A Experimental setup.

A.1 Architectures and hyperparameters

In this section, we detail the model architectures examined in the experiments and list all hyperparameters used in the experiments.

VGG [60] In the main text use two types of VGG networks, namely VGG-19 and VGG-16. Both architectures consist of five stages, each consisting of a combination of convolutional layers with ReLU activation and max pooling layers. The VGG-19 has 19 layers, including 16 convolutional layers and three fully connected layers. The first two fully connected layers are followed by ReLU activation. On the other hand, VGG-16 has a total of 16 layers, including 13 convolutional layers and three fully connected layers. In additional experiments, we extend our analysis by VGG-11 and VGG-16. The base number of channels in consecutive stages for VGG architectures equals 64, 128, 256, 512, and 512.

ResNet [19] In experiments, we utilize two variants of the ResNet family of architectures, i.e., ResNet-18 and ResNet-34. ResNet-N is a five-staged network characterized by depth, with a total of N layers. The initial stage consists of a single convolutional layer – with kernel size 7×7 and 64 channels and ReLU activation, followed by max pooling 2×2, which reduces the spatial dimensions. The subsequent stages are composed of residual blocks. Each residual block typically contains two convolutional layers and introduces a shortcut connection that skips one or more layers. Each convolutional layer in the residual block is followed by batch normalization and ReLU activation. The remaining four stages in ResNet-18 and ResNet-34 architectures consist of 3x3 convolutions with the following number of channels: 64, 128, 256, and 512.

MLP [5] An MLP (Multi-Layer Perceptron) network is a feedforward neural network architecture type. It consists of multiple layers of artificial neurons – in our experiments, we consider MLPs with 6,8,10,12 layers with ReLU activations (except last layer, which has linear activation). In our experiments, the underlying architecture has 1024 neurons per layer.

In VGGs, MLPs, and ResNets without skips, we use the 98% threshold to estimate the tunnel for the plots. In the case of ResNets, we use the 95% threshold. In the case of ResNets, we report the results for the 'conv2' layers. Due to computational constraints, we randomly choose a subset of 8000 features to compute the numerical rank.

Hyperparameters Hyperparameters used for neural network training are presented in the leftmost Table A.1. Each column shows the values of the hyperparameters corresponding to a different architecture. The presented hyperparameters are recommended for the best performance of these models on the CIFAR-10 dataset [33]. However, in experiments focused on continual learning scenario (Section 4.2), we refrain from decaying the learning rate and shorten the network's training to 30 epochs to mimic the actual settings used in continual learning settings.

Hyperparameters used for training linear probes in our experiment are presented in the rightmost table. Linear probes were trained with Adam optimizer instead of SGD.

Parameter	VGG	ResNet	MLP
Learning rate (LR)	0.1	0.1	0.05
SGD momentum	0.9	0.9	0.0
Weight decay	10^{-4}	10^{-4}	0
Number of epochs	160	164	1000
Mini-batch size	128	128	128
LR-decay-milestones	80, 120	82, 123	-
LR-decay-gamma	0.1	0.1	0.0

Parameter	Value
Learning rate	0.001
Weight decay	0
Number of epochs	30
Mini-batch size	512

A.2 Datasets

In this article, we present the results of experiments conducted on following datasets: **CIFAR-10 [26]** CIFAR-10 is a widely used benchmark dataset in the field of computer vision. It consists of 60,000 color images in 10 different classes, with each class containing 6,000 images. The dataset is divided into 50,000 training images and 10,000 test images. The images in CIFAR-10 have a resolution of 32×32 pixels.

CIFAR-100 [26] CIFAR-100 is a dataset commonly used for image classification tasks in computer vision. It contains 60,000 color images, with 100 different classes, each containing 600 images. The dataset is split into 50,000 training images and 10,000 test images. The images in CIFAR-100 have a resolution of 32×32 pixels. Unlike CIFAR-10, CIFAR-100 offers a higher level of granularity, with more fine-grained categories such as flowers, insects, household items, and various types of animals and vehicles.

CINIC-10 [11] CINIC-10 is a dataset that stands as a 'bridge' between CIFAR-10 and ImageNet for image classification tasks. It combines 60,000 images of CIFAR-10, and 210,000 downsampled images of ImageNet. The images in CINIC-10 have a resolution of 32×32 pixels.

Food-101 [6] The Food-101 dataset is a collection of food images commonly used for image classification tasks. It contains 101 categories of food, with each category consisting of 1,000 images. The dataset covers a wide range of food items from various cuisines, including fruits, vegetables, desserts, and main dishes.

