Supplementary Material of Diverse Imagenet Models Transfer Better

A. Measuring Transferability

Much of the analysis in this work requires comparing accuracies across datasets of differing difficulty. Directly comparing the top-1 accuracy across datasets is problematic (e.g., as done in [37, 40]). The meaning of a 1% additive increase in accuracy is different if it is relative to a base accuracy of 50% vs. 99%. Instead, we follow the transferability evaluation protocol proposed by [44] and consider the log odds, i.e., the accuracy after the $logit$ transformation $logit(p) = log(p/(1-p)) = sigmoid^{-1}(p)$. We repeat the details here for completeness: The logit transformation is the most commonly used transformation for analysis of proportion data, and an additive change Δ in logit-transformed accuracy has a simple interpretation as a multiplicative change e^{Δ} in the odds of correct classification. Given logit-transformed accuracies y_m^d of model $m \in \mathcal{M}$ on dataset $d \in \mathcal{D}$, we compute adjusted accuracies $acc(m,d) = y_m^d - \sum_{m' \in \mathcal{M}} y_{m'}^d / |\mathcal{M}|$. For each model, we take the mean and standard error of the adjusted accuracy across datasets, and multiply the latter by a correction factor $|\mathcal{M}|/(|\mathcal{M}|-1)$.

B. Training and Fine-tuning Settings

Our pipeline consists of three stages we shall refer to as Pretrain, Label Injection and Transfer Learning. The Pretrain may be any SSL model, while we experiment with MoCo-v2, SwAV, SimCLR, DINO and MAE. This stage may be completely skipped for standard ImageNet Supervised Training. During Label Injection we train the initialized model in an augmented supervised manner on ImageNet while preserving label diversity (see Section 4). Finally we evaluate performance of the proposed networks on downstream tasks by either fine-tuning the entire network or just the last layer, i.e. linear probing, in a standard supervised manner with identical hyper-parameters for all datasets.

B.1. Label Injection

We train Resnet50 following [80] on the full ImageNet [47] training set using a NVIDIA 8×V100 GPU cluster, RandAugment [19], batch size 200, Cosine learning rate schedule [54] with one cycle, 3 warmup epochs with lr=$1e^{-4}$, initial learning rate $1e^{-1}$, using SGD optimizer [9] with weight decay of $1e^{-4}$ and model EMA for 100 epochs unless otherwise stated (e.g Vanilla from scratch). Fully connected layer is initialized using Linear Probe or from class centroids. Label injection control cycle $\mathcal{T} \in [1, 2, 3, 4, 5, 10, 50, 100]$ where $\mathcal{T} = 1$ is effectively a standard fine-tuning. ImageNet top-1 accuracy is reported.

ViT models follow the ViT-B architecture and finetuning scripts from [32] with batch size 32, AdamW optimizer [55] with base learning rate $5e^{-4}$, layer decay 0.65, weight decay 0.05, drop path 0.1, input resolution 224, mixup [84] 0.8, cutmix [82] 1.0 and cutout [22] 0.25 for 100 epochs. We do not use gradient accumulation effectively reducing out batch size.

B.2. Fine-tuning

The same fine-tuning settings is used for all downstream datasets, without a designated hyperparameter search for each one. The single hyperparameter set used for all fine-tuning used Adam with Momentum 0.9, Weight decay with a $1e^{-4}$ coefficient, single cycle cosine learning rate schedule, AutoAugment [18], batch size of 128, cutout with length 0.5, model exponent moving average (EMA) and label smoothing [74] with a 0.1 coefficient for 60 epochs on a single NVIDIA V100 GPU. The initialization of the fully connected is detailed in Appendix I.

For ViT fine-tuning we use the same scripts and parameters used for label injection except increase in gradient accumulation to 4 which increases the effective batch size to 128.

C. Filters of Neural Layers

In this section we specify the way we extract filters out of the weight tensors of convolutional [48], fully-connected and multi-head self-attention (MSA) layers. [76], that compose modern Resnets [35] and vision transformers (ViT) [25].

C.1. Convolutional Layers Filters.

Denote by $W^{(l)} \in \mathbb{R}^{k \times k \times n_l \times n_{l+1}}$ the weights of the l-th convolutional layer of kernel size k and n_l, n_{l+1} the number of input and output channels respectively. Thus, every weight matrix can be reshaped to $\hat{W}^{(l)} \in \mathbb{R}^{k^2 \cdot n_l \times n_{l+1}}$, such that the i-th filter of the l-th layer, $\mathbf{w}_i^{(l)} \in \mathbb{R}^{k^2 \cdot n_l}$ with $i \in \{1, \dots, n_{l+1}\}$, is the i-th column of the reshaped weight matrix $\hat{W}^{(l)} = [\mathbf{w}_1^{(l)}, \dots, \mathbf{w}_{n_{l+1}}^{(l)}]$.

C.2. Fully-connected Layers Filters.

The l-th fully-connected layer performs a vector-matrix multiplication with a weight matrix $W^{(l)} \in \mathbb{R}^{w_l \cdot h_l \cdot n_l \times n_{l+1}}$, where w_l, h_l are the spatial width and height of the input tensor. Thus, its i-th filter, $\mathbf{w}_i^{(l)} \in \mathbb{R}^{w_l \cdot h_l \cdot n_l}$ with $i \in \{1, \dots, n_{l+1}\}$, is the i-th column of the weight matrix $\hat{W}^{(l)} = [\mathbf{w}_1^{(l)}, \dots, \mathbf{w}_{n_{l+1}}^{(l)}]$.

C.3. Multi-head Self-attention Layers Filters.

A multi-head self-attention (MSA) layer [76] operates directly on its input $x^{(l)} \in \mathbb{R}^{w_l \cdot h_l \cdot n_l}$ by dividing it into $H^{(l)}$ groups of channels $x^{(l)} \in \mathbb{R}^{w_l \cdot h_l \cdot \frac{n_l}{H^{(l)}}}$ and applying direct three vector-matrix multiplications on each group with the corresponding matrices $Q_h^{(l)}, K_h^{(l)} \in \mathbb{R}^{w_l \cdot h_l \cdot \frac{n_l}{H^{(l)}} \times d_{QK}}$ and $V_h^{(l)} \in \mathbb{R}^{w_l \cdot h_l \cdot \frac{n_l}{H^{(l)}} \times d_V}$ for $h = 1, \dots, H^{(l)}$ and some desing parameters $d_{QK}, d_V \in \mathbb{N}_+$. For each head $h = 1, \dots, H^{(l)}$:

- The i-th filter, $q_{h,i}^{(l)} \in \mathbb{R}^{w_l \cdot h_l \cdot \frac{n_l}{H^{(l)}}}$ with $i \in \{1, \dots, d_{QK}\}$, is the i-th column of the weight matrix $Q_h^{(l)} = [q_{h,1}^{(l)}, \dots, q_{h,d_{QK}}^{(l)}]$.
- The i-th filter, $k_{h,i}^{(l)} \in \mathbb{R}^{w_l \cdot h_l \cdot \frac{n_l}{H^{(l)}}}$ with $i \in \{1, \dots, d_{QK}\}$, is the i-th column of the weight matrix $K_h^{(l)} = [k_{h,1}^{(l)}, \dots, k_{h,d_{QK}}^{(l)}]$.
- The i-th filter, $v_{h,i}^{(l)} \in \mathbb{R}^{w_l \cdot h_l \cdot \frac{n_l}{H^{(l)}}}$ with $i \in \{1, \dots, d_V\}$, is the i-th column of the weight matrix $V_h^{(l)} = [q_{h,1}^{(l)}, \dots, q_{h,d_V}^{(l)}]$.

Effectively, for the purpose of quantifying the filter diversity of a MSA layer, the filter diversity of each such direct operation on the input is quantified separately and eventually averaged together with all the others.

D. Threshold-Agnostic Clustering Filter Diversity

As explained in section 3.1, the agglomerative clustering stops when the similarity between all pairs of clusters is below the threshold τ (Equation (2)) . We define the *cluster ratio* $\mathcal{C}_\tau(W)$ for a given threshold τ as the ratio between the number of clusters and the number of filters. Note, that the more filters are clustered together, the lower will be the cluster ratio, indicating low overall filter diversity.

