider $f(x;\theta)$ as a function that also maps an image to different label spaces greedily (not considering confidence level). We denote labels and label predictions with $\hat{}$ symbol. The predicted hyponym is then given by:

$$\hat{f}_{\mathcal{H}}(x;\theta) = \text{argmax}_{h\in\mathcal{H}} f_h(x;\theta), \tag{3}$$

and the true label $\hat{y}_{\mathcal{H}}(x) = \text{argmax}_{h\in\mathcal{H}} y_h(x)$.

We explore manifestation of hypernym bias through the convergence dynamics by introducing a trivial mapping $\mathcal{T}_{\mathcal{H}\to\mathcal{S}} : \mathcal{H} \to \mathcal{S}$, which maps each hyponym to its parent hypernym at a specified level in the WordNet tree. The predicted hypernym then:

$$\hat{f}_{\mathcal{S}}(x;\theta) = \mathcal{T}_{\mathcal{H}\to\mathcal{S}}(\hat{f}_{\mathcal{H}}(x;\theta)), \tag{4}$$

$$\hat{y}_{\mathcal{S}}(x) = \mathcal{T}_{\mathcal{H}\to\mathcal{S}}(\hat{y}_{\mathcal{H}}(x)). \tag{5}$$

In our experiments $\mathcal{T}_{\mathcal{H}\to\mathcal{S}}$ is implemented as simple reverse traversal of the WordNet tree starting from $\hat{f}_{\mathcal{H}}$; it returns the first match with any element of $\mathcal{S}$.

To evaluate the specific impact of hypernym relationships in the WordNet tree, we introduce $\mathcal{R}$, a randomly generated label space that is isomorphic to $\mathcal{S}$. Isomorphism here means that there is a one-to-one correspondence between labels in $\mathcal{R}$ and $\mathcal{S}$, and the sizes of the corresponding superclasses are identical. Formally there exists bijection $g : \mathcal{R} \to \mathcal{S}$, such that $\forall r \in \mathcal{R} \quad \exists s \in \mathcal{S} : |r| = |g(r)|$. This ensures that any differences in network performance on $\mathcal{R}$ and $\mathcal{S}$ can be attributed to the semantic structure encoded in $\mathcal{S}$. For example, while $\mathcal{S}$ might group classes into "Animals", "Artifacts", and "Others", $\mathcal{R}$ would have three groups of the same sizes but with randomly assigned classes.

Given that $\mathcal{T}_{\mathcal{H}\to\mathcal{R}}$ and $\mathcal{T}_{\mathcal{H}\to\mathcal{S}}$ are predefined and not learned, we can assume that the same network $f(x;\theta)$ maps each image to three label spaces simultaneously without interfering training process:

$$\hat{f}_{\mathcal{H}}(x;\theta) \in \mathcal{H}, \hat{f}_{\mathcal{S}}(x;\theta) \in \mathcal{S}, \hat{f}_{\mathcal{R}}(x;\theta) \in \mathcal{R}. \tag{6}$$

Corresponding ground truths labels are $\hat{y}_{\mathcal{H}}(x), \hat{y}_{\mathcal{S}}(x), \hat{y}_{\mathcal{R}}(x)$. Thus we can study accuracy metrics in each label space.

Metrics. Let the test dataset $D = \{x_i\}_{i=1}^{N_D}$ contain N_D images for which label function $\hat{y}$ is defined. The accuracy of classification in label space $\mathcal{X}$ after t epochs is estimated as:

$$A(\mathcal{X}, t) = \frac{1}{N_D} \sum_{i=1}^{N_D} \mathbb{I}(\hat{f}_{\mathcal{X}}(x_i; \theta_t) = \hat{y}_{\mathcal{X}}(x_i)). \quad (7)$$

We normalize $A(\mathcal{X}, t) \in [0, 100]$.

Given the accuracy of recognizing hyponyms $A(\mathcal{H}, t)$, accuracy of recognizing random superclasses $A(\mathcal{R}, t)$ can be estimated theoretically. We derive and validate exact formula in Appendix A.

Absolute accuracy can be misleading indicator of convergence since it varies a lot (as recognizing superclasses is inherently easier due to higher probability of guessing correctly). To address this, we introduce relative accuracy $A_R(\mathcal{X}, t)$ at epoch t for assessment. Let $T = \text{argmax}_t A(\mathcal{X}, t)$, then relative accuracy is defined as:

$$A_R(\mathcal{X}, t) = \frac{A(\mathcal{X}, t)}{A(\mathcal{X}, T)}, \quad (8)$$

which ensures that $A_R(\mathcal{X}, t) \in [0, 1]$.

Expression (8) does not account for different probabilities of random guessing that depends on superclass sizes distribution. To address this, we also introduce the relative gain in accuracy, $G_R(\mathcal{X}, t)$, defined as:

$$G_R(\mathcal{X}, t) = \frac{A(\mathcal{X}, t) - B(\mathcal{X})}{A(\mathcal{X}, T) - B(\mathcal{X})}, \quad (9)$$

where $B(\mathcal{X})$ is the accuracy of a random guess, taking into account sets imbalance. This baseline accuracy can be estimated as $B(\mathcal{X}) = \mathbb{E}A(\mathcal{R}, 0)$, i.e. expected accuracy before training. In the next section we show that estimates of accuracy gain for random superclasses $G_R(\mathcal{R})$ and for hyponyms $G_R(\mathcal{H})$ are identical, allowing us to measure impact of hypernymy relation fairly by comparing $A_R(\mathcal{R})$ against $A_R(\mathcal{S})$.

Though observation of accelerated formation of hypernym features from the perspective of different accuracy metrics are very practical and intuitive manifestation of hypernym bias, these metrics are not integral (need arbitrary choices of hypernym level), and importantly they doesn't generalize to describe features of deeper layers.

3.2 Assessing evolution of class hierarchical structure of the manifold

Assuming that features lie near low-dimensional manifold, we evaluate how well this feature manifold aligns with WordNet graph on each epoch. Our approach has three major steps: a) define the distances between classes in feature space, b) define distances between classes in WordNet graph and c) compare two distance matrices via cophenetic correlation coefficient.

Graph in feature space. To estimate distances between feature sets within manifold we adapt approach from (Jin et al., 2020). First we sample $2 \times K$ training examples per each of C classes, and divide them into non-overlapping query and support sets Q and S:

$$Q = \bigcup_{c=1}^{C} U^c,\ U^c = \{u_k^c\},\ S = \bigcup_{c=1}^{C} V^c,\ V^c = \{v_k^c\}. \quad (10)$$

where $U^c \cap V^c = \emptyset,\ \forall c \in [0, C]$. Assuming suitable metric, we define distance $d_f(u^{c_i}, V^{c_j})$ between query feature of class c_i and support set V^{c_j} of another class c_j as minimum individual distance:

$$d_f(u^{c_i}, V^{c_j}) = \min_{v \in V^{c_j}} d_f(u^{c_i}, v). \quad (11)$$

Next we measure probability that distance between query point of class c_i and support set V^{c_j} is less than predefined radius r:

$$P_r(c_i, c_j) = P\left(d_f(u^{c_i}, V^{c_j}) < r\right) = \quad (12)$$

$$= \mathbb{E}_{u \sim U^{c_i}} \left[\mathbb{I}\{d_f(u, V^{c_j}) < r\}\right], \quad (13)$$

where $\mathbb{I}\{d_f(u, V^{c_j}) < r\}$ is the indicator function.

Finally we estimate similarity of two classes as:

$$\rho(c_i, c_j) = \frac{1}{r_{\max}} \int_0^{r_{\max}} P_r(c_i, c_j)\, dr, \quad (14)$$

where $r_{\max}$ is maximum radius. The value $\rho(c_i, c_j) \in [0, 1]$ represents mutual cover when $i \neq j$ and self-cover when $i = j$, as described by Jin et al. (2020). Similarity $\rho(c_i, c_j) = 1$ indicates that every query feature u^{c_i} is within zero distance to *at least one* feature v^{c_j} in support set of the class j, low values indicate that there are many query features that are far from *any* features in V^{c_j}. We construct similarity matrix $A = [a_{ij}]$, where $a_{ij} = \rho(c_i, c_j)$ and treat its elements as probability that edge between i-th and j-th node of the graph exists. Transformation from similarity to a distance matrix D_F is straightforward:

$$D_F = [d_F(i, j)] = 1 - A. \quad (15)$$

Label distance in WordNet graph. Distance matrix between class labels according to WordNet structure can be defined as:

$$D_W = [d_W(i, j)], \quad (16)$$

where $d_w(i, j)$ is the shortest path between synset i and synset j in the graph. With this definition we can use full (not tree) WordNet graph.

Graph comparison. To compare graphs we utilize Cophenetic Correlation Coefficient:

$$\mathrm{CCC} = \frac{\sum_{i<j}(d_W(i,j) - \overline{D_W})(d_F(i,j) - \overline{D_F})}{\sqrt{\sum_{i<j}(d_W(i,j) - \overline{D_W})^2 \cdot \sum_{i<j}(d_F(i,j) - \overline{D_F})^2}}, \tag{17}$$

where $\overline{D_F}$ and $\overline{D_W}$ are the mean of all pairwise distances in the original distance matrices D_F and D_W. $CCC \in [-1, 1]$ measures linear correlation of two distance matrices.

Metric. Examples are not distributed uniformly in feature space in the course of training: fixed volume contains different number of sample points and distance should be adjusted accordingly. As can be seen from Figure 2a with the use of Euclidean metric in feature space there is no trend to increase mutual to self-cover ratio during training. This suggests the potential issues in the measurement of similarity, as the Euclidean metric likely captures the properties of the entire manifold rather than specific characteristics of individual class manifolds and their mutual relationships. In this work we address the issue by utilizing the

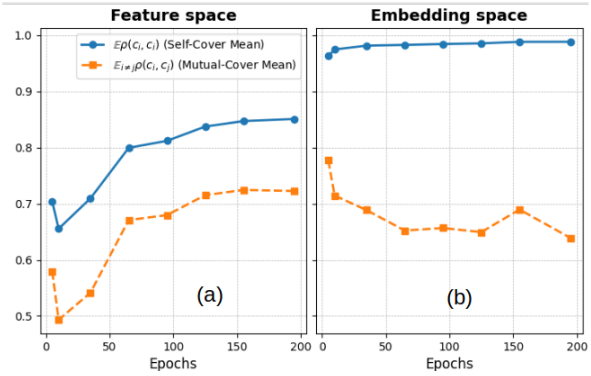

Figure 2: Mean of mutual- and self-covers of the features of penultimate ResNet-50 layer in the course of training: original feature space (a) and UMAP embeddings space (b)

formally grounded UMAP algorithm (McInnes et al., 2018), which approximates a manifold where the data is uniformly distributed and effectively normalizes distances according to local connectivity before embedding optimization. We transform the feature space into an M-dimensional embedding space, where the usage of Euclidean metric $d_f(u, v) = \|u - v\|_2$ is justified. The dynamics of self-to-mutual cover ratio in UMAP embedding space depicted in Figure 2b, exhibit expected behavior of an increasing cover difference.