102-Flower [47] The 102-Flower dataset is a collection of images representing 102 different categories of flowers. Each image in the dataset has a fixed resolution of 256 pixels in both width and height. The dataset provides a diverse set of flower images.

The Oxford-IIIT Pet Dataset [51] The Oxford-IIIT Pet Dataset is a collection of images of cats and dogs belonging to 37 different breeds. The dataset includes a total of 7,349 images.

Places-365 [72] The Places-365 dataset is a large-scale dataset consisting of 365 different scene categories. It contains over 1.8 million images, each depicting a specific scene or environment. The images in the dataset have 256x256 pixels.

STL-10 [9] STL-10 dataset is a benchmark image dataset consisting of 10 different classes, including various animals, vehicles, and household objects. It contains a total of 5,000 training images and 8,000 test images, each with a resolution of 96 pixels by 96 pixels. The STL-10 dataset is derived from the larger ImageNet dataset but is specifically designed for low-resolution image classification tasks.

SVHN [43] The SVHN (Street View House Numbers) dataset is a large-scale dataset for digit recognition from real-world images. It consists of labeled images of house numbers captured from Google Street View. The dataset includes over 600,000 images for training and 26,032 images for testing. Each image is RGB and has a resolution of 32 pixels by 32 pixels.

We preprocess all datasets with standardization, additionally we rescale each image to $32px \times 32px$.

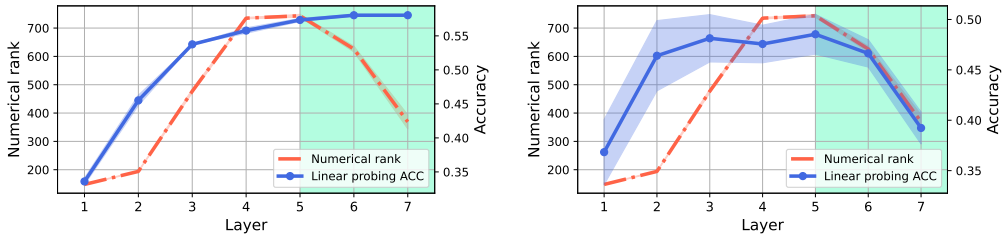
A.3 Compute

We conducted approximately 300 experiments to finalize our work, each taking about three wall-clock hours on a single NVIDIA A5000 GPU. We had access to a server with eight NVIDIA A5000 GPUs, enabling us to parallelize our experiments and reduce total computation time. We estimate to perform over 2000 experiments (including failed ones) during the development phase of the project.

B Full results

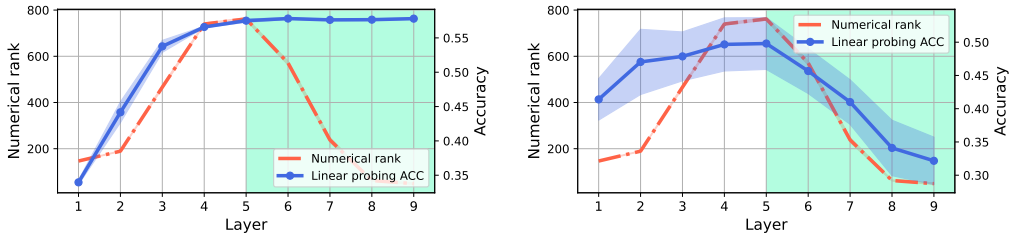
B.1 MLPs

In this section, we present the results of the tunnel effect for MLP architectures with different depths. All models are trained on CIFAR-10, and their OOD properties are evaluated on ten randomly selected classes of CIFAR-100.



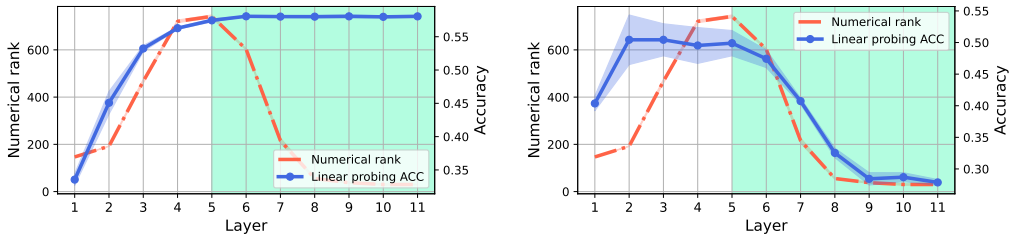
(a) In distribution (b) Out of distribution

Figure 11: In and out of distribution linear probing performance for MLP-6 trained on CIFAR-10. The shaded area depicts the tunnel, the red dashed line depicts the numerical rank and the blue curve depicts linear probing accuracy (in and out of distribution) respectively. Out-of-distribution performance is computed with random 10 class subsets of CIFAR-100.