A well known main caveat of agglomerative clustering is its sensitivity to the threshold τ. Moreover, due to different neural layers of the same model learning different levels of abstractions, a single threshold τ value does not fit all. We observe this sensitivity also in our experiments, illustrated in Figure 7 (Left), which shows that the cluster ratio $\mathcal{C}_\tau(W)$ is highly sensitive to the choice of thresholds. At the same time, we note that layers that maintain high cluster ratio for high values of τ are more diverse. We wish to capture this property to obtain a threshold-agnostic measure. Hence, differently from [3], we evaluate the clusterability of the entire model by averaging the cluster ratio of layers across a spectrum of threshold values.

This is done by computing the area under the curve (AUC) $\mathcal{D}_C(W^{(l)}) = \int_0^1 \mathcal{C}_\tau(W^{(l)}) d\tau$ for each layer $l = 1, \dots, L$, and averaging over all layers, as shown in Figure 7 (Right). Thus, the final clustering-based filter diversity measure is computed as:
$\bar{\mathcal{D}}_C = \frac{1}{L} \sum_{l=1}^{L} \mathcal{D}_C(W^{(l)})$.

E. Spectral Filter Diversity

Here we provide the technical details of the computation of the proposed Spectral Filter Diversity in section 3. We compute the singular value decomposition (SVD) [73] of the filter covariance matrix $W^T \cdot W$ to get the orthogonal singular vectors U_W and the rectangular diagonal matrix of positive singular values Σ_W^2, such that, $\sigma_{W1}^2 > \sigma_{W2}^2 > \cdots > \sigma_{Wd}^2 > 0$ respectively, and $W^T \cdot W = U_W \Sigma_W^2 U_W^T$ with $\{\sigma_{Wi}\}_{i=1}^d$ the non-zero eigenvalues of W. The variance explained by the K first principal components can be computed as: $\bar{\Sigma}_W^K = \frac{\sum_{k=1}^K \sigma_{Wk}^2}{\sum_{k=1}^d \sigma_{Wk}^2}$. The faster $\bar{\Sigma}_W^K$ increases, the more variance is explained by fewer principal components implying lower diversity. This can be quantified by the area under the $\bar{\Sigma}_W^K$ curve, i.e. $\sum_{K=1}^d \bar{\Sigma}_W^K$, as intuitively illustrated in Figure 8. Finally, we define the spectral filter diversity of layer $l = 1, \dots, L$ as $\mathcal{D}_\Sigma(W^{(l)}) = 1 - \sum_{K=1}^d \bar{\Sigma}_{W^{(l)}}^K$ and the overall filter diversity of a model is taken as the average over all layers: $\bar{\mathcal{D}}_\Sigma = \frac{1}{L} \sum_{l=1}^L \mathcal{D}_\Sigma(W^{(l)})$.

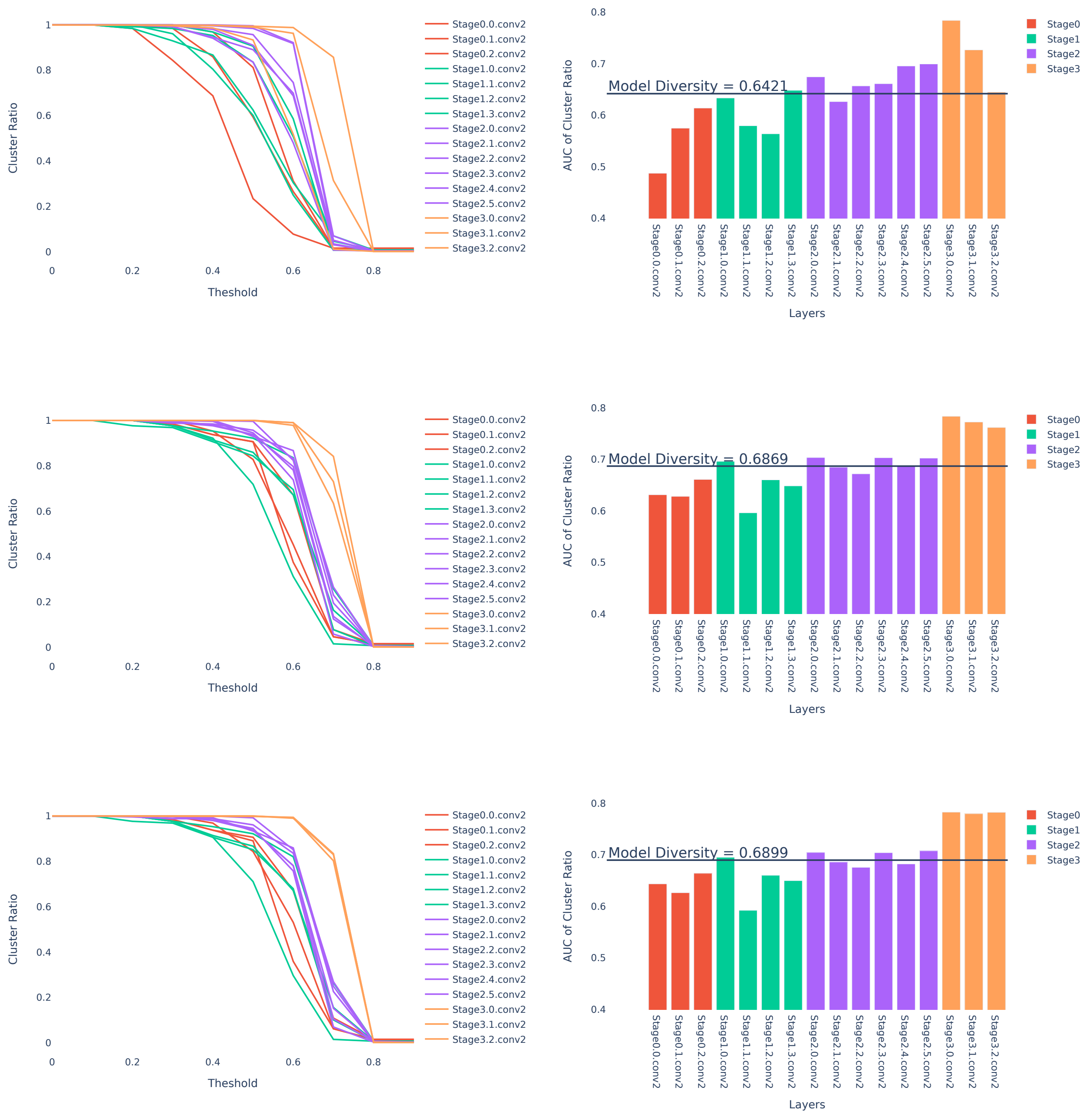

Figure 7. Clustering Diversity of Convolutional Layers of Supervised (Top), SwAV($\mathcal{T} = 4$) (Center) and SwAV (Bottom) pretrained models. (Left) The cluster ratio $\mathcal{C}_\tau(W)$ of various convolutional layers of Resnet50 as a function of the clustering threshold τ. (Right) The corresponding area-under-curve (AUC) of the cluster ratio $\mathcal{D}_C(W^{(l)})$. The average AUC over all curves is taken as the overall filter diversity measure.