3.3 Neural collapse in hypernym label space

Neural collapse (Papyan et al., 2020) is a phenomenon that is observed in penultimate layer of the classifier when recognition accuracy on the training set reaches 100%. Neural Collapse is characterized by four manifestations in the last-layers weights and last-layer activations: within-class variability collapse (NC1), convergence to simplex ETF (NC2), convergence to self-duality (NC3) and simplification to nearest-class center (NC4). We give brief overview of neural collapse in Appendix A. Originally NC is defined on the label space $\mathcal{H}$, where training loss is estimated. Assuming hypernym bias we expect at least some properties of neural collapse in label spaces of hypernyms emerge earlier. We replicate calculations as in (Papyan et al., 2020) but in different label spaces after applying $\mathcal{T}_{\mathcal{H}\to\mathcal{R}}$ or $\mathcal{T}_{\mathcal{H}\to\mathcal{S}}$, described in Section 3.1. To compute NC2-NC4 we must consider that the weight matrix $\mathbf{W} = [\mathbf{w}_c]$ contains C weights rows, consistent with the number of labels. We interpret each row $\mathbf{w}_c$ as prototype (Snell et al., 2017) of its own hyponym. Assuming linear relationships to obtain prototypes of hypernyms $s \in \mathcal{S}$ we can average columns under hypernym relation. Thus rows of the resulting weight matrix can be calculated as $\mathbf{w}_s \triangleq \mathrm{Ave}_c\{\mathbf{w}_c\}$, $c \in s$, where Ave is averaging operator.

4 EMPIRICAL RESULTS

4.1 Experimental setup

Top level hypernyms. Top level of the WordNet tree contains 9 synsets. we expect manifestation of hypernym bias should be more pronounced in large hyperyms. Since top-level synsets are unevenly saturated with classes, we grouped them into the sets of synsets and used as top level hypernym superclasses; for clarity, most of the results are reported with respect to these label spaces: 1) hyponyms $\mathcal{H}$: 1000 classes used for training, 2) hypernym superclasses $\mathcal{S}_3$: three superclasses formed from 522 (artifacts), 398 (animals), and 80 (others) classes according to the WordNet tree top level synsets 3) random superclasses $\mathcal{R}_3$: three superclasses formed from 80, 522, and 398 randomly selected classes.

Lower level hypernyms. We demonstrate the robustness of the observed phenomenon to different specifications of $\mathcal{S}$ with the use of different levels of hypernymy tree.

Architectures and hyper parameters. In experiments we used the ResNet-50, ResNet-18 (He et al., 2016), ViT-B/16 (Alexey Dosovitskiy, 2020) and MobileNet-V3 (Howard et al., 2019) architectures. All models were trained with the use of augmentation,

starting from randomly initialized weights. Except for Vit-B/16, the training parameters gave results comparable with the state of the art results for these architectures. For neural collapse analysis, we carefully reproduced all the parameters used by Papyan et al. (2020) and trained ResNet-152 for 350 epochs. We did not perform a learning rate search and fixed it to be 0.1. For the greedy hypernym classifier we report metrics on validation set, while neural collapse is evaluated on the train set. To compute CCC we embedded features into a 10-dimensional feature space.

Details on superclass formation with the use of WordNet tree and training parameters can be found in Appendix Appendix B.

4.2 Results

Greedy hypernym classifier. Table 1 shows final absolute accuracy, and Figure 1a displays relative accuracy curves in hyponym, hypernym and random superclass label spaces. Across all architectures during initial epochs of training relative accuracy for hypernyms increases significantly faster than for the rest of considered label spaces. From the table, it is evident that the final recognition accuracy in hypernym label space $A(\mathcal{S}_3)$ is significantly higher than accuracy $A(\mathcal{R}_3)$ in random superclasses label space. However, the final hypernym accuracy strongly depends on the size of the network. For instance, ResNet-50 outperforms ResNet-18 by 61% in terms of error reduction for hypernyms, compared to a 43% improvement for hyponyms. This should be taken into account when using interpretation through the lens of simplicity bias. Note, that hypernym accuracy never fully plateaus until the end of training. Figure 3 shows relative accuracy gain computed according to (9) for ResNet-50. The metric behaves identically for random superclasses and hyponyms showing its invariance to label imbalance. It clearly demonstrates accelerated dynamics of learning hypernyms, but more importantly it justifies drawing conclusions about hypernym vs hyponym training dynamics by comparing $A_R(\mathcal{S}_n)$ against $A_R(\mathcal{R}_n)$ (instead of $A_R(\mathcal{H}_{1000})$).

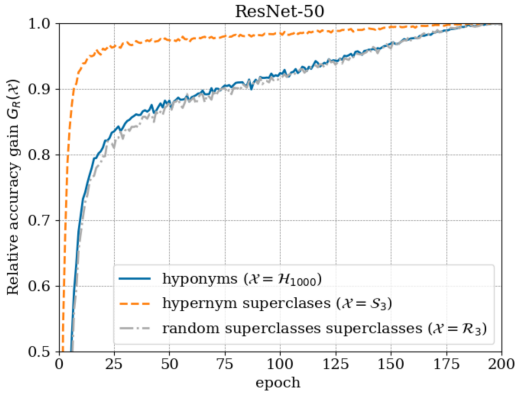

Figure 3: Relative accuracy gain during training of ResNet-50.

As an indicator of the training convergence with respect to number of hypernyms we considered the number of epochs required to reach 95% of the maximum accuracy. The results are depicted in Figure 4.

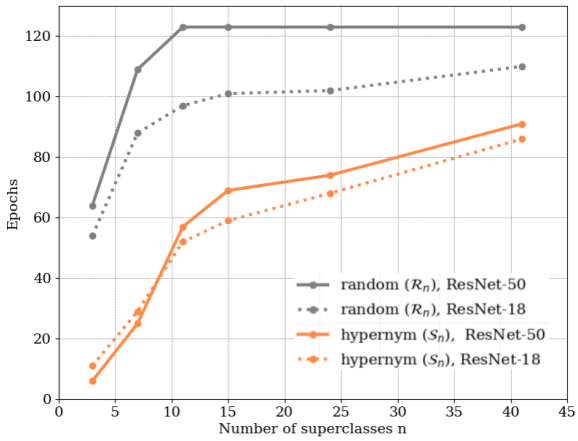

Figure 4: 95% accuracy convergence period (in epochs) for different number of hypernyms

As observed, the recognition accuracy of hypernyms converges significantly faster, although smaller hypernyms tend to take longer to reach convergence. For instance, ResNet-50 training reaches 95% accuracy in 6 epochs for $\mathcal{S}_3$, while for randomly formed superclasses $\mathcal{R}_3$, it takes 64 epochs. Interestingly, the effect of accelerated hypernym learning is more salient in larger ResNet-50 than in smaller ResNet-18. The convergence rate for hypernyms is approximately equal for both networks (orange lines in Figure 4), whereas the features required for hyponym classification consistently take longer to converge (gray lines on the same graph) in ResNet-50.

Although our choice of grouping synsets within one WordNet level was arbitrary, the absence of such grouping and the choice of a different convergence threshold do not affect the conclusions obtained in this section. More experiments, including experiments with animals hypernyms, different synset grouping strategies can be found in Appendix C.

Neural collapse. In Figure 5 we plot graphs demonstrating that some properties of neural collapse are observed in $\mathcal{S}_3$ hypernym label space on early iterations of training. UMAP embeddings of penultimate

Table 1: Top-1 recognition accuracy for different label spaces

Set	ResNet-50	ResNet-18	ViT-B/16	MobileNet-V3
hyponym, $A(\mathcal{H}_{1000})$	79	69.8	75.6	75.6
hypernym, $A(\mathcal{S}_3)$	97.7	96.3	97.1	97.2
random, $A(\mathcal{R}_3)$	87.6	83	86.3	86.3

Figure 5: Estimation of NC1, NC3 and NC4 properties of neural collapse in different label spaces

features are shown in Figure 1. We provide all the graphs as well as more UMAP visualizations in the Appendix C.

4.3 Manifold hierarchical structure, layer-wise and data frequency analysis

Alignment with WordNet graph and layer-wise dynamics. It is well-known that layers of convolutional neural networks converge faster than the entire neural network (Raghu et al., 2017; Wang et al., 2023b). Residual skip connections can possibly help transmit less distorted low-level feature information to the upper layer responsible for decision-making. Thus, layer-wise dynamics can possibly be related to hypernym bias: hypernyms can be recognized without use of high level features. We apply framework developed in Section 3.2 to evaluate how well feature manifold of different levels aligns with WordNet graph on each epoch. The results presented in Figure 6a deny suggested hypothesis.

As can be seen from the figure, WordNet graph and manifold distances' linear alignment rapidly grows in the first 5 epochs. Maximum correlation of 0.6 is achieved in the middle of the training by top layers and starts to decay after that. Bottom layers consistently show lower values. These conclusion is in coherence with the results of using linear probes (Alain and Bengio, 2016) in hypernym label spaces (Figure 6c), which show layers' redness for hypernym classification. We observe that probes from higher levels consistently and similarly provide better accuracy for both hyponym and hypernym spaces. In no experiment do we observe that hypernym classification based

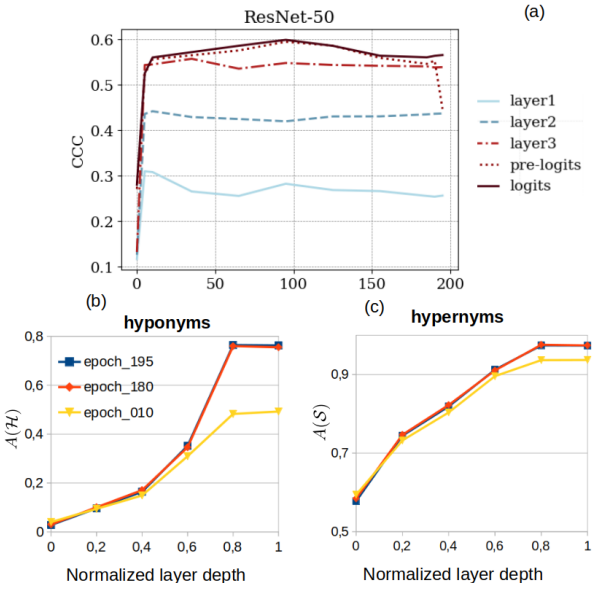

Figure 6: Cophenetic correlation coefficient of the WordNet graph and the mutual covers distance matrix (a), linear probing of several checkpoints of ResNet-50 in the hyponym (b) and hypernym (c) spaces

on intermediate layers can be performed with accuracy comparable to classification based on the higher layer. Thus, we cannot conclude that the first layers contain sufficient knowledge for recognizing hypernyms.

Frequency analysis. The Frequency Principle (Xu et al., 2019) is often linked to image frequencies, suggesting that hypernyms could be recognized using low frequencies, which are learned early in training, while hyponyms might require high frequencies learned later.

We tested this hypothesis and conclude that metrics in hypernyms label spaces do not exhibit a lesser dependency on high frequencies in the first phase of training.

These and some other experimental results, the description of the probing methods are given in Appendix C.

4.4 Generalization

We conducted additional experiments to validate that our findings generalize beyond ImageNet.

CIFAR-100. CIFAR-100 (Krizhevsky et al., 2009) comprises 60,000 32×32 images distributed across 100 classes, which are further grouped into 20 superclasses. Let $\mathcal{S}_{20}$ represent the hypernym label space (superclasses) and $\mathcal{R}_{20}$ denote a random label space isomorphic to $\mathcal{S}_{20}$. We trained ResNet-18 and achieved 77% validation accuracy in hyponym label space. We employed the greedy classifier model described in Section 3.1 and observed accelerated growth of $A_R(\mathcal{S}_{20})$ compared to $A_R(\mathcal{R}_{20})$, as expected.