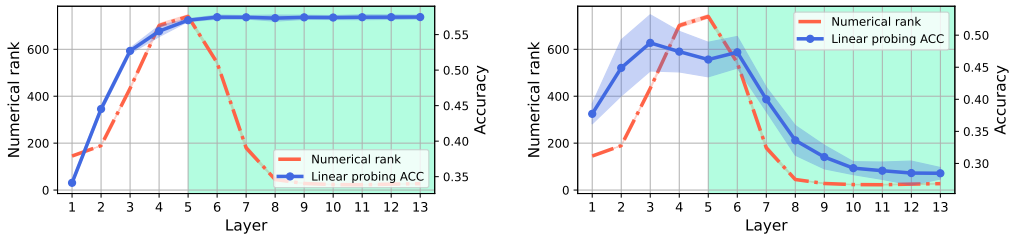
(a) In distribution (b) Out of distribution

Figure 12: In and out of distribution linear probing performance for MLP-8 trained on CIFAR-10. The shaded area depicts the tunnel, the red dashed line depicts the numerical rank and the blue curve depicts linear probing accuracy (in and out of distribution) respectively. Out-of-distribution performance is computed with random 10 class subsets of CIFAR-100.



(a) In distribution (b) Out of distribution

Figure 13: In and out of distribution linear probing performance for MLP-10 trained on CIFAR-10. The shaded area depicts the tunnel, the red dashed line depicts the numerical rank and the blue curve depicts linear probing accuracy (in and out of distribution) respectively. Out-of-distribution performance is computed with random 10 class subsets of CIFAR-100.

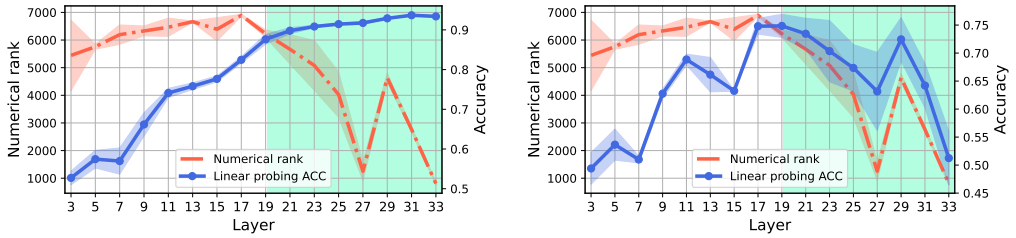


(a) In distribution (b) Out of distribution

Figure 14: In and out of distribution linear probing performance for MLP-12 trained on CIFAR-10. The shaded area depicts the tunnel, the red dashed line depicts the numerical rank and the blue curve depicts linear probing accuracy (in and out of distribution) respectively. Out-of-distribution performance is computed with random 10 class subsets of CIFAR-100.

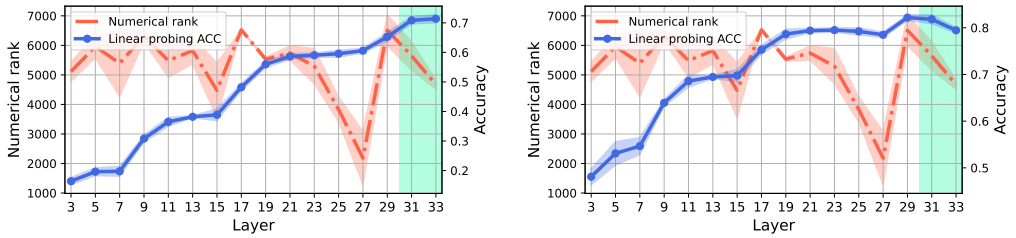
B.2 ResNet-34

In this section, we present the results of the tunnel effect for ResNet architectures with different depths. All models are trained on datasets CIFAR-10, CIFAR-100, and CINIC-10.



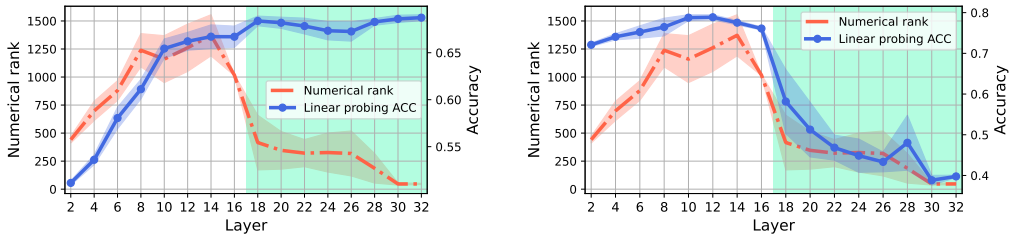
(a) In distribution (b) Out of distribution

Figure 15: In and out of distribution linear probing performance for ResNet-34 trained on CIFAR-10. The shaded area depicts the tunnel, the red dashed line depicts the numerical rank and the blue curve depicts linear probing accuracy (in and out of distribution) respectively. Out-of-distribution performance is computed with random 10 class subsets of CIFAR-100.