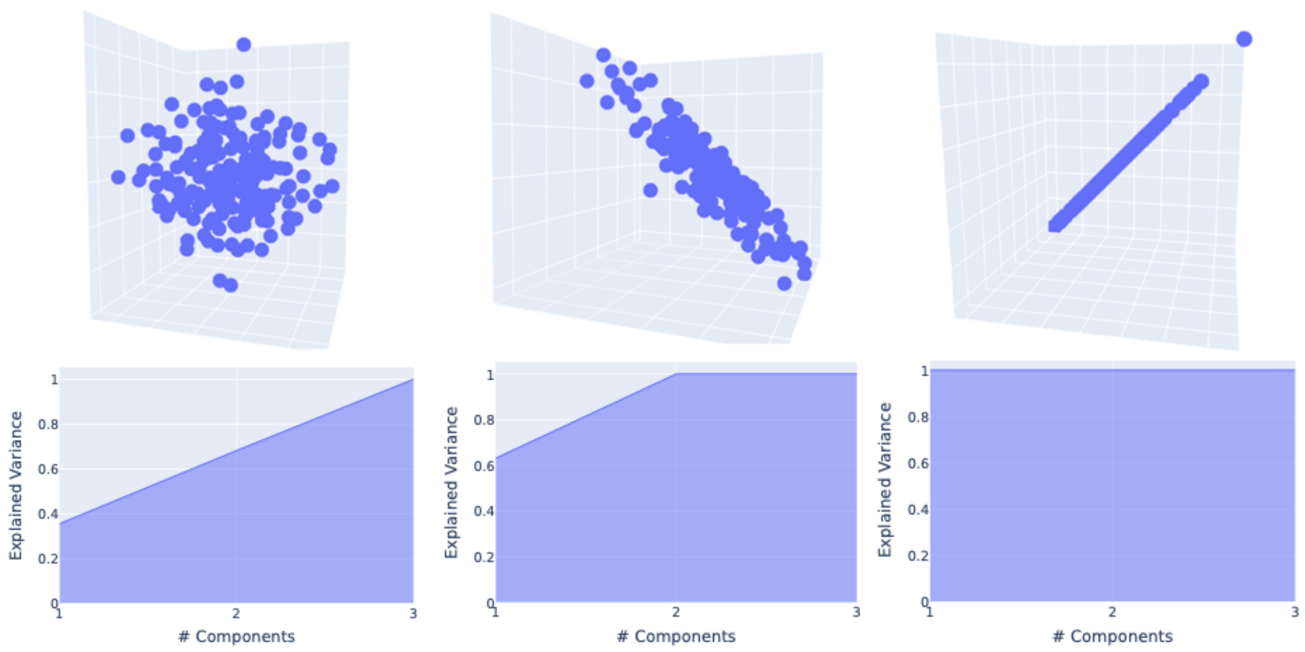

Figure 8. (Top) Samples drawn from a normal Gaussian distribution of 3D (left) 2D (middle) and 1D (right) support. (Buttom) The corresponding variance explained by the first principal components and colored AUC. The more diverse the data - the smaller the AUC.

F. Measuring the Abstraction of Representations by Centered Kernel Alignment (CKA)

Here we explain the way CKA is utilized for measuring the level of abstraction of the representations learnt by the pre-trained model. This measure is considered as a factor that implies on the transferablity of the model by [37, 42] and in Section 6 and Appendix H.

We follow the exact calculation of [42] and provide here the details for completeness. Linear CKA provides a way to measure similarity of neural network representations that is invariant to rotation and isotropic scaling in representation space [17, 43, 72] . Unlike other ways of measuring representational similarity between neural networks, linear CKA can identify architectural correspondences between layers of networks trained from different initializations [43]. Given two matrices $X \in \mathbb{R}^{n \times p_1}$ and $Y \in \mathbb{R}^{n \times p_2}$ containing activations to the same n examples, linear CKA computes the cosine similarity between the reshaped $n \times n$ covariance matrices between examples:

$$CKA_{linear}(X, Y) = \frac{vec(X^TX) \cdot vec(Y^TY)}{||X^TX||_F||Y^TY||_F} \tag{3}$$

We measured CKA between all possible pairings of ResNet resolution stages of all the models in Table 6. To reduce memory requirements, we used minibatch CKA [60] with minibatches of size 600 and processed the ImageNet validation set for 3 epochs.

Figure 9 shows the similarity between different stages of the same model in terms of the CKA for several different pretraining procedures. The numbers in the titles are the overall CKA score reported in Table 6, calculated as the average of the off-diagonal values.

G. Intra-class Variance and Class Separation

Supervised learning models learn feature representations by objectives that also increase the inter-class separation. Related However, although might be harmful for in-domain performance, [37] argued that increasing the intra-class variation was beneficial for learning rich feature representations in transfer learning. This measure is considered as a factor that implies on the transferablity of the model by [37, 42] and in Section 6 and Appendix H.

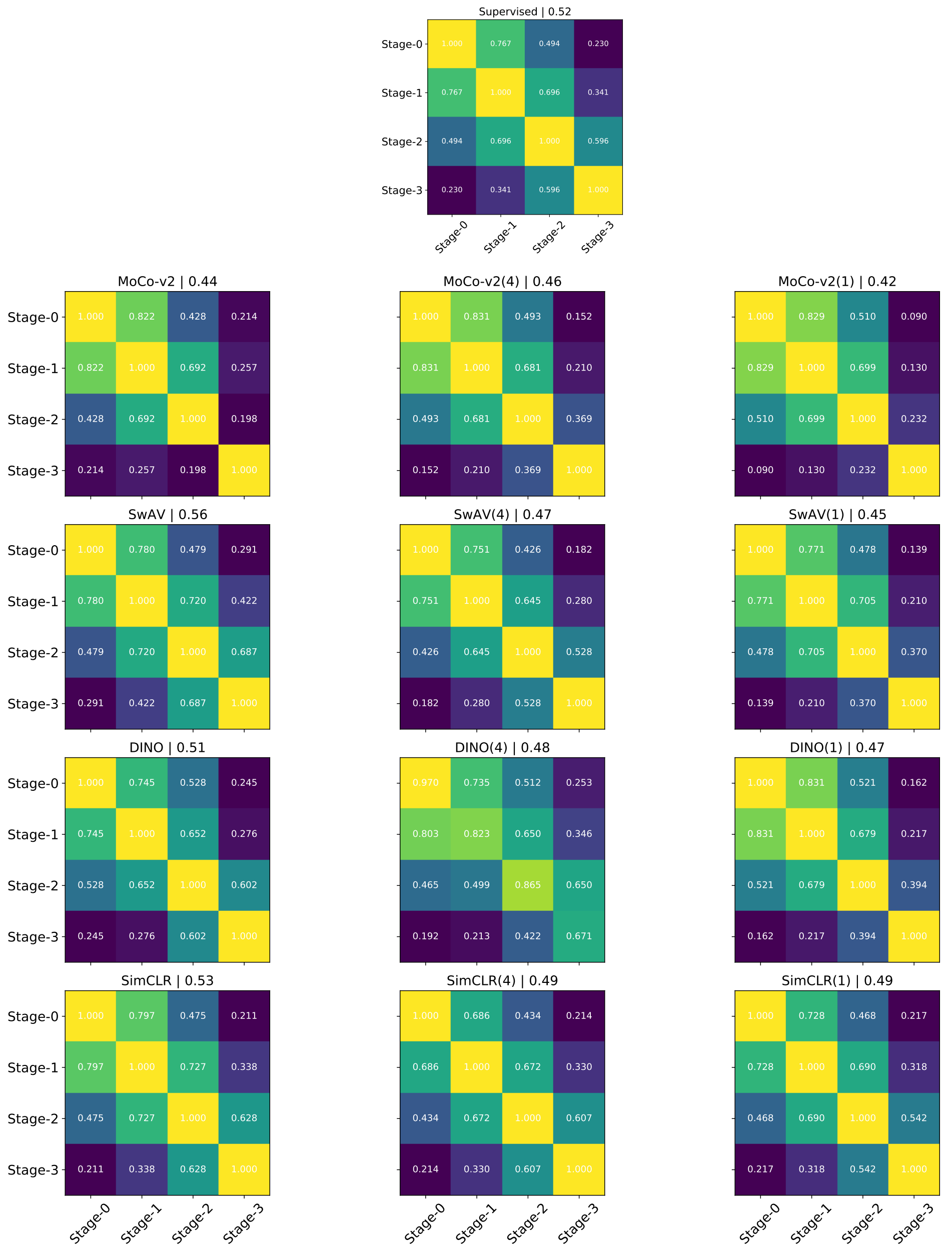

Figure 9. Similarity between different stages of the same model in terms of the CKA for several different pretraining procedures. The label injection control cycle appears in the parentheses. The corresponding CKA score appears in the titles.

The intra-class variation and inter-class separation are computed as follows [42]:

$$V_{intra} = \sum_{k=1}^{K} \sum_{m=1}^{N_k} \sum_{n=1}^{N_k} \frac{1 - cosine(x_{k,m}, x_{k,n})}{K N_k^2} \tag{4}$$

$$S_{inter} = \sum_{j=1}^{K} \sum_{k=1}^{K} \sum_{m=1}^{N_j} \sum_{n=1}^{N_k} \frac{1 - cosine(x_{k,m}, x_{k,n})}{K^2 N_k N_j} \tag{5}$$

where $x_{k,m}$ is the embedding of example m in class $k \in \{1, \ldots, K\}$ and N_k is the number of examples in class k. Those metrics for all of the pretrained CNN models are listed in Table 6.