DBPedia. DBPedia (Lehmann et al., 2015) is a text classification dataset containing 630,000 samples. It includes 14 classes, which we grouped into 5 hypernyms ($\mathcal{S}_5$) based on the first level of the DBPedia ontology. We fine-tuned a pretrained BERT-base model, and the test errors were as follows: 0.48% for random superclass labels, and 0.37% for hypernym labels. The relative accuracy curves demonstrate that the difference between $A(\mathcal{R}_5)$ and $A(\mathcal{S}_5)$ emerges during the early iterations, as predicted. Hypernym bias is evident.

Further details on the CIFAR-100 and DBPedia experiments, including relative accuracy curves, are provided in Appendix C.1.4.

5 DISCUSSION AND FUTURE DIRECTIONS

Hierarchical classification. To the best of our knowledge, the WordNet graph has not been widely adopted in state-of-the-art classification algorithms. Our experiments suggest that this may be because the hyponym-hypernym relationships are implicitly learned by the network itself. Our results show that when the loss function incorporates multiple cross-entropy losses from different hierarchy levels (as in (Redmon and Farhadi, 2017)), the contributions of higher-level hierarchy components are significant only during the early iterations of training, with a much lower impact in later stages.

For example, Brust and Denzler (2019) report similar accuracy when leveraging the hyponymy relation on CIFAR-100, but they observe the faster convergence and a shift in the initial training phase. Our findings explain this by showing that hypernym relations are not independently learned by the network only in the first epochs of training. Improvements reported in (Brust and Denzler, 2019) for ImageNet classifiers we attribute to their use of MSE loss, which is suboptimal for classification.

Future work. Future research could explore hypernym bias in other domains and datasets, such as AudioSet (Gemmeke et al., 2017) that has a hierarchical organization of categories well suited to the proposed framework.

A promising area of research involves moving away from reliance on predefined external class structures to learning class hierarchies directly through data-driven hierarchical clustering. For example, the class similarity matrix A from Equation (15) can be treated as an adjacency matrix, reframing the problem into identifying hierarchical communities in the graph. A data-driven approach has two key benefits: (a) hierarchical clustering can uncover more accurate and intuitive representations of class relationships, revealing latent patterns missed by external structures, and (b) analyzing the training dynamics of classifiers in conjunction with these learned hierarchies could provide valuable insights for designing more efficient and adaptive clustering algorithms.

Finally, exploring theoretical connections between hypernym bias and other known biases could deepen its theoretical understanding and bridge practical findings with theoretical models, which often lack strong empirical validation.

6 CONCLUSIONS

Our primary objective was to establish that hypernym bias is a consistent phenomenon across different neural architectures trained in practical settings, addressing a gap in prior research on biases. Our experimental framework provides a systematic and quantifiable approach to studying biases in practical scenarios, which can serve as a foundation for future research in this area. To our knowledge, we are the first to link early-stage training bias to neural collapse in the final stage by estimating it across label spaces, suggesting hypernym bias as a factor in neural collapse research. Our experiments, which did not reveal parallels with existing phenomena (layer-wise training dynamics and data frequency biases) reinforce the idea that hypernym bias is a distinct phenomenon requiring independent investigation.

7 Acknowledgments

The paper was prepared with the financial support of the Ministry of Science and Higher Education of the Russian Federation, grant agreement No. FSRF-2023-0003, "Fundamental principles of building of noise-immune systems for space and satellite communications, relative navigation, technical vision and aerospace monitoring"

References

Alain, G. and Bengio, Y. (2016). Understanding intermediate layers using linear classifier probes. *arXiv preprint arXiv:1610.01644*.

Alexey Dosovitskiy, Lucas Beyer, A. K. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Arpit, D., Jastrzębski, S., Ballas, N., Krueger, D., Bengio, E., Kanwal, M. S., Maharaj, T., Fischer, A., Courville, A., Bengio, Y., et al. (2017). A closer look at memorization in deep networks. In *International conference on machine learning*, pages 233–242. PMLR.

Benjdiraa, B., Alia, A. M., and Koubaa, A. (2023). Guided frequency loss for image restoration. *arXiv preprint arXiv:2309.15563*.

Beyazit, E., Kozaczuk, J., Li, B., Wallace, V., and Fadlallah, B. (2024). An inductive bias for tabular deep learning. *Advances in Neural Information Processing Systems*, 36.

Brust, C.-A. and Denzler, J. (2019). Integrating domain knowledge: using hierarchies to improve deep classifiers. In *Asian conference on pattern recognition*, pages 3–16. Springer.

Chen, M., Radford, A., Child, R., Wu, J., Jun, H., Luan, D., and Sutskever, I. (2020). Generative pretraining from pixels. In *International conference on machine learning*, pages 1691–1703. PMLR.

Cubuk, E. D., Zoph, B., Mane, D., Vasudevan, V., and Le, Q. V. (2018). Autoaugment: Learning augmentation policies from data. *arXiv preprint arXiv:1805.09501*.

Dang, H., Tran, T., Nguyen, T., and Ho, N. (2024). Neural collapse for cross-entropy class-imbalanced learning with unconstrained relu feature model. *arXiv preprint arXiv:2401.02058*.

Decelle, A., Seoane, B., and Rosset, L. (2023). Unsupervised hierarchical clustering using the learning dynamics of restricted boltzmann machines. *Physical Review E*, 108(1):014110.

Fang, C., He, H., Long, Q., and Su, W. J. (2021). Exploring deep neural networks via layer-peeled model: Minority collapse in imbalanced training. *Proceedings of the National Academy of Sciences*, 118(43):e2103091118.

Fellbaum, C. (2010). Wordnet. In *Theory and applications of ontology: computer applications*, pages 231–243. Springer.

Geirhos, R., Jacobsen, J.-H., Michaelis, C., Zemel, R., Brendel, W., Bethge, M., and Wichmann, F. A. (2020). Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673.

Gemmeke, J. F., Ellis, D. P., Freedman, D., Jansen, A., Lawrence, W., Moore, R. C., Plakal, M., and Ritter, M. (2017). Audio set: An ontology and human-labeled dataset for audio events. In *2017 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 776–780. IEEE.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Hinton, G., Vinyals, O., and Dean, J. (2015). Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Howard, A., Sandler, M., Chu, G., Chen, L.-C., Chen, B., Tan, M., Wang, W., Zhu, Y., Pang, R., Vasudevan, V., et al. (2019). Searching for mobilenetv3. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1314–1324.

Hu, W., Xiao, L., Adlam, B., and Pennington, J. (2020). The surprising simplicity of the early-time learning dynamics of neural networks. *Advances in Neural Information Processing Systems*, 33:17116–17128.

Hui, L., Belkin, M., and Nakkiran, P. (2022). Limitations of neural collapse for understanding generalization in deep learning. *arXiv preprint arXiv:2202.08384*.

Jin, P., Lu, L., Tang, Y., and Karniadakis, G. E. (2020). Quantifying the generalization error in deep learning in terms of data distribution and neural network smoothness. *Neural Networks*, 130:85–99.

Krizhevsky, A., Hinton, G., et al. (2009). Learning multiple layers of features from tiny images.

Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P. N., Hellmann, S., Morsey, M., Van Kleef, P., Auer, S., et al. (2015). Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic web*, 6(2):167–195.

Malashin, R. (2016). Extraction of object hierarchy data from trained deep-learning neural networks via analysis of the confusion matrix. *J. Opt. Tech.*, 83:599–603.

Mangalam, K. and Prabhu, V. U. (2019). Do deep neural networks learn shallow learnable examples first? In *ICML 2019 Workshop on Identifying and Understanding Deep Learning Phenomena*.

McInnes, L., Healy, J., and Melville, J. (2018). Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*.

Papyan, V., Han, X., and Donoho, D. L. (2020). Prevalence of neural collapse during the terminal phase of deep learning training. *Proceedings of the National Academy of Sciences*, 117(40):24652–24663.

Pezeshki, M. (2022). Dynamics of learning and generalization in neural networks.

Pinson, H., Lenaerts, J., and Ginis, V. (2023). Linear cnns discover the statistical structure of the dataset using only the most dominant frequencies. In *International Conference on Machine Learning*, pages 27876–27906. PMLR.

Qiu, G., Kuang, D., and Goel, S. (2024). Complexity matters: Dynamics of feature learning in the presence of spurious correlations. *arXiv preprint arXiv:2403.03375*.

Raghu, M., Gilmer, J., Yosinski, J., and Sohl-Dickstein, J. (2017). Svcca: Singular vector canonical correlation analysis for deep learning dynamics and interpretability. *Advances in neural information processing systems*, 30.

Raghu, M., Unterthiner, T., Kornblith, S., Zhang, C., and Dosovitskiy, A. (2021). Do vision transformers see like convolutional neural networks? *Advances in Neural Information Processing Systems*, 34:12116–12128.

Rahaman, N., Baratin, A., Arpit, D., Draxler, F., Lin, M., Hamprecht, F., Bengio, Y., and Courville, A. (2019). On the spectral bias of neural networks. In *International conference on machine learning*, pages 5301–5310. PMLR.

Redmon, J. and Farhadi, A. (2017). Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7263–7271.

Rogers, T. T. and McClelland, J. L. (2004). *Semantic cognition: A parallel distributed processing approach*. MIT press.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al. (2015). Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115:211–252.

Saxe, A. M., McClelland, J. L., and Ganguli, S. (2019). A mathematical theory of semantic development in deep neural networks. *Proceedings of the National Academy of Sciences*, 116(23):11537–11546.

Shah, H., Tamuly, K., Raghunathan, A., Jain, P., and Netrapalli, P. (2020). The pitfalls of simplicity bias in neural networks. *Advances in Neural Information Processing Systems*, 33:9573–9585.

Shwartz-Ziv, R. and Armon, A. (2022). Tabular data: Deep learning is not all you need. *Information Fusion*, 81:84–90.

Snell, J., Swersky, K., and Zemel, R. (2017). Prototypical networks for few-shot learning. *Advances in neural information processing systems*, 30.

Ulyanov, D., Vedaldi, A., and Lempitsky, V. (2018). Deep image prior. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9446–9454.

Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D., et al. (2016). Matching networks for one shot learning. *Advances in neural information processing systems*, 29.

Wang, S., Veldhuis, R., Brune, C., and Strisciuglio, N. (2022). Frequency shortcut learning in neural networks. In *NeurIPS 2022 Workshop on Distribution Shifts: Connecting Methods and Applications*.

Wang, S., Veldhuis, R., Brune, C., and Strisciuglio, N. (2023a). What do neural networks learn in image classification? a frequency shortcut perspective. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1433–1442.

Wang, Y., Sun, D., Chen, K., Lai, F., and Chowdhury, M. (2023b). Egeria: Efficient dnn training with knowledge-guided layer freezing. In *Proceedings of the Eighteenth European Conference on Computer Systems*, pages 851–866.

Xiao, T., Zhang, J., Yang, K., Peng, Y., and Zhang, Z. (2014). Error-driven incremental learning in deep convolutional neural network for large-scale image classification. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 177–186.

Xu, Z. J. and Zhou, H. (2021). Deep frequency principle towards understanding why deeper learning is faster. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 10541–10550.

Xu, Z.-Q. J., Zhang, Y., and Luo, T. (2024). Overview frequency principle/spectral bias in deep learning. *Communications on Applied Mathematics and Computation*, pages 1–38.