(a) In distribution (b) Out of distribution

Figure 16: In and out of distribution linear probing performance for ResNet-34 trained on CIFAR-100. The shaded area depicts the tunnel, the red dashed line depicts the numerical rank and the blue curve depicts linear probing accuracy (in and out of distribution) respectively. Out-of-distribution performance is computed on CIFAR-10.

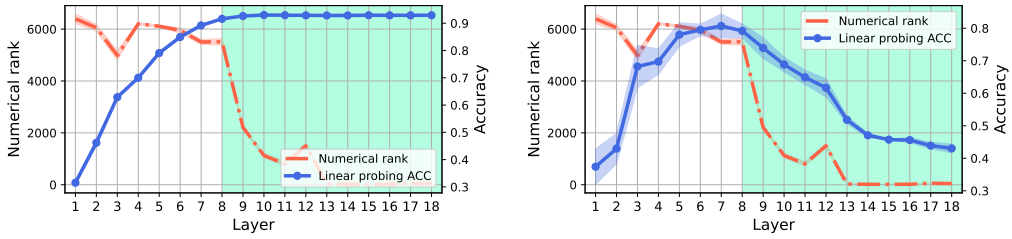


(a) In distribution (b) Out of distribution

Figure 17: In and out of distribution linear probing performance for ResNet-34 trained on CINIC-10. The shaded area depicts the tunnel, the red dashed line depicts the numerical rank and the blue curve depicts linear probing accuracy (in and out of distribution) respectively. Out-of-distribution performance is computed on subset of ten classes from CIFAR-100.

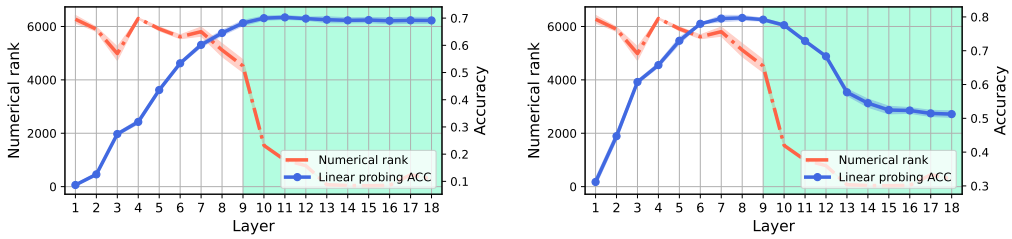
B.2.1 VGG-19

In this section, we present the results of the tunnel effect for VGG architectures with different depths. All models are trained on datasets CIFAR-10, CIFAR-100, and CINIC-10.



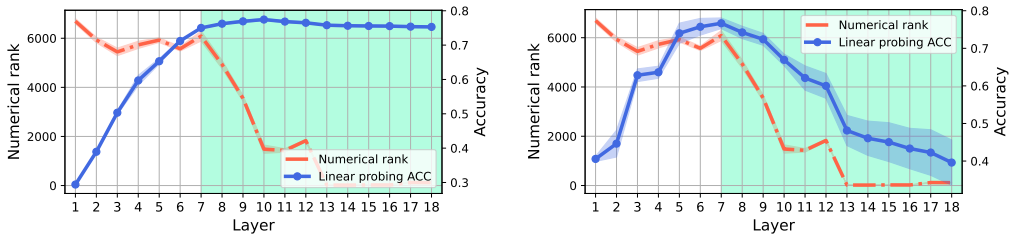
(a) In distribution (b) Out of distribution

Figure 18: In and out of distribution linear probing performance for VGG-19 trained on CIFAR-10. The shaded area depicts the tunnel, the red dashed line depicts the numerical rank and the blue curve depicts linear probing accuracy (in and out of distribution) respectively. Out-of-distribution performance is computed with random 10 class subsets of CIFAR-100.



(a) In distribution (b) Out of distribution

Figure 19: In and out of distribution linear probing performance for VGG-19 trained on CIFAR-100. The shaded area depicts the tunnel, the red dashed line depicts the numerical rank and the blue curve depicts linear probing accuracy (in and out of distribution) respectively. Out-of-distribution performance is computed on CIFAR-10.