G.1. The Mean Silhouette Coefficient

(MSC) [69] is an appropriate metric for quantifying class separation in the embedding space. This measure is considered as a factor that implies on the transferablity of the model in Section 6 and Appendix H.

For each embedding vector $x_{k,m}$ of example m in class $k \in \{1, \ldots, K\}$ and N_k, the number of examples in class k, the Silhouette coefficient SC_m is the relationship between the intra-class distances v_m and the nearest class distances s_m as follows:

$$SC_m = \frac{s_m - v_m}{\max(s_m - v_m)} \tag{6}$$

$$v_m = \sum_{n=1}^{N_k} \frac{1 - cosine(x_{k,m}, x_{k,n})}{N_k - 1} \tag{7}$$

$$s_m = \min_{j \in \{1,\ldots,K\} \setminus \{k\}} \sum_{n=1}^{N_j} \frac{1 - cosine(x_{k,m}, x_{k,n})}{N_j} \tag{8}$$

Since $|s - v| \leq \max(s, v)$, the mean Silhouette coefficient $MSC = \frac{\sum_{k=1}^{K} \sum_{m=1}^{N_k} SC_m}{\sum_{k=1}^{K} N_k}$ is bounded between -1 to 1. Those values for all of the pretrained CNN models are listed in Table 6. There is a positive correlation between MSC and Imagenet accuracy, indeed validating that representations with higher class separation obtain higher accuracy on the upstream task, and thus class seperation by its own does not add much information over Imagenet accuracy.

H. Quantifying Feature Importance by Gradient Boosting Decision Trees (XGBoost)

In section 6 we use gradient boosting decision trees for quantifying the importance of different factors on the transferability. A benefit of using gradient boosting [12] for solving a regression problem is that after the boosted trees are constructed, it is relatively straightforward to retrieve importance scores for each input feature. Generally, importance provides a score that indicates how useful or valuable each feature was in the construction of the boosted decision trees within the model. The more a feature is used to make key decisions with decision trees, the higher its relative importance. This importance is calculated explicitly for each feature, allowing features to be ranked and compared to each other. Importance is calculated for a single decision tree by the amount that each feature split point improves the performance measure, weighted by the number of observations the node is responsible for, see [26] for more details. The feature importance scores are then averaged across all of the the decision trees within the model. To this end, XGBoost [12], with the default *Scikit-Learn* parameters of 100 trees of maximal depth of 6, is fed with the set of inspected features {Imagenet accuracy, feature diversity, CKA, MSC, intra-class variation, inter-class separation} together with the corresponding transfer learning scores for each of the 40 pretrained models forming our dataset in Table 6. The derived feature importance is presented in Figure 1. It again clearly shows that most of the importance is attributed to Imagenet accuracy together with filter diversity, as the rest of the features, previously considered important [37,42], are overshadowed by the former. Figure 10 presents the importance of all the examined factors multiplied by Imagenet accuracy. The significant importance attributed to CIS shows that any of the alternative suggested factors cannot replace the role of Filter Diversity in CIS.

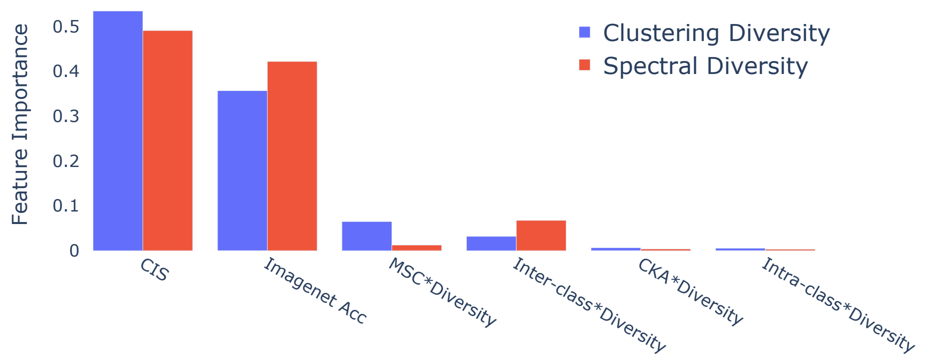

Figure 10. The relative importance of different factors **multiplied by Imagenet accuracy** are quantified by the popular feature importance derived from XGBoost. Most of the importance is attributed to the Calibrated Imagenet Score (CIS).

Pretrain	CIS (Clustering)	CIS (Spectral)	ImNet	Inter	CKA	Intra	MSC
Supervised	51.40	19.12	79.40	0.422	63.81	0.510	0.09
SupCon [40]	51.67	20.71	77.33	0.276	57.84	0.275	0.14
CE+SelfSupCon [37]	50.25	19.34	76.380	0.296	59.625	0.322	0.137
Moco-V2 ($\mathcal{T}$ = 1)	51.78	20.36	78.73	0.276	56.13	0.278	0.18
Moco-V2 ($\mathcal{T}$ = 2)	50.52	19.53	77.67	0.273	57.27	0.270	0.16
Moco-V2 ($\mathcal{T}$ = 3)	49.59	18.99	77.01	0.269	58.25	0.266	0.14
Moco-V2 ($\mathcal{T}$ = 4)	48.95	18.54	76.04	0.266	59.19	0.264	0.13
Moco-V2 ($\mathcal{T}$ = 5)	48.26	18.24	75.67	0.263	59.96	0.262	0.12
Moco-V2 ($\mathcal{T}$ = 10)	46.49	17.22	73.70	0.257	61.85	0.258	0.08
Moco-V2 ($\mathcal{T}$ = 50)	43.62	16.01	69.52	0.271	63.40	0.268	0.03
Moco-V2 ($\mathcal{T}$ = 100)	42.72	15.96	67.83	0.290	63.91	0.279	0.00
Moco-V2	44.17	17.62	67.70	0.377	57.63	0.419	0.04
SwAV ($\mathcal{T}$ = 1)	52.51	21.05	79.17	0.321	58.42	0.356	0.12
SwAV ($\mathcal{T}$ = 2)	53.38	22.21	78.96	0.294	59.20	0.335	0.10
SwAV ($\mathcal{T}$ = 3)	53.78	22.64	78.59	0.277	59.80	0.317	0.08
SwAV ($\mathcal{T}$ = 4)	53.63	22.61	78.08	0.269	60.15	0.307	0.06
SwAV ($\mathcal{T}$ = 5)	53.33	22.52	77.58	0.265	60.34	0.302	0.06
SwAV ($\mathcal{T}$ = 10)	52.46	22.16	76.12	0.260	60.71	0.296	0.04
SwAV ($\mathcal{T}$ = 50)	50.47	21.35	73.20	0.265	61.45	0.308	0.01
SwAV ($\mathcal{T}$ = 100)	49.65	21.00	72.00	0.267	61.52	0.629	0.00
SwAV	49.67	21.01	72.00	0.259	67.23	0.367	-0.00
DINO ($\mathcal{T}$ = 1)	52.14	20.89	77.62	0.301	60.05	0.692	0.11
DINO ($\mathcal{T}$ = 2)	52.99	21.81	77.67	0.280	60.92	0.656	0.08
DINO ($\mathcal{T}$ = 3)	53.40	22.27	77.60	0.266	61.14	0.625	0.07
DINO ($\mathcal{T}$ = 4)	53.47	22.34	77.46	0.260	61.34	0.608	0.06
DINO ($\mathcal{T}$ = 5)	53.27	22.28	77.08	0.257	61.32	0.600	0.05
DINO ($\mathcal{T}$ = 10)	52.96	22.19	76.53	0.254	61.66	0.594	0.03
DINO ($\mathcal{T}$ = 50)	52.73	22.11	76.11	0.260	62.69	0.625	0.01
DINO ($\mathcal{T}$ = 100)	52.52	22.03	75.82	0.263	63.22	0.643	0.00
DINO	51.93	21.80	74.98	0.274	63.10	0.781	0.00
SimCLR ($\mathcal{T}$ = 1)	51.03	19.24	78.07	0.404	62.04	0.958	0.08
SimCLR ($\mathcal{T}$ = 2)	50.50	19.57	77.07	0.399	61.89	0.944	0.06
SimCLR ($\mathcal{T}$ = 3)	49.62	19.14	75.81	0.394	61.69	0.925	0.05
SimCLR ($\mathcal{T}$ = 4)	49.00	18.90	74.95	0.391	61.79	0.914	0.04
SimCLR ($\mathcal{T}$ = 5)	48.42	18.73	74.30	0.390	61.87	0.909	0.03
SimCLR ($\mathcal{T}$ = 10)	47.69	18.42	73.09	0.389	61.88	0.906	0.02
SimCLR ($\mathcal{T}$ = 50)	45.54	17.62	69.90	0.375	62.16	0.979	0.00
SimCLR ($\mathcal{T}$ = 100)	45.24	17.50	69.43	0.371	62.30	0.977	0.00
SimCLR	44.37	17.16	68.09	0.365	64.69	0.973	-0.00