Xu, Z.-Q. J., Zhang, Y., Luo, T., Xiao, Y., and Ma, Z. (2019). Frequency principle: Fourier analysis sheds light on deep neural networks. *arXiv preprint arXiv:1901.06523*.

Yang, Y., Yuan, H., Li, X., Lin, Z., Torr, P., and Tao, D. (2023). Neural collapse inspired feature-classifier alignment for few-shot class incremental learning. *arXiv preprint arXiv:2302.03004*.

Zhang, X., Xiong, H., and Wu, D. (2020). Rethink the connections among generalization, memorization and the spectral bias of dnns. *arXiv preprint arXiv:2004.13954*.

Zhang, Y., Luo, T., Ma, Z., and Xu, Z.-Q. J. (2021). A linear frequency principle model to understand the absence of overfitting in neural networks. *Chinese Physics Letters*, 38(3):038701.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes, see Section 3]
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes, see Appendix D]
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries.[No]

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes]
   (b) Complete proofs of all theoretical results. [Yes, see Appendix A]
   (c) Clear explanations of any assumptions. [Yes]

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [No]
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes, see Section 4, Appendix B]
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes, see Section 3, Appendix C.1.2]
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes, see Appendix D]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. [Yes]
   (b) The license information of the assets, if applicable. [Not Applicable]
   (c) New assets either in the supplemental material or as a URL, if applicable. [Not Applicable]
   (d) Information about consent from data providers/curators. [Not Applicable]
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. [Not Applicable]
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

A METHODOLOGY

A.1 Relation of $A(\mathcal{R}, t)$ and $A(\mathcal{H}, t)$

Let $f_{\mathcal{H}}(x; \theta_t)$ denote a classifier trained in the hyponym label space $\mathcal{H}$ at epoch t. If $A(\mathcal{H}, t)$ represents the absolute classification accuracy in $\mathcal{H}$, then the classification accuracy in a random superclass label space, $A(\mathcal{R}, t)$, can be theoretically estimated based on the sizes of the subsets within $\mathcal{R} = \{r\}$.

To see this, let $D = \{x_i\}_{i=1}^{N}$ be the dataset with N examples and $A(\mathcal{X}, t) \in [0, 100]$ in every label space $\mathcal{X}$. Accuracy $A(\mathcal{H}, t)$ is essentially an estimation of probability $P_{\mathcal{H}}(x; \theta_t)$ that classifier f assigns correct hyponym label $\hat{y}_{\mathcal{H}}(x)$ to an image x:

$$A(\mathcal{H}, t)/100 \approx P_{\mathcal{H}}(x; \theta_t) = P(\hat{f}_{\mathcal{H}}(x; \theta_t) = \hat{y}_{\mathcal{H}}(x)), \tag{18}$$

where $\hat{f}_{\mathcal{H}}(x; \theta_t)$ is the predicted label.

Assume that we use classifier f to greedily predict the label in the random superclass label space $\mathcal{R}$, accordingly to Section 3.1. Let $\hat{f}_{\mathcal{R}}(x; \theta_t)$ be a superclass label that is predicted. The probability $P(\hat{y}_{\mathcal{R}}(x) = r)$ that a randomly chosen image x belongs to a superclass r can be estimated according to the size of the superclass in the dataset:

$$P(\hat{y}_{\mathcal{R}} = r) \approx P_r \triangleq \frac{|D_r|}{N_D}, \tag{19}$$

where $D_r = \{x \in D \mid \hat{y}_{\mathcal{R}}(x) = r\}$ is a set of examples with label r.

Since hyponyms are assigned to r independently, prior probability P_R that greedy hypernym classifier predicts correct label of superclass r:

$$P_R(\hat{f}_{\mathcal{R}}(x) = \hat{y}_{\mathcal{R}}(x) | \hat{y}_{\mathcal{R}}(x) = r) = P_r. \tag{20}$$

The probability of assigning a correct label in the superclass label space given a hyponym recognition probability $P_{\mathcal{H}}(x; \theta_t)$, can be estimated by considering two possible events: (a) correctly classifying the hyponym, or (b) misclassifying the hyponym but still assigning the correct superclass label by chance:

$$P_{\mathcal{R}}(x; \theta_t) = P_{\mathcal{H}}(x; \theta_t) + (1 - P_{\mathcal{H}}(x; \theta_t)) \sum_{r \in \mathcal{R}} P_r^2. \tag{21}$$

$P_{\mathcal{R}}(x; \theta_t)$ gives good estimate of $A(\mathcal{R}, t)$ in real experiments (Figure 7) except at the beginning of training, due to the nonuniform behavior of the randomly initialized network.

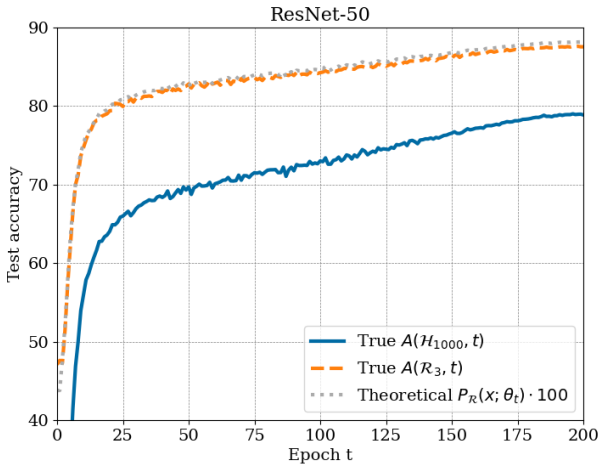

Figure 7: Experimentally observed accuracy in the random superclasses label space and the theoretical estimation of the same, given accuracy in the hyponym label space

A.2 Neural Collapse

Here we briefly overview Neural Collapse (NC) proposed by Papyan et al. (2020). Originally NC is defined only in hyponym label space (used for training), so we extend it to the hypernym label spaces.

Hyponym Label Spaces. Let $\mathbf{h}_{i,c} \in \mathbb{R}^p$ be a p-dimensional feature vector, where c is one of $|\mathcal{X}|$ classes, where $\mathcal{X}$ is the label space. In the original ImageNet hyponym label space ($\mathcal{X} = \mathcal{H}$) each class corresponds to a hyponym, so $|\mathcal{H}| = 1000$. The matrix $\mathbf{W} \in \mathbb{R}^{|\mathcal{H}| \times p}$ and vector $\mathbf{b} \in \mathbb{R}^{|\mathcal{H}|}$ represent the weights and bias of the last fully-connected layer. An image $\mathbf{x}_i$ is mapped to its feature representation $\mathbf{h}_i$ by all but one layers of the network f and the predicted label $\hat{f}_{\mathcal{H}}(\mathbf{x}_i)$ is determined by the index of the largest element in the vector $\mathbf{W}\mathbf{h}_i + \mathbf{b}$, as follows:

$$\hat{f}_{\mathcal{H}}(\mathbf{x}_i) = \arg\max_{c'} \langle \mathbf{w}_{c'}, \mathbf{h}_i \rangle + b_{c'}, \tag{22}$$

where $\mathbf{w}_{c'}$ is the c'-th row of $\mathbf{W}$.

The global mean is defined as $\boldsymbol{\mu}_G \triangleq \mathrm{Ave}_{i,c}\{\mathbf{h}_{i,c}\}$, and the train class means are defined as:

$$\boldsymbol{\mu}_c \triangleq \mathrm{Ave}_i\{\mathbf{h}_{i,c}\}, \quad c = 1, \ldots, |\mathcal{H}|, \tag{23}$$

where Ave is the averaging operator.

Additionally, the between-class covariance, $\boldsymbol{\Sigma}_{\boldsymbol{B}} \in \mathbb{R}^{p \times p}$ is defined as:

$$\boldsymbol{\Sigma}_{\boldsymbol{B}} \triangleq \mathrm{Ave}_c \left\{ (\boldsymbol{\mu}_c - \boldsymbol{\mu}_G)(\boldsymbol{\mu}_c - \boldsymbol{\mu}_G)^\top \right\}, \tag{24}$$

and the within-class covariance, $\boldsymbol{\Sigma}_{\boldsymbol{W}} \in \mathbb{R}^{p \times p}$ is defined as:

$$\boldsymbol{\Sigma}_{\mathbf{W}} \triangleq \mathrm{Ave}_{i,c} \left\{ (\mathbf{h}_{i,c} - \boldsymbol{\mu}_c)(\mathbf{h}_{i,c} - \boldsymbol{\mu}_c)^\top \right\}. \tag{25}$$

NC manifests through four key observations as described by Papyan et al. (2020).

(NC1) Variability collapse: $\boldsymbol{\Sigma}_{\boldsymbol{W}} \to \mathbf{0}$.

Following Papyan et al. (2020), in the Figure 5 of the paper, we plot $\mathrm{tr}\left(\Sigma_W \Sigma_B^\dagger / C\right)$, where tr denotes the trace operation and $[\cdot]^\dagger$ is the Moore–Penrose pseudoinverse.

(NC2) Convergence to simplex ETF:

$$|\|\boldsymbol{\mu}_c - \boldsymbol{\mu}_G\|_2 - \|\boldsymbol{\mu}_{c'} - \boldsymbol{\mu}_G\|_2| \to 0 \quad \forall\, c, c' \tag{26}$$

$$\langle \tilde{\boldsymbol{\mu}}_c, \tilde{\boldsymbol{\mu}}_{c'} \rangle \to \frac{C}{C-1}\delta_{c,c'} - \frac{1}{C-1} \quad \forall\, c, c', \tag{27}$$

where $\tilde{\boldsymbol{\mu}}_c = \frac{(\boldsymbol{\mu}_c - \boldsymbol{\mu}_G)}{\|\boldsymbol{\mu}_c - \boldsymbol{\mu}_G\|_2}$ are the renormalized class means.

Following the original paper, we plot the following metrics in Section C.2:

1. Norms equality level measured by two ratios: a) the standard deviation to the average length of class means and b) the standard deviation of the length of the rows of the weight matrix to their average length:

$$\beta_\mu = \mathrm{Std}_c\left(\|\boldsymbol{\mu}_c - \boldsymbol{\mu}_G\|_2\right) / \mathrm{Avg}_c\left(\|\boldsymbol{\mu}_c - \boldsymbol{\mu}_G\|_2\right), \tag{28}$$

$$\beta_w = \mathrm{Std}_c\left(\|\mathbf{w}_c\|_2\right) / \mathrm{Avg}_c\left(\|\mathbf{w}_c\|_2\right), \tag{29}$$

   where $\mathbf{w}_c$ is the c-th row of $\mathbf{W}$ (classifier of the c-th class).

2. Angles equality level measured as the standard deviation of cosines of the angles between class means (α_μ) and the standard deviation of cosines of angles between rows of weight matrix (α_w).

(NC3) Convergence to self-duality:

$$\left\| \frac{\mathbf{W}^\top}{\|\mathbf{W}\|_F} - \frac{\dot{\mathbf{M}}}{\|\dot{\mathbf{M}}\|_F} \right\|_F \to 0, \tag{30}$$

where $\dot{\mathbf{M}} = [\boldsymbol{\mu}_c - \boldsymbol{\mu}_G, c = 1, \ldots, C] \in \mathbb{R}^{p \times C}$ is the matrix obtained by stacking the class means into the columns of a matrix. Here, $\delta_{c,c'}$ is the Kronecker delta symbol.

We report value, estimated by equation (30), in Figure 5.

(NC4) Simplification to Nearest Class Centroid (NCC):

$$\arg\max_{c'} \langle \mathbf{w}_{c'}, \mathbf{h} \rangle + b_{c'} \to \arg\min_{c'} \|\mathbf{h} - \boldsymbol{\mu}_{c'}\|_2. \tag{31}$$

We report the proportion of mismatch between two ways of label estimation in Figure 5.

Hypernym Label Spaces. To estimate NC in the hypernym label space $\mathcal{S}$, we compute the mean of each superclass $s \in \mathcal{S}$:

$$\boldsymbol{\mu}_s \triangleq \text{Ave}_{c \in s}\{\boldsymbol{\mu}_c\}. \tag{32}$$

And we assume that the superclass weight matrix $\mathbf{W}_s \in \mathbb{R}^{|\mathcal{S}_n| \times p}$ consists of $|\mathcal{S}_n|$ rows, where each row $\mathbf{w}_s$ is obtained by averaging the rows of $\mathbf{W}$:

$$\mathbf{w}_s \triangleq \text{Ave}_{c \in s}\{\mathbf{w}_c\}, \tag{33}$$

and similarly for the bias vector $b_s \triangleq \text{Ave}_{c \in s}\{b_c\}$.

We can then use $\{\boldsymbol{\mu}_s\}, \mathbf{W}_s$, and $\mathbf{b}_s = [b_s]$ instead of $\{\boldsymbol{\mu}_s\}, \mathbf{W}$, and $\mathbf{b}$ as defined above, to estimate Neural Collapse in the hypernym spaces $\mathcal{S}$.

B EXPERIMENTAL DETAILS

In this section, we provide additional details on the experimental setup for the experiments discussed in Section 4.

B.1 Architectures and Training Parameters

The primary conclusions are based on experiments with ResNet; however, we also include results for ViT and MobileNet V3-L 1.0 to demonstrate that the findings generalize beyond convolutional or parameter inefficient architectures. Table 2 summarizes key parameters of the neural networks used in this study.

Table 2: Networks used in the experiments.

Name	Basic Block	#Params (M)	Top-1 Accuracy (%)
ResNet-18	Convolution	12.0	69.8
ResNet-50	Convolution	25.0	79.0
ResNet-152	Convolution	60.2	65.0
ViT-B/16	Attention	86.0	75.6
MobileNet V3-L 1.0	Convolution	5.4	75.6

All architectures were trained using 224 × 224 images. For ResNet-18, ResNet-50 and ViT-B/16 the following hyperparameters we used:

- 200 epochs;
- Cosine learning rate schedule;

- SGD optimizer with an initial learning rate of 0.05;
- AutoAugment (Cubuk et al., 2018);
- Loss function with Jensen-Shannon divergence.

MobileNet was trained for 600 epochs using RMSprop, with noise added to the learning rate, a batch size of 512, warmup, and Exponential Moving Average (EMA).

ViT training resulted in relatively low accuracy. Nevertheless, the presence of hypernym bias under suboptimal training conditions supports the hypothesis that the phenomenon is general.

ResNet-152 was trained in neural collapse settings (Papyan et al., 2020):

- No augmentation;
- 600 images per class in the train set;
- Batch size of 256;
- 350 epochs;
- SGD optimizer with initial learning rate 0.1;
- Learning rate is annealed by a factor of 10 at 1/2 and 3/4 of epochs.

These hyperparameters achieve nearly 100% accuracy on training set, which is necessary for NC to manifest, although validation accuracy is low in this case.

B.2 Constructing Hypernym Label Spaces

High-Level Hypernyms. At the top level of the WordNet tree, there are 9 synsets:

1. Plants (2 classes in ImageNet),
2. Geological formations (10 classes),
3. Natural objects (16 classes),
4. Sports (0 classes),
5. Fungi (7 classes),
6. People (3 classes),
7. Miscellaneous (42 classes),
8. Artifacts (522 classes),
9. Animals (398 classes).

Since the synsets have an uneven distribution of classes, we used the following groupings to form a top-level hypernym space, $\mathcal{S}_3$:

1. {1-7} (miscellaneous, 80 classes),
2. {8} (artifacts, 522 classes),
3. {9} (animals, 398 classes).

Low-Level Hypernym spaces. The low-level hypernym superclasses were created according WordNet tree. Similar to the 3 top-level hypernym superclasses, we aimed to maintain approximately equal class saturation in ImageNet classes, and some superclasses were formed by grouping several synonym sets at the same hierarchical level. For instance, to obtain 7 superclasses:

- The "miscellaneous" superclass remained unchanged.
- The "animals" superclass was divided into 3 new superclasses based on the lower levels of the WordNet hierarchy:
  1. Chordates (207 classes),
  2. Domestic animals (123 classes),
  3. Others (including invertebrates (61 classes) and game birds (7 classes)).

Similarly, the "artifacts" superclass was divided into the following three superclasses:

1. Instrumentality (358 classes),
2. Covering (85 classes),
3. Others (including categories like "commodity" and "decoration", totaling 79 classes).

For a more granular categorization, we divided the "animals" superclass into 18 new superclasses, such as "domestic cat", "reptile", and "terrier", each containing an average of 22 ImageNet classes.

The grouping procedure was performed only once in all cases. This was done to maintain consistency in the sizes of the hypernym sets, ensuring that the reported metrics are comparable. Highly imbalanced superclasses tend to yield higher accuracy regardless of their number, as can be seen from equations in Section A.1.

C ADDITIONAL EXPERIMENTS

C.1 Greedy Hypernym Classification

C.1.1 Confusion Matrix Evolution

A straightforward way to observe the manifestation of hypernym bias is by plotting a confusion matrix with class labels arranged according to the structure of the WordNet hierarchy. In this case, the order of classes was determined by performing a depth-first traversal of the tree, ensuring that sibling classes appear consecutively in the confusion matrix. We visualize the confusion matrices at the 1st, 5th, and 200th epochs of training ResNet-50 in Figure 8.

In the early stages, the block-diagonal structure is apparent, although the WordNet tree only partially mirrors the misclassification patterns. This can be observed in the structural discontinuities within the blocks.

C.1.2 Residual Error

To provide a clearer understanding of the training dynamics of ResNet-50, as shown in Figures 1 and 3, we introduce the residual relative error metric, E_R, defined as:

$$E_R(\mathcal{X}, t) = \frac{100 - A(\mathcal{X}, t)}{100 - A(\mathcal{X}, T)} - 1, \tag{34}$$

where $A(\mathcal{X}, t)$ is the accuracy at epoch t, and $A(\mathcal{X}, T)$ is the final accuracy after T epochs.

Figure 9 illustrates the residual relative error over the course of ResNet-50 training. The graph shows that the error is significantly higher for hypernym recognition during the first 10 to 20 epochs. After this initial phase, the residual relative error becomes similar across all three label spaces, indicating that the majority of the improvement in hypernym recognition accuracy occurs early in the training process.

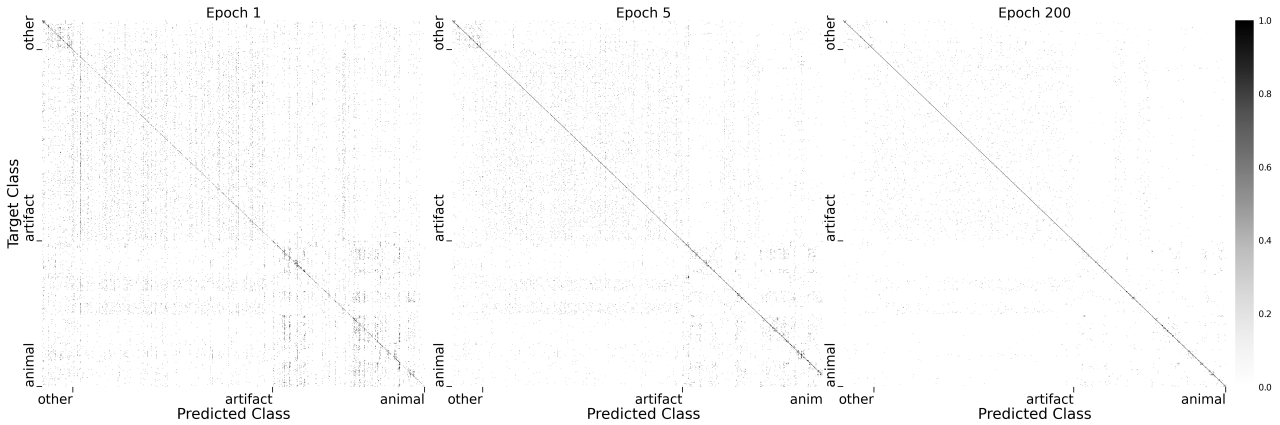

Figure 8: Confusion matrix of ResNet-50 during training. In early stages of training, block-diagonal structure is clearly visible.

C.1.3 Different Hypernym Spaces

Animal Hypernym Space. We examined the recognition of images within the "animals" hypernym label space using ResNet-50. This analysis is motivated by the well-known hierarchical structure of species, which resembles a tree-like hierarchy. Although the WordNet tree has known limitations, this structure is likely to provide a more accurate description for this portion of the ImageNet dataset. The results are presented in Table 3.

Table 3: Number of epochs required by ResNet-50 to achieve 95% relative accuracy in different labels spaces of varying sizes (animal images only)

n	$\mathcal{S}_n$	$\mathcal{R}_n$
2	6	46
3	12	96
4	13	105
7	20	109
8	25	114
18	48	114

Comparing the results in Table 3 with those for the entire set of classes (Figure 4), we observe that for a random division into 7 superclasses, 95% accuracy is achieved in 109 epochs for both cases. However, when utilizing the animal hierarchy, training converges faster, reaching the target in 20 epochs compared to 25.

Unbalanced Hypernym Space. To demonstrate that intuitive *manual* grouping of synsets inside the same hypernym level of the tree does not affect the conclusions, we also evaluated superclasses derived directly from one of the levels of the WordNet hierarchy without additional grouping for class balance. This resulted in 72 unbalanced superclasses, with the corresponding results presented in Table 4.

Table 4: Number of epochs to achieve 95% recognition accuracy using 72 superclasses

Architecture	$\mathcal{R}_{72}$	$\mathcal{S}_{72}$
ResNet-50	114	31
ResNet-18	88	31

C.1.4 Other Datasets

We have conducted additional experiments to provide further evidence of hypernym bias existence across different datasets and domains.