Table 6. Full metrics for all models used in computing feature importance for CNN models, presented in Figure 1 (Right)

I. Fully Connected Initialization

A standard practice for fully-connected initializion when fine tuning a pretrained backbone on a new task is either linear probe or random initializaiton. Linear probe uses a fixed backbone as feature extractor and the classification head is trained with gradient descent. We find that it is undesirable to start training with a randomly initialized fully connected layer on top of a pretrained backbone when fine-tuning on the downstream tasks, as it requires the linear probing to have multiple epochs and many hyper parameters. Instead we look at the fully connected as class centroids that approximate to nearest neighbor classification. Hence, we initialize the i^{th} column of the fully connected to be the mean class representation for the i^{th} class and such that $W^i = \frac{1}{|C_i|}\sum_{x_i \in C_i} f(x_i)$.

J. Linear Probing

Despite the focus of this work is on the fine-tuning of the pretrained models, in this section we show that the conclusions presented throughout the paper are also valid for linear probing over the CNN models, when the pretrained backbones are fixed and a logistic regression classifier is trained on the downstream datasets, using features extracted from the penultimate layer their as inputs.

J.1. Linear Probing by Logistic Regression

In this section we present the technical details involved in performing the linear probing by solving a fitting a logistic regression. Although we follow the protocol proposed by [44], we give here the details for completeness. For each dataset, we extracted features from the penultimate layer of the network. We trained a multinomial logistic regression classifier implemented by *Scikit-Learn* with the default settings, using L-BFGS, with an L2 regularization parameter applied to the sum of the per-example losses, selected from a range of 45 logarithmically spaced values from 10^{-6} to 10^{5} on the validation set. Since the optimization problem is convex, we used the solution at the previous point along the regularization path as a warm start for the next point, which greatly accelerated the search. For these experiments, we did not perform data augmentation or scale aggregation, and we used the entire image, rather than cropping the central 87.5% as is common for testing on ImageNet.

J.2. The Same Conclusions Hold For Linear Probing

Here we show the corresponding results of Figure 1 and Figure 3 in Figure 11, Figure 12 and Figure 13 respectively.

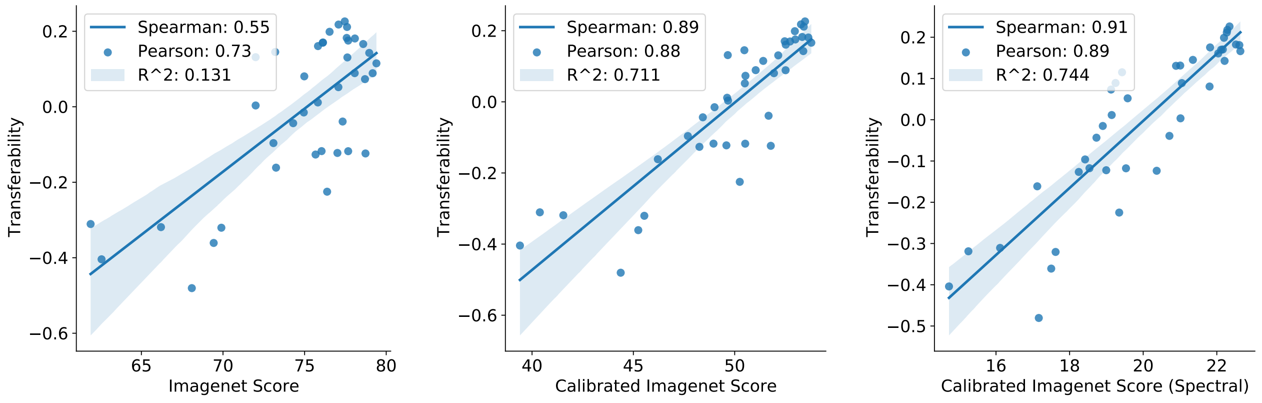

Figure 11. Linear probing transferability vs Imagent Score (Left), the Clustering Filter Diversity based CIS (Middle) and the Spectral Filter Diversity based CIS (Right) for 40 models that were pre-trained with supervised learning, self-supervised learning or their combination. The Calibrated Imagenet Score correlates with transferability significantly better. CIS correlates with transferability significantly better than imagenet score in both cases.

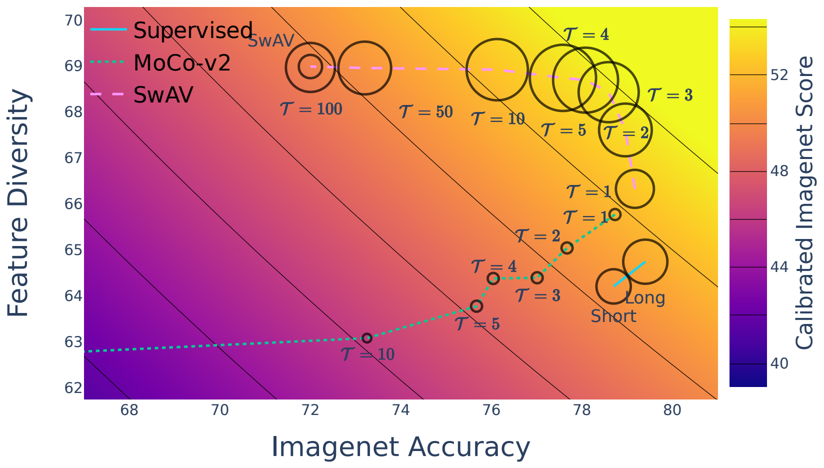

Figure 12. Circle size corresponds to the linear probing transferability averaged over 14 downstream tasks, as a function of Imagenet top-1 accuracy (x axis) and filter diversity (y axis). Results shown for 3 different training methods (see legend). The background colors and curves show the Calibrated Imagenet Score. Evidently, models with both high Imagenet accuracy and high Filter Diversity, that together result in high Calibrated Imagenet Score (in yellow), transfer better (larger circles).

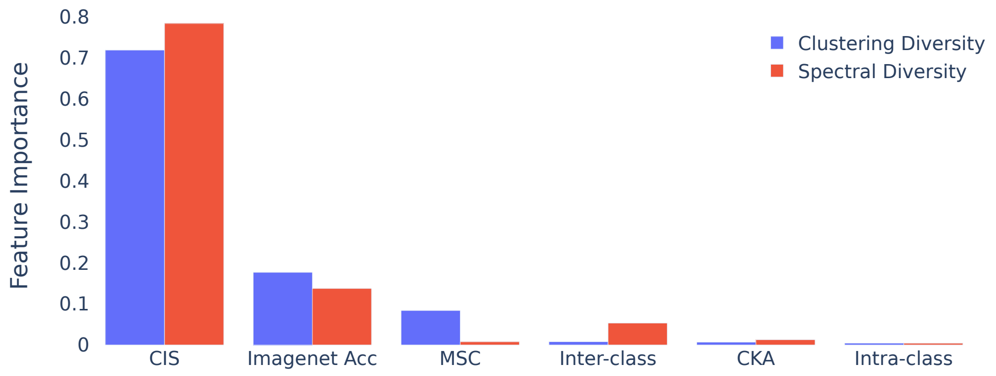

Figure 13. The relative importance for linear probing transferability of different factors are quantified by the popular feature importance derived from XGBoost. Most of the importance is attributed to the Calibrated Imagenet Score (CIS).