CIFAR-100. CIFAR-100 comprises 60,000 32×32 images distributed across 100 classes, which are further grouped into 20 superclasses. Let $\mathcal{S}_{20}$ represent the hypernym label space (superclasses) and $\mathcal{R}_{20}$ denote a

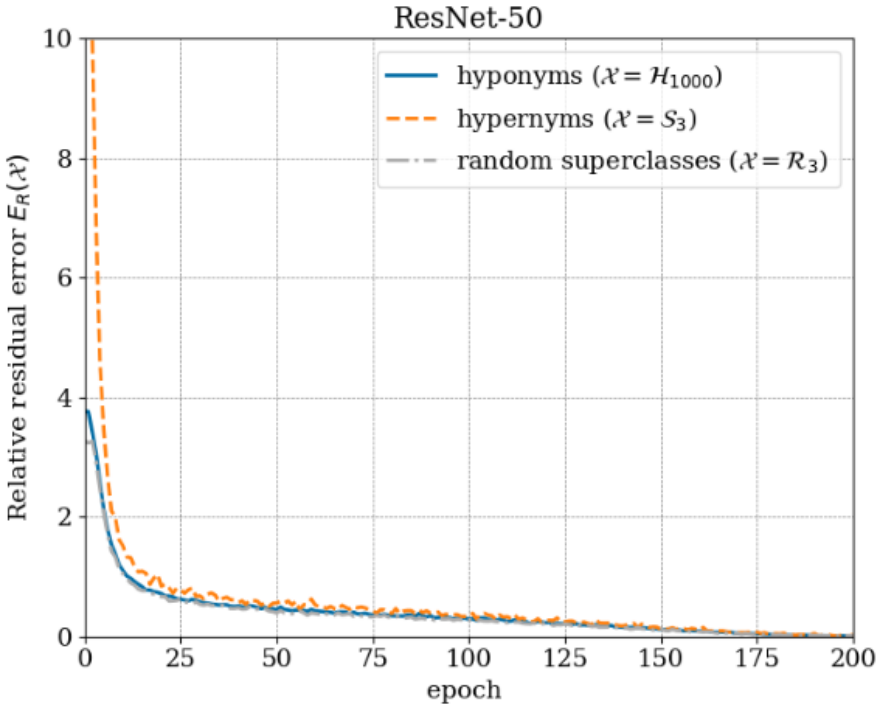

Figure 9: Residual relative error over the course of ResNet-50 training.

random label space isomorphic to $\mathcal{S}_{20}$.

We trained cifar-adapted ResNet-18 with augmentation and cosine annealing of learning rate and achieved 77% validation accuracy in the hyponym label space.

We employed the greedy classifier model described in Section 3.1 to estimate relative accuracy in hypernym and random label spaces. The results demonstrate an accelerated growth of $A_R(\mathcal{S}_{20})$ compared to $A_R(\mathcal{R}_{20})$, which aligns with our expectations. The manifestation of hypernym bias is evident but less pronounced than in Figure 1a (ImageNet). This discrepancy is fully explainable:

1. The hierarchical structure of CIFAR-100 is less well-defined compared to WordNet, with loosely distinct superclasses such as vehicles_1 and vehicles_2.
2. As noted in Section 4.1, larger hypernyms tend to exhibit a more pronounced bias. In CIFAR-100, each superclass comprises only 5 labels.

In this setup, the difference in dynamics between hypernyms and random superclasses is most evident on the training set, where the estimates are less noisy. This is particularly noticeable when $A_R(\mathcal{X}) \approx 0.6$ as shown in Figure 10.

We also observe a consistent manifestation of hypernym bias across a variety of training parameters on CIFAR-100.

DBPedia. DBPedia is a text classification dataset containing 630,000 samples. It includes 14 classes, which we grouped into 5 hypernyms ($\mathcal{S}_5$) based on the first level of the DBPedia ontology as shown in Table 5.

We fine-tuned a pretrained BERT-base model on DBPedia for 3 epochs using a batch size of 32 and a learning rate of $2 \cdot 10^{-5}$. The test errors were as follows: 0.67% for hyponym labels, 0.48% for random superclass labels, and 0.37% for hypernym labels. The relative accuracy curves demonstrate that the difference between $A(\mathcal{R}_5)$ and $A(\mathcal{S}_5)$ emerges during the early iterations, as predicted. Hypernym bias is evident.

Thus, we demonstrated the generalizability of our approach to two domains (images and text)

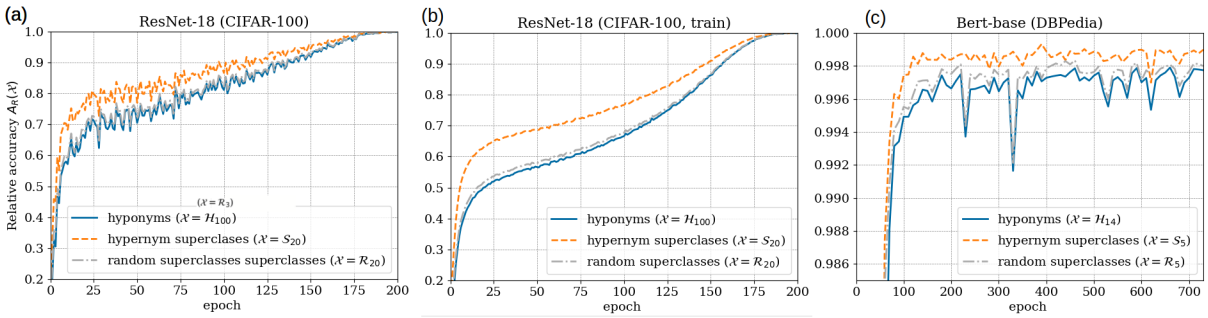

Figure 10: Generalization of hypernym bias. Relative accuracy in different labelspaces for CIFAR-10 (a,b) and DBPedia (c) classifiers during course of training. Graph for DBPedia shows only first 750 training iterations

Table 5: Superclasses structure according to DBPedia ontology

Superclass	**Classes**
Eukaryote	Artist, Athlete, OfficeHolder, Animal, Plant
Organization	Company, EducationalInstitution
Place	NaturalPlace, Village
Work	Album, Film, WrittenWork
Others	MeanOfTransportation, Building

C.2 Neural Collapse

ETF convergence estimation. Figure 5 shows the curves for NC1, NC3, and NC4 in the label spaces of top-level hypernyms $\mathcal{S}_3$, random superclasses $\mathcal{R}_3$, and hyponyms $\mathcal{H}_{1000}$. Here, in Figure 11, we provide additional graphs that illustrate the level of convergence to the equiangular tight frame (ETF), as computed according to Appendix A.2.

As can be seen from the figure we do not observe convergence to ETF in the hypernym label space.

UMAP visualization. In Figure 12, we show UMAP embeddings of the penultimate features of two versions of trained ResNet-152: one trained with and one trained without horizontal flip augmentation.

As shown in Figure 12, data augmentation prevents the neural network from reaching 100% accuracy, and Neural Collapse (NC) is not observed. Similarly, NC is absent on the validation set. The graphs in Figure 5 of and Figure 11 similarly to those in (Papyan et al., 2020), display only moderate convergence towards neural collapse in ImageNet. However, NC is evident in UMAP embeddings shown in Figure 1, suggesting complete removal of hyponymy information from penultimate features. Yet, when we plot the UMAP embeddings of the class centers, the hypernym structure remains preserved until the final epoch, although information gradually decays (see Figure 13).

This is likely due to averaging, which effectively reduces noise. We anticipate that training a larger network for a longer period will result in full NC on ImageNet.

C.3 Manifold Distances

Here we briefly compare the method of defining class distances based on mutual covers in feature space, as described in Section 3.2, with other approaches. Interestingly, these simpler methods also exhibit high cophenetic correlation coefficient (CCC) between the distances in feature space and hyponym distance in the WordNet graph.

Let $\mathbf{h}^l_{i,c}$ be ResNet-50 feature of layer l in response to an image with hyponym label c[3]. We use UMAP with default parameters to create 10-dimensional embeddings $\boldsymbol{\phi}^l_{i,c}$ of the features, these embeddings are used further.

Class Means Distance. We can construct the class distance matrix D_F using distances between class means

[3]For layers below the pre-logit layer, we applied global max-pooling to the feature maps before estimating distances.

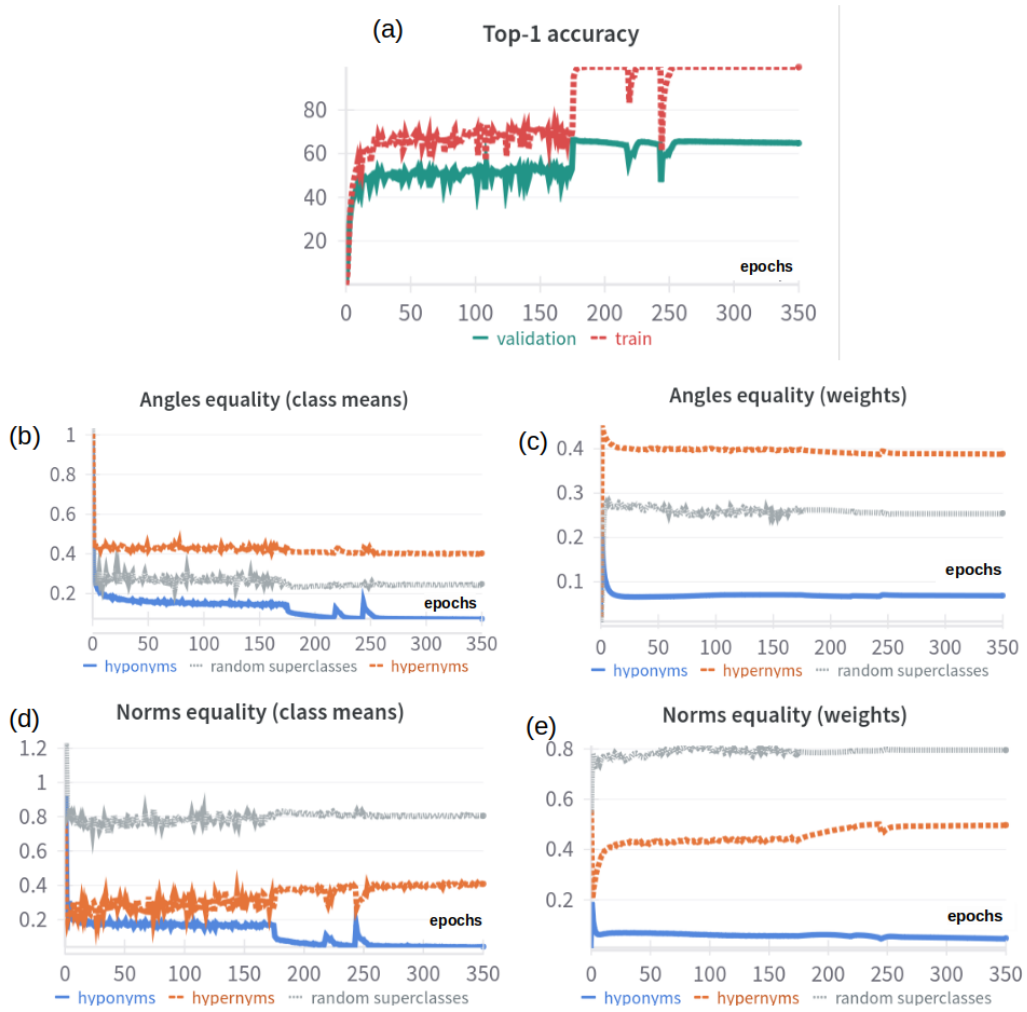

Figure 11: Train and validation accuracy during training in NC settings (a), level of angle equality between class means α_μ (b) and rows of the weight matrix α_w (c), level of norm equality of class means β_μ (d) and rows of weights matrix β_w (e)

$d_f(c, c') = \|\boldsymbol{\mu_c} - \boldsymbol{\mu_{c'}}\|_2$, where class means $\mu_\mathbf{c}$ is average embeddings of class c (similarly to equation (23)).

Correlation from Direct Comparison of Examples. We can omit the explicit computation of class distances by directly calculating the correlation coefficient between individual distances $d_f(u^{c_i}, v^{c_j})$ in the feature (embedding) space and class distances $d_w(c_i, c_j)$ in the WordNet graph.

Results for all methods are reported in Table 6.

From the table, method based on mutual cover provides higher CCC in bottom layers, while class means distances preserve more information about hypernym relation in top layers. Mutual cover distances, though, provide CCC that gradually increase from bottom to top layers.

C.4 Linear Probes

We experimented with two methods of training the linear probes:

1. Additional fully-connected layers. This technique was proposed by Alain and Bengio (2016). The additional layers are trained simultaneously with the entire network, but gradients from the probes do not propagate

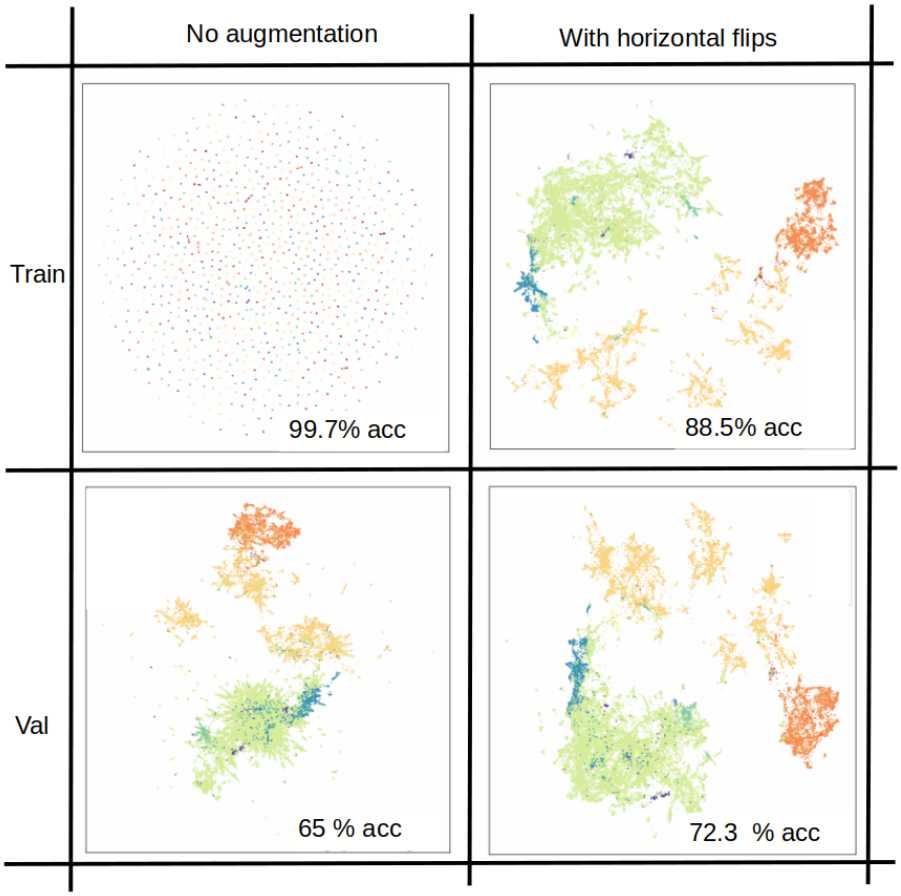

Figure 12: UMAP embeddings of penultimate features for 50 training and 50 validation examples per class in the end of training. Two regimes of training are considered: with and without augmentation. The color indicates the WordNet distance to the anchor class.

into the network itself.

2. Few-shot learning. In this approach, features from several examples per class are collected during a specific phase of training and used to train a linear model. This probing method has been employed in previous studies, such as (Alexey Dosovitskiy, 2020; Raghu et al., 2021). We applied two types of linear learners:
   - An analytical solution using the least squares method with regularization as proposed by Raghu et al. (2021).
   - Logistic regression.

The advantage of using additional layers trained simultaneously with the network is that the entire training set can be fully utilized, though the adaptation is influenced by the continuously changing weights of the neural network. In contrast, few-shot learning requires more memory but provides a solution tailored to the specific state of the network.

In all cases, we trained the probes in the hyponym label space $\mathcal{H}_{1000}$ and evaluated accuracy in both the hypernym label space $\mathcal{S}_3$ and the hyponym label space $\mathcal{H}_{1000}$. Training in $\mathcal{H}_{1000}$ provides a regularization effect and is more effective in hypernym labels too, as shown in Table 7.

C.4.1 Additional Layers

The linear layers were trained for ResNet-18 using the same optimizer and parameters as the entire network, but with cross-entropy loss as the objective function for all intermediate layers, without any additional loss components.

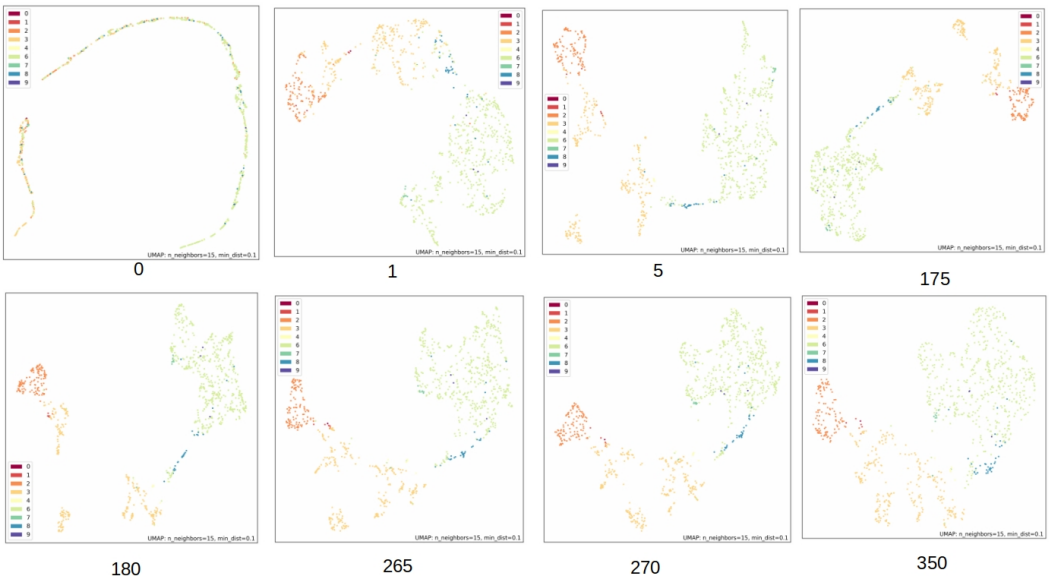

Figure 13: UMAP embeddings of the class means of the penultimate features on different epochs. The color indicates the WordNet distance to the anchor class.

Table 6: Cophenetic correlation coefficient of WordNet graph distances and distances between features of different labels (ResNet-50)

Layer name	Direct comparison	Class Means	Mutual Cover
Layer1 (max pool)	0.06	0.26	**0.31**
Layer2 (max pool)	0.20	0.44	**0.44**
Layer3 (max pool)	0.49	**0.62**	0.56
Pre-logits	0.58	**0.61**	0.59
Logits	0.59	**0.62**	0.60
Average	0.32	0.44	**0.45**

We followed the recommendations of Alain and Bengio (2016):

- A linear layer is added at the end of each convolutional block.
- The weights of the linear layer are randomly initialized.
- To reduce the number of parameters, the feature maps are downsampled in both height and width using average pooling by a specified factor.
- Evaluation is conducted on the validation set.

We conducted experiments using feature maps at various resolutions using different average pooling window sizes.

C.4.2 Few-Shot Linear Learner

For few-shot training, we use 10 examples per class sampled from a portion of the validation set, and evaluate the classifier on the remaining part of the validation set.

Initially, we followed the approach described in (Raghu et al., 2021), which employs an analytical solution. Additional investigations, described further, revealed that the analytical solution performed significantly worse in the bottom layers of the neural network compared to trainable linear layers. Compared to the analytical

Table 7: Hypernym linear probes classification accuracy $A(\mathcal{S}_3)$ using different label spaces for training. Results are for the ResNet-18 layers and the logistic regression probes.

Layer	Hypernym ($\mathcal{S}_3$)	Hyponym ($\mathcal{H}_{1000}$)
layer1	70,4	**73,2**
layer2	77,9	**79,6**
layer3	87,3	**87,7**
layer4	93,4	**95,5**
logits	92,6	**95,8**

solution, iterative logistic regression yielded slightly more robust results, which are reported in the paper, though it remains inferior to trained layers.

ResNet. For ResNets we used features obtained similarly to trained probes with medium size average pooling window.

ViT. For ViT, to connect probes we tested the averaged token embedding as suggested by Chen et al. (2020) and classification token embedding only as done in (Raghu et al., 2021). Classification token worked significantly better and we report results with the usage of it only. The transformer-decoder processes a sequence of discrete input tokens, $x_1, \ldots, x_n$, and generates a d-dimensional embedding for each position. The decoder consists of a stack of L blocks, with the l-th block producing an intermediate embedding, $h^1, \ldots, h^l$, each of dimension d.

The ViT block updates the input tensor h^l as follows:

$$\begin{aligned} n^l &= \text{layer\_norm}(h^l), \\ a^l &= h^l + \text{multihead\_attention}(n^l), \\ h^{l+1} &= a^l + \text{mlp}(\text{layer\_norm}(a^l)). \end{aligned}$$

The probe is taken based on the first normalized classification token embedding n_0^l at each layer l, where 0 is the index of classification token.

C.4.3 Results

Comparison. For ResNet-18, training probes provide significantly better results than few-shot learning (Figure 14). Logistic probes are slightly better than those based on analytical solutions.

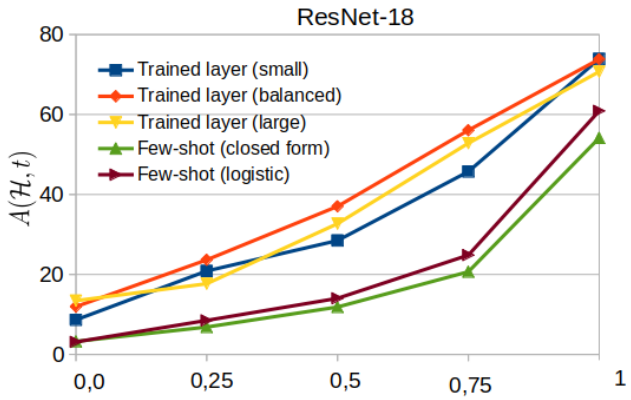

Figure 14: Comparison of probes types for fully trained ResNet-18. Small, balance, and large trainable probes were obtained using features with large, medium and small pooling windows, respectively. The horizontal axis represents the normalized layer depth.

However, we have observed that trainable layers doesn't change relative accuracy of probing and the same conclusions can be made. Therefore, in the paper we present results for logistic regression probes only, but we verify conclusions for several architectures.

Probes for ViT and ResNet-18. Figure 6 present results for few-shot linear probes applied to ResNet-50. Here, in Figure 15, we extend the analysis to include additional experiments with ViT-B/16 and ResNet-18.