K. Transfer Learning Results - Full Tables

K.1. Finetune Tables

In this section we present the full transfer learning results in Tables 7 and 8 for CNN and ViT models respectively.

K.2. Linear Probing CNN Tables

In this section we present the full transfer learning results for linear probing in Tables 9 for CNN.

Pretrain	ImNet	Caltech	CIFAR10	CIFAR100	CUB	DTD	Aircraft	Food	Indoor	Birds	Flowers	Pets	Cars	Dogs	SUN	Transfer
Supervised	78.7	86.9	97.6	86.0	85.9	69.3	82.0	85.3	81.3	74.9	99.1	92.4	94.4	82.9	65.3	0.012
SupCon [40]	77.3	86.3	97.6	85.7	85.0	69.8	84.1	86.1	81.4	73.1	99.0	91.9	94.7	83.0	65.2	0.007
CE+SelfSupCon [37]	76.4	86.3	97.6	85.8	85.9	69.5	83.3	86.3	80.9	74.1	98.7	92.4	94.8	85.3	64.4	0.003
MoCo-v2 ($\mathcal{T} = 1$)	78.7	87.3	97.6	86.3	85.9	70.0	83.0	86.1	82.2	73.9	99.2	92.8	94.4	84.4	65.1	0.054
MoCo-v2 ($\mathcal{T} = 2$)	77.7	87.0	97.6	86.4	85.7	70.7	82.9	86.1	81.9	73.8	99.1	92.4	94.4	82.8	65.3	0.026
MoCo-v2 ($\mathcal{T} = 3$)	77.0	86.8	97.6	86.1	85.5	70.4	83.2	86.1	81.6	73.4	98.9	92.0	94.3	82.0	64.8	-0.001
MoCo-v2 ($\mathcal{T} = 4$)	76.0	86.4	97.6	86.1	85.7	69.7	82.8	86.1	82.3	73.3	98.9	92.0	94.3	81.5	64.8	-0.010
MoCo-v2 ($\mathcal{T} = 5$)	75.7	86.2	97.7	85.9	85.2	70.7	83.6	86.3	82.5	73.3	98.9	91.7	94.5	80.7	64.6	-0.006
MoCo-v2 ($\mathcal{T} = 10$)	73.2	85.3	97.6	85.7	84.9	69.5	82.7	86.1	81.0	72.3	98.9	91.2	94.3	78.9	63.9	-0.056
MoCo-v2 ($\mathcal{T} = 50$)	66.2	82.8	97.3	84.4	83.7	67.1	82.9	85.7	79.1	69.6	98.9	89.2	94.3	75.7	61.3	-0.152
MoCo-v2 ($\mathcal{T} = 100$)	62.6	81.6	97.1	83.8	82.6	66.9	83.4	85.8	77.6	68.2	98.9	88.2	94.2	74.9	60.0	-0.202
MoCo-v2	61.9	78.1	96.7	82.0	79.4	66.3	80.1	85.4	75.6	61.1	98.5	86.8	93.5	73.4	56.3	-0.352
SwAV ($\mathcal{T} = 1$)	79.2	87.5	97.7	86.5	86.5	71.3	83.4	86.8	82.3	75.8	99.0	92.5	94.6	83.6	66.4	0.062
SwAV ($\mathcal{T} = 2$)	79.0	87.8	97.7	86.9	86.9	72.3	83.7	87.2	83.4	76.0	99.3	92.4	94.7	82.7	67.1	0.104
SwAV ($\mathcal{T} = 3$)	78.6	88.2	97.8	87.0	86.8	72.4	83.3	87.4	84.1	76.4	99.4	92.5	94.7	82.4	67.6	0.125
SwAV ($\mathcal{T} = 4$)	78.1	88.4	97.8	87.1	86.6	72.9	82.8	87.6	84.4	76.1	99.3	91.9	94.6	82.1	67.8	0.118
SwAV ($\mathcal{T} = 5$)	77.6	88.6	97.9	87.2	86.4	73.0	83.2	87.6	84.1	76.1	99.3	91.6	94.6	81.8	67.9	0.118
SwAV ($\mathcal{T} = 10$)	76.1	88.4	97.9	87.2	85.8	72.5	82.4	87.6	84.8	75.6	99.4	91.3	94.3	81.4	68.0	0.103
SwAV ($\mathcal{T} = 50$)	73.2	88.0	97.8	86.8	84.9	71.7	82.7	87.6	83.2	75.4	99.0	90.7	93.9	80.6	67.7	0.027
SwAV ($\mathcal{T} = 100$)	72.0	87.8	97.9	86.6	85.0	71.7	82.9	87.5	83.6	75.2	99.0	90.4	93.8	80.5	67.6	0.028
SwAV	72.0	87.0	97.8	86.6	84.3	72.1	82.3	87.4	83.1	75.1	98.9	90.3	93.6	80.6	67.8	0.005
DINO ($\mathcal{T} = 1$)	77.6	87.4	97.7	86.7	86.3	71.1	83.0	87.1	82.7	76.2	99.4	92.5	94.7	82.9	66.2	0.095
DINO ($\mathcal{T} = 2$)	77.7	87.4	97.7	86.9	86.7	72.1	82.8	87.3	83.1	76.3	99.4	92.1	94.7	82.4	67.0	0.106
DINO ($\mathcal{T} = 3$)	77.6	88.1	97.8	87.1	86.8	71.5	82.4	87.6	83.9	76.7	99.3	92.3	94.5	82.3	67.5	0.103
DINO ($\mathcal{T} = 4$)	77.5	88.2	97.8	87.5	86.5	72.1	82.8	87.6	84.1	76.5	99.4	92.1	94.7	82.1	67.6	0.126
DINO ($\mathcal{T} = 5$)	77.1	88.3	97.9	87.3	86.0	71.9	82.6	87.7	84.7	76.4	99.3	92.1	94.7	81.7	67.7	0.116
DINO ($\mathcal{T} = 10$)	76.5	88.2	97.9	87.4	85.6	71.3	82.3	87.7	84.0	75.9	99.3	91.5	94.4	80.9	67.7	0.088
DINO ($\mathcal{T} = 50$)	76.1	87.8	98.0	87.2	84.8	72.1	82.5	87.5	82.7	75.4	99.1	90.4	94.0	80.2	67.6	0.036
DINO ($\mathcal{T} = 100$)	75.8	87.7	97.8	86.8	84.7	71.7	82.1	87.6	82.4	75.4	99.0	90.1	93.9	80.0	67.4	0.010
DINO	75.0	87.2	97.8	86.9	83.7	72.1	80.6	87.5	83.2	74.5	98.7	89.6	93.8	80.1	67.6	-0.024
SimCLR ($\mathcal{T} = 1$)	78.1	86.3	97.6	86.3	85.6	70.0	82.8	85.9	80.7	72.8	99.1	91.4	94.5	80.8	65.5	-0.013
SimCLR ($\mathcal{T} = 2$)	77.1	86.3	97.8	86.4	84.6	69.3	82.1	85.6	80.4	70.7	98.9	90.5	94.2	79.7	64.9	-0.058
SimCLR ($\mathcal{T} = 3$)	75.8	86.6	97.9	86.5	83.9	69.8	81.7	85.1	81.0	69.2	98.7	90.1	93.7	78.5	65.1	-0.087
SimCLR ($\mathcal{T} = 4$)	75.0	86.6	97.7	86.7	82.7	69.5	81.1	84.9	79.3	68.0	98.7	89.5	93.5	77.9	64.9	-0.120
SimCLR ($\mathcal{T} = 5$)	74.3	86.6	97.9	86.7	82.4	69.3	80.8	84.8	79.3	67.3	98.5	89.5	93.4	77.3	64.6	-0.138
SimCLR ($\mathcal{T} = 10$)	73.1	86.3	98.0	86.3	81.1	68.7	79.2	84.5	78.4	65.5	97.8	88.6	92.7	75.9	64.1	-0.214
SimCLR ($\mathcal{T} = 50$)	69.9	85.3	97.9	86.0	78.8	67.0	77.4	84.0	75.0	63.1	97.4	86.3	92.0	73.9	63.0	-0.321
SimCLR ($\mathcal{T} = 100$)	69.4	85.3	97.8	86.0	78.7	66.6	77.3	84.0	74.9	62.9	97.4	86.2	91.9	73.7	63.0	-0.327
SimCLR	68.1	85.4	97.9	86.3	78.3	65.9	77.2	84.0	75.4	62.6	97.6	86.6	91.6	73.8	62.9	-0.31.8

Table 7. Performance of different **CNN** models (an extension of Table 4), including different levels of label injected models) fine-tuned on the downstream datasets in terms of top-1 accuracy (%) (averaged over 3 runs) and the overall transferability score. The models are grouped by the underlying base SSL method.

Pretrain	ImNet	Caltech	CIFAR10	CIFAR100	CUB	DTD	Aircraft	Food	Indoor	Birds	Flowers	Pets	Cars	Dogs	SUN	Transfer
Supervised	81.0	90.9	98.6	89.6	82.3	70.8	60.8	87.9	83.2	79.6	91.3	93.8	85.4	91.5	68.1	-0.101
DINO ($\mathcal{T} = 1$)	83.2	93.1	99.0	91.2	84.7	74.6	72.1	89.9	86.3	84.9	94.7	94.3	89.4	90.5	71.5	0.148
DINO ($\mathcal{T} = 2$)	83.2	93.3	98.7	91.1	84.7	75.7	71.3	89.9	86.7	85.1	95.2	94.6	89.1	89.4	71.3	0.140
DINO ($\mathcal{T} = 3$)	82.8	93.1	98.9	90.7	84.7	75.0	71.7	89.8	86.1	84.9	95.1	94.7	89.1	88.6	71.3	0.133
DINO ($\mathcal{T} = 4$)	82.3	92.8	98.8	91.0	84.7	75.2	71.2	90.0	85.8	85.2	95.4	94.7	89.1	88.3	71.5	0.131
DINO ($\mathcal{T} = 5$)	82.1	92.5	98.9	90.9	84.5	74.2	72.0	89.6	85.2	85.0	95.3	94.6	89.3	88.0	70.9	0.113
DINO ($\mathcal{T} = 10$)	80.9	92.4	98.8	90.8	84.5	74.4	71.7	89.7	85.3	84.6	95.1	94.3	88.9	87.0	70.8	0.089
DINO ($\mathcal{T} = 50$)	78.1	92.0	98.6	90.4	83.3	73.5	68.6	89.5	84.9	84.2	94.9	93.7	87.4	85.3	69.7	0.012
DINO ($\mathcal{T} = 100$)	77.0	91.7	98.6	90.0	83.3	72.9	68.2	89.5	85.5	84.2	94.2	93.8	86.8	84.6	70.0	-0.015
DINO	78.2	91.1	98.5	90.0	79.5	72.1	53.0	89.3	84.5	83.9	90.1	92.2	80.9	85.2	69.3	-0.187
MAE ($\mathcal{T} = 1$)	83.4	93.0	98.8	90.6	84.3	73.7	73.1	90.6	85.4	86.2	94.4	94.8	89.8	89.5	71.1	0.131
MAE ($\mathcal{T} = 2$)	82.7	93.1	98.8	90.3	84.5	74.2	72.3	90.6	85.3	86.1	94.4	94.8	89.4	88.9	71.2	0.122
MAE ($\mathcal{T} = 3$)	81.8	92.7	98.8	90.4	84.0	74.1	71.6	90.2	85.0	85.5	94.7	94.8	89.7	88.1	71.1	0.107
MAE ($\mathcal{T} = 4$)	81.3	92.2	98.8	90.1	84.5	73.2	72.4	90.2	85.0	85.6	94.6	94.6	89.2	88.2	71.0	0.095
MAE ($\mathcal{T} = 5$)	80.7	92.5	98.8	89.9	83.6	73.2	71.8	90.1	84.2	85.5	94.7	94.4	89.2	87.6	70.8	0.078
MAE ($\mathcal{T} = 10$)	79.0	91.8	98.7	89.5	83.6	72.7	71.1	89.7	84.4	85.0	93.8	94.4	88.5	86.9	70.5	0.029
MAE ($\mathcal{T} = 50$)	74.2	90.4	98.6	88.6	81.8	71.7	67.8	89.4	83.7	83.5	92.6	93.6	87.3	85.2	69.5	-0.080
MAE ($\mathcal{T} = 100$)	71.9	90.2	98.5	88.0	80.9	70.7	66.3	89.2	83.1	83.5	92.3	92.8	87.1	84.7	69.1	-0.122
MAE	68.0	89.2	98.1	86.9	75.4	68.5	53.8	88.8	82.2	81.2	70.6	91.2	79.6	83.2	67.6	-0.425

Table 8. Performance of different **ViT** models (an extension of Table 5) fine-tuned on the downstream datasets in terms of top-1 accuracy (%) and the overall transferability score. The models are grouped by the underlying base SSL method.

L. Filter Diversity for Vision Transformers

Similarly to Figure 3 and Figure 12, Figure 14 (presenting Table 8) shows how the control label injection (CLI) can start off from different SSL pre-trained ViT models of both higher (MAE) and lower (DINO) filter diversity and generate ViT models of different levels of Imagenet accuracy and filter diversity for different control cycle values. Those generated ViT models allow us to make observations about the connection between Imagenet Score and Filter Diversity, associated with Multi-head Self-attention (MSA) layers, to the transferability through the Calibrated Imagenet Score. Similarly to the behaviour of CNNs, the trajectory for every origin SSL model traverses the Calibrated Imagenet Score contour lines towards