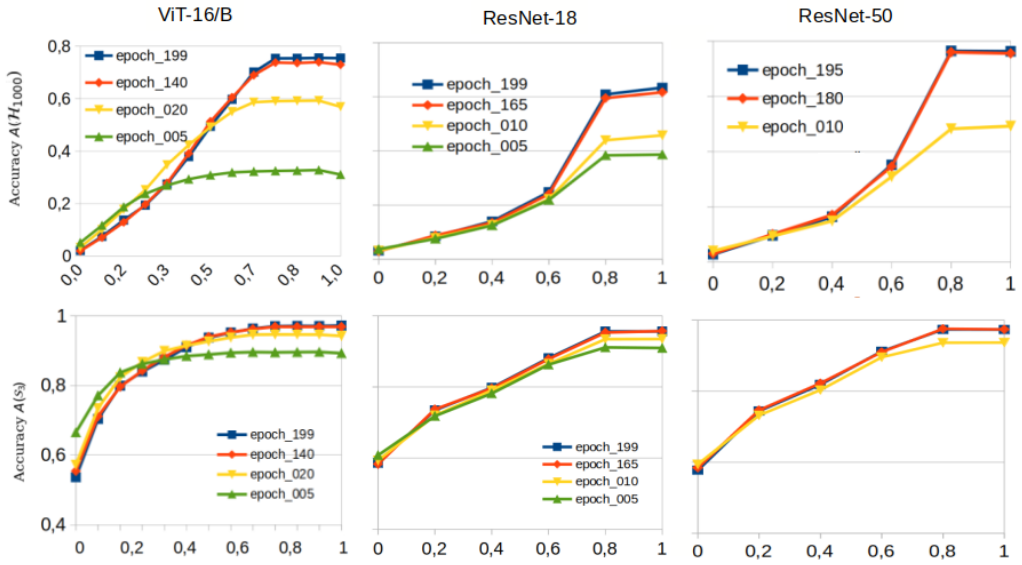

Figure 15: Classification accuracy based on linear probes for different architectures in different phases of training. The horizontal axis of each graph represents the normalized layer depth. The top row corresponds to the hyponym label space $\mathcal{H}_{1000}$, while the bottom row corresponds to the hypernym label space $\mathcal{S}_3$.

The following conclusions can be drawn from the graphs in Figure 15:

1. Classification accuracy improves as the network progresses towards the top layer, for both hyponym and hypernym classification. This trend is observed across all architectures.
2. During the early stages of ViT training, classification accuracy based on the initial layers is higher compared to the end of training. This pattern is evident for both the hyponym and hypernym label spaces and can be attributed to the removal of excess information from the classification token as training progresses.
3. In all experiments, hypernym classification accuracy using intermediate layers is consistently lower than when using the top layer. Therefore, we cannot conclude that the early layers of the neural network contain sufficient information for accurate hypernym recognition.

C.5 Image Frequency Analysis

One can suggest that recognizing hypernyms is possible based on low image frequencies, which are learned in first phase of training, while hyponyms require high frequencies that are learned later (Figure 16). This explanation through Frequency Principle seems intuitive and lead to interesting interpretations of the fact that infants perceive world in low frequencies. However our experiments did't reveal this relation.

We removed high frequencies from the images using Gaussian kernel of different size. The focus was on the accuracy of hypernym recognition at the beginning and at the end of training when using a significant level of blurring. For this, we used the 10th and 195th epochs of ResNet-50 training and the 5th and 140th epochs of ViT-B/16.

As can be seen from the relative accuracy gain graph depicted in Figure 17 and Figure 18 there is no tendency to depend on low frequencies more in the beginning of training than in the end of training for both hypernyms and hyponyms.

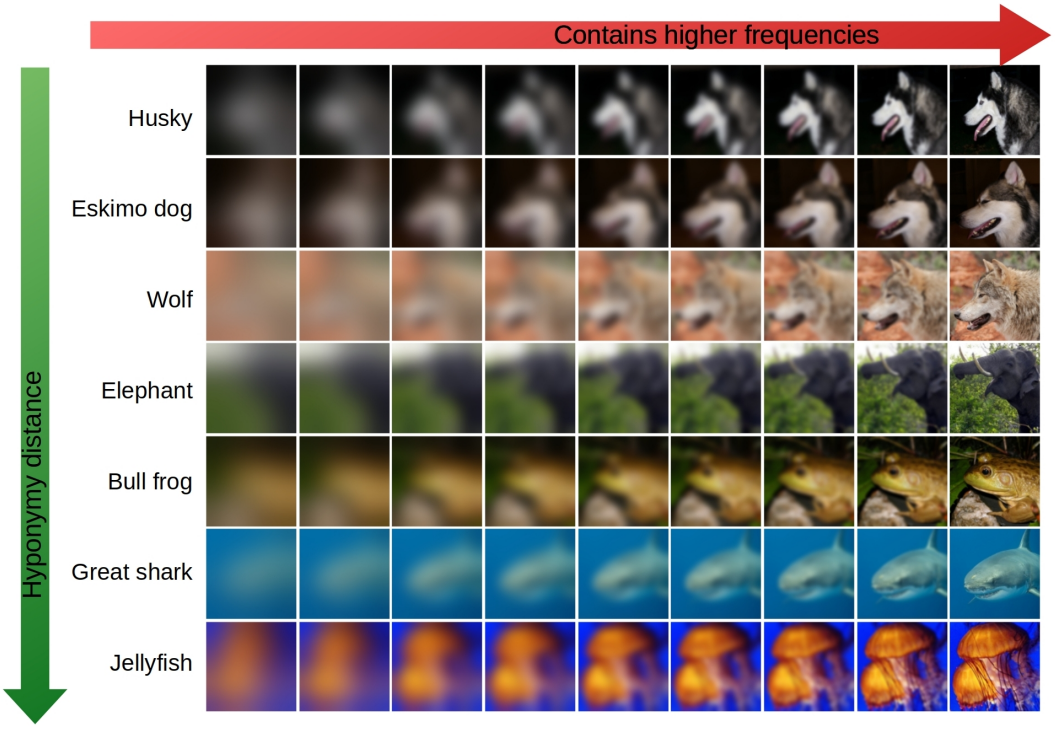

Figure 16: Examples of images from classes with different hyponymy distances and varying amounts of high-frequency content. Each column applies the same low-pass filter parameters, producing a consistent level of detail across all classes, while rows represent different categories with increasing hyponymy distance from the class presented in the top row.

D Complexity and infrastructure

For experiments we used NVIDIA A6000 and RTX 3090 gpus.

Our framework consists of three components, each with its own computational complexity:

1. **Greedy classifier model**. Involves single argmax computation over logits during forward pass. Mappings between labelspaces are efficiently managed using hash tables. Computational burden is negligible in practice.

2. **Manifold assessment**. Complexity: $O(EL \cdot (N \log N \cdot p + K^2 d + c^2 K^2))$, where

   - E and L are number of checkpoints and layers to consider respectively,
   - $N = 2K$ is number of data points (query and support sets),
   - c is number of classes,
   - p is feature size,
   - d is number of UMAP output dimensions.

   GPU acceleration is utilized for computing the distance matrix and mutual covers. In our implementation, the experiments described in the paper required approximately 40 GB of GPU memory.

3. **Neuronal collapse estimation**. Main computation burden comes from penultimate feature extraction. In our (suboptimal) implementation number of forward passes: $(2 + M)SE$, where

   - $S = 6 \times 10^5$ (size of balanced ImageNet),
   - $M = 1$ (number of extra label spaces),
   - $E = 350$ (checkpoints).

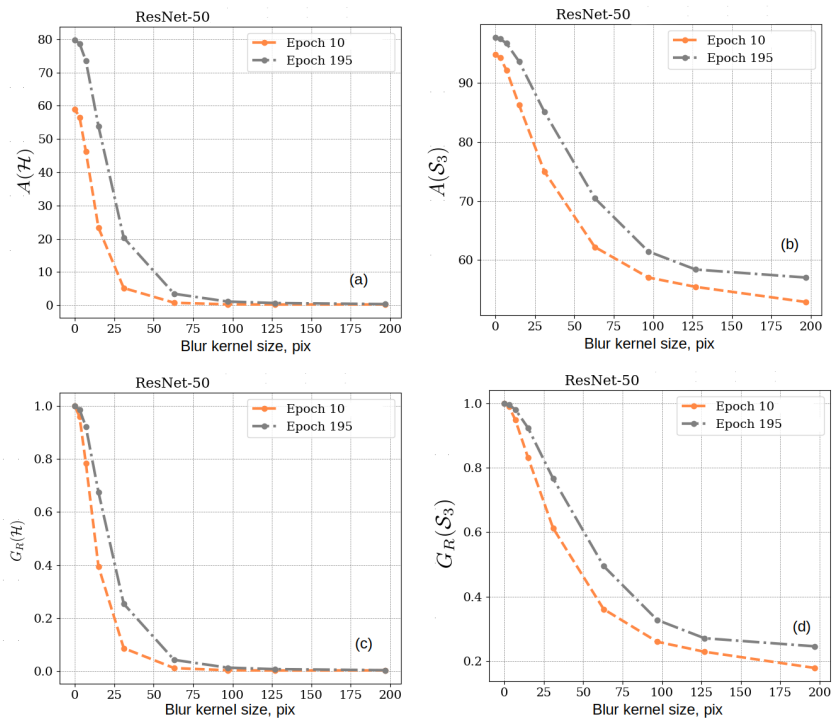

Figure 17: Accuracy (a,b) and relative accuracy gain (c,d) of ResNet-50 in hyponym label space (a,c) and hypernym label space (b,d) with the respect to size of the blur kernel. G_R is computed with respect to accuracy on undistorted images for the same checkpoint.

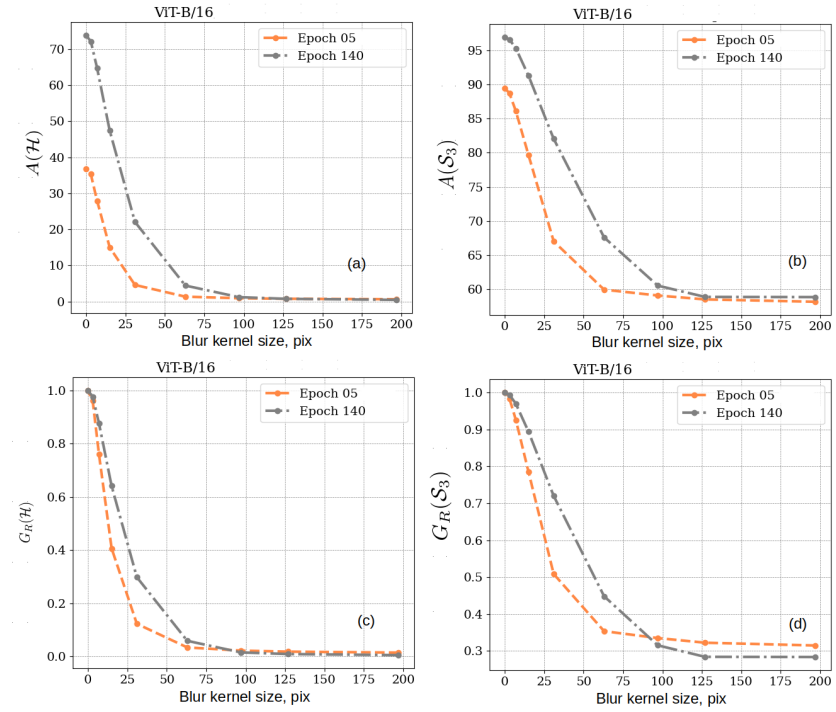

Figure 18: Accuracy (a,b) and relative accuracy gain (c,d) of ViT-B/16 in hyponym label space (a,c) and hypernym label space (b,d) with the respect to size of the blur kernel. G_R is computed with respect to accuracy on undistorted images for the same checkpoint