Pretrain	ImNet	Aircraft	Birds	CIFAR10	CIFAR100	CUB	Caltech	Cars	DTD	Dogs	Flowers	Food	Indoor	Pets	SUN	Transfer
Supervised	78.7	46.4	60.9	93.0	77.1	71.5	89.1	67.4	69.5	90.4	86.5	70.0	78.7	93.0	63.1	7.3
SupCon [40]	77.3	50.9	56.6	94.9	79.2	69.0	88.5	72.6	70.2	90.5	89.1	68.6	79.3	92.5	63.6	12.6
CE + SelfSupCon [37]	77.3	40.2	52.8	93.3	76.5	63.0	87.3	58.1	67.6	94.0	85.6	67.4	76.6	92.6	61.3	-3.9
MoCo-v2 ($\mathcal{T} = 1$)	78.7	35.8	55.9	92.4	74.6	65.6	87.3	52.7	67.8	91.6	80.6	65.4	76.3	93.1	60.5	-12.4
MoCo-v2 ($\mathcal{T} = 2$)	77.7	38.6	56.4	92.5	74.6	65.6	87.7	55.0	67.4	89.7	81.5	66.2	76.7	92.6	60.4	-11.8
MoCo-v2 ($\mathcal{T} = 3$)	77.0	39.9	57.4	92.4	75.1	66.3	87.2	55.7	68.0	88.2	82.3	66.3	76.0	92.1	60.3	-12.2
MoCo-v2 ($\mathcal{T} = 4$)	76.0	41.3	57.3	92.2	74.4	66.2	87.0	57.0	66.9	87.5	83.5	67.2	76.9	91.9	60.8	-11.8
MoCo-v2 ($\mathcal{T} = 5$)	75.7	41.8	56.9	92.0	75.0	66.0	86.9	56.9	66.9	86.3	83.9	66.8	75.8	92.5	59.9	-12.7
MoCo-v2 ($\mathcal{T} = 10$)	73.2	42.0	55.6	92.4	74.1	65.8	85.9	58.0	66.3	83.8	84.0	66.4	75.4	91.4	59.8	-16.1
MoCo-v2 ($\mathcal{T} = 50$)	66.2	41.5	49.6	91.5	72.5	61.5	82.7	56.3	65.1	76.6	82.5	65.0	72.5	88.9	56.4	-31.9
MoCo-v2 ($\mathcal{T} = 100$)	62.6	39.8	46.0	91.0	71.8	59.1	81.7	55.4	64.5	73.5	81.7	64.1	69.8	87.1	54.5	-40.4
MoCo-v2	61.9	43.9	38.9	93.4	76.4	53.8	83.5	59.3	69.7	68.0	85.3	68.5	76.0	84.6	60.6	-31.1
SwAV ($\mathcal{T} = 1$)	79.2	44.0	62.8	93.3	77.4	71.2	89.2	64.2	70.7	91.1	86.6	70.2	80.1	93.3	63.5	8.9
SwAV ($\mathcal{T} = 2$)	79.0	47.7	64.0	93.6	78.4	72.8	89.3	66.7	71.4	90.0	88.6	71.7	80.1	93.6	64.6	14.3
SwAV ($\mathcal{T} = 3$)	78.6	50.9	65.0	93.0	78.3	73.4	89.8	67.8	72.8	88.4	89.6	72.9	81.3	93.2	65.8	16.6
SwAV ($\mathcal{T} = 4$)	78.1	53.4	65.7	93.1	78.8	73.2	89.8	69.8	72.2	87.0	90.4	73.2	81.6	93.1	65.8	18.1
SwAV ($\mathcal{T} = 5$)	77.6	54.2	65.4	93.6	79.1	73.0	89.4	70.3	72.7	85.9	90.3	73.8	81.9	92.9	65.6	18.2
SwAV ($\mathcal{T} = 10$)	76.1	55.2	65.0	93.6	78.8	72.2	88.9	71.9	72.6	83.9	90.7	74.1	82.0	92.3	65.8	17.0
SwAV ($\mathcal{T} = 50$)	73.2	56.0	63.5	93.6	79.2	71.3	88.3	73.3	72.7	80.7	91.2	74.5	82.0	91.1	65.6	14.5
SwAV ($\mathcal{T} = 100$)	72.0	56.3	62.1	93.5	79.1	70.9	88.2	73.6	72.8	79.9	91.1	74.3	82.0	90.7	65.7	13.1
SwAV	72.0	52.0	53.3	93.2	77.8	66.7	86.5	71.0	71.4	76.4	90.6	73.2	81.6	88.9	65.1	0.3
DINO ($\mathcal{T} = 1$)	77.6	46.0	63.4	93.5	78.3	73.2	89.2	66.2	71.0	90.9	87.0	71.2	80.9	93.8	64.2	13.1
DINO ($\mathcal{T} = 2$)	77.7	49.9	64.8	93.7	78.4	73.5	89.1	68.3	72.6	89.3	89.5	72.9	80.8	93.7	65.1	17.5
DINO ($\mathcal{T} = 3$)	77.6	52.0	65.9	94.0	79.1	73.6	89.7	69.8	72.8	88.3	91.0	74.2	81.6	93.2	65.9	21.1
DINO ($\mathcal{T} = 4$)	77.5	53.8	66.0	93.8	79.5	74.3	89.7	70.5	73.0	86.9	91.8	74.8	82.2	93.3	66.0	22.6
DINO ($\mathcal{T} = 5$)	77.1	54.5	66.0	93.8	79.4	74.1	89.5	71.3	72.5	85.8	91.9	75.2	81.8	93.0	65.9	21.8
DINO ($\mathcal{T} = 10$)	76.5	55.6	65.6	93.8	79.5	73.5	88.8	72.2	72.8	83.1	92.3	75.4	81.8	92.3	65.8	19.9
DINO ($\mathcal{T} = 50$)	76.1	56.9	63.8	93.7	79.3	72.1	88.2	74.2	73.1	79.8	92.6	75.6	82.2	90.9	65.8	17.0
DINO ($\mathcal{T} = 100$)	75.8	56.6	62.4	93.9	79.6	72.0	88.1	74.5	72.9	78.7	92.6	75.3	82.2	90.6	65.9	16.1
DINO	75.0	54.8	54.8	93.7	78.6	68.9	87.1	74.5	72.7	75.9	92.5	74.7	81.7	89.3	66.1	8.1
SimCLR ($\mathcal{T} = 1$)	78.1	47.8	61.1	93.8	77.7	71.3	88.7	65.8	70.9	88.9	87.5	70.5	80.2	92.9	64.0	8.9
SimCLR ($\mathcal{T} = 2$)	77.1	49.9	58.6	93.8	77.6	70.1	88.3	66.5	69.8	86.6	87.6	70.3	78.9	92.5	63.7	5.2
SimCLR ($\mathcal{T} = 3$)	75.8	50.0	56.9	93.7	77.9	68.9	88.1	65.9	69.8	84.2	87.7	69.8	77.8	91.7	63.2	1.1
SimCLR ($\mathcal{T} = 4$)	75.0	50.4	55.3	93.7	77.6	67.5	87.5	66.8	69.8	82.4	88.0	69.2	77.0	91.3	63.0	-1.5
SimCLR ($\mathcal{T} = 5$)	74.3	50.1	54.3	93.2	77.1	66.8	87.4	65.8	69.8	81.2	88.2	69.2	75.9	91.2	62.8	-4.3
SimCLR ($\mathcal{T} = 10$)	73.1	49.9	51.3	93.2	77.3	64.4	86.8	65.7	69.4	77.7	87.8	68.4	76.4	89.9	62.1	-9.6
SimCLR ($\mathcal{T} = 50$)	69.9	44.3	39.4	92.9	75.0	54.5	84.3	60.2	66.1	69.4	87.0	64.5	73.1	86.6	60.2	-32.0
SimCLR ($\mathcal{T} = 100$)	69.4	43.8	38.0	92.5	74.2	52.5	83.7	59.1	66.2	68.3	86.6	64.0	72.5	85.8	59.8	-36.1
SimCLR	68.1	43.4	35.3	89.1	69.0	50.5	82.4	56.2	65.4	65.4	85.2	62.2	72.4	83.8	58.2	-48.0

Table 9. Linear probing performance of different **CNN** models (an extension of Table 3), including different levels of label injected models) fit on the downstream datasets in terms of top-1 accuracy (%) and the overall transferability score. The models are grouped by the underlying base SSL method.

more transferable regions, as expressed by the size of the circles and the background color. Specifically, the upper trajectory, originated in MAE, represents a case where CLI maintains the filter diversity while improving Imagenet score as the main mean for improving the diversity. Another case is represented by the lower DINO originated trajectory, where at some point both Imagenet score and filter diversity improve together towards better transferability. Especially interesting is the two right most points of this DINO trajectory ($\mathcal{T} = 2$ and $\mathcal{T} = 1$), that share the very same Imagenet score, yet differ in the filter diversity resulting in the difference in transferability.

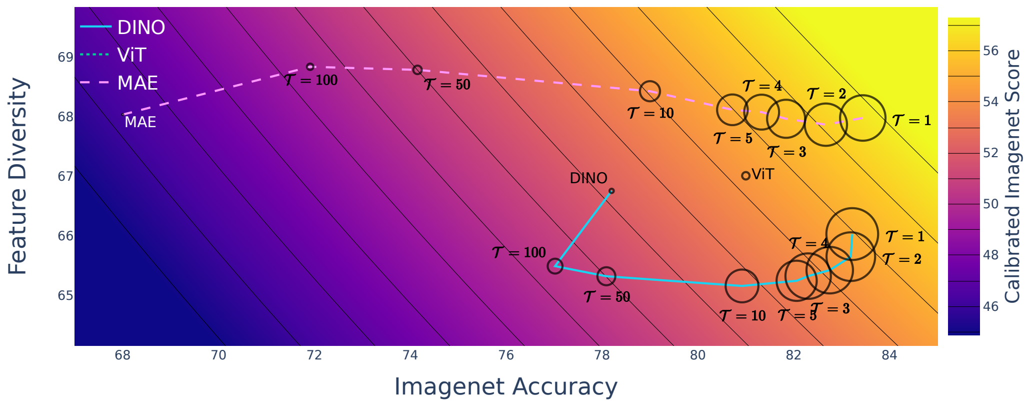

Figure 14. Circle size corresponds to the transferability of **finetuned ViT** models averaged over 14 downstream tasks, as a function of Imagenet top-1 accuracy (x axis) and filter diversity (y axis). Results shown for 3 different training methods (see legend). The background colors and curves show the Calibrated Imagenet Score. Evidently, models with both high Imagenet accuracy and high Filter Diversity, that together result in high Calibrated Imagenet Score (in yellow), transfer better (larger circles).