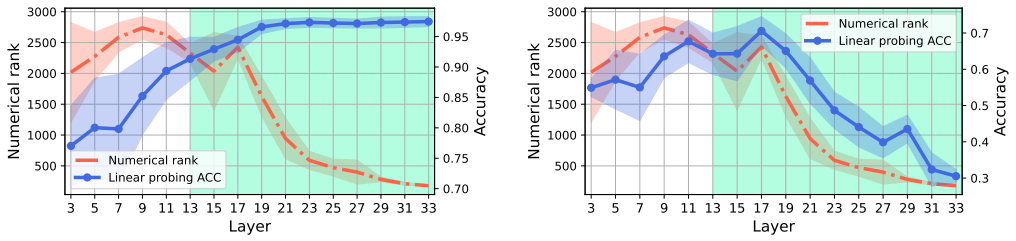
(a) In distribution

(b) Out of distribution

Figure 20: In and out of distribution linear probing performance for VGG-19 trained on CINIC-10. The shaded area depicts the tunnel, the red dashed line depicts the numerical rank and the blue curve depicts linear probing accuracy (in and out of distribution) respectively. Out-of-distribution performance is computed on subset of ten classes from CIFAR-100.

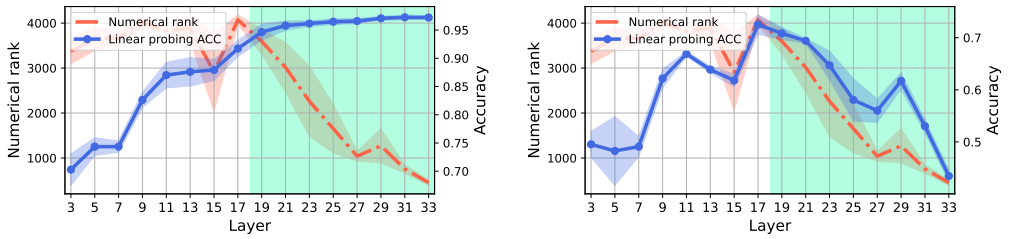
B.3 Dataset complexity experiments

B.3.1 ResNet-34



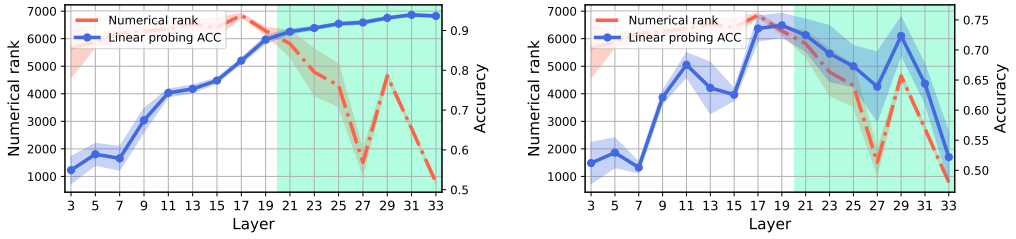
(a) In distribution (b) Out of distribution

Figure 21: In and out of distribution linear probing performance for ResNet-34 trained on a 3-class subset of CIFAR-10. The shaded area depicts the tunnel, the red dashed line depicts the numerical rank, and the blue curve depicts linear probing accuracy (in and out of distribution) respectively. Out-of-distribution performance is computed with random 10 class subsets of CIFAR-100.



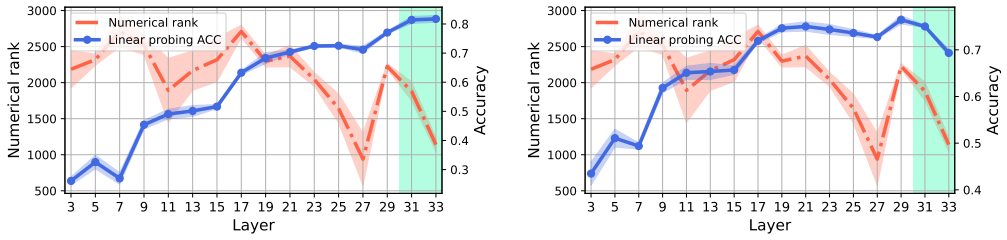
(a) In distribution (b) Out of distribution

Figure 22: In and out of distribution linear probing performance for ResNet-34 trained on a 5-class subset of CIFAR-10. The shaded area depicts the tunnel, the red dashed line depicts the numerical rank, and the blue curve depicts linear probing accuracy (in and out of distribution) respectively. Out-of-distribution performance is computed with random 10 class subsets of CIFAR-100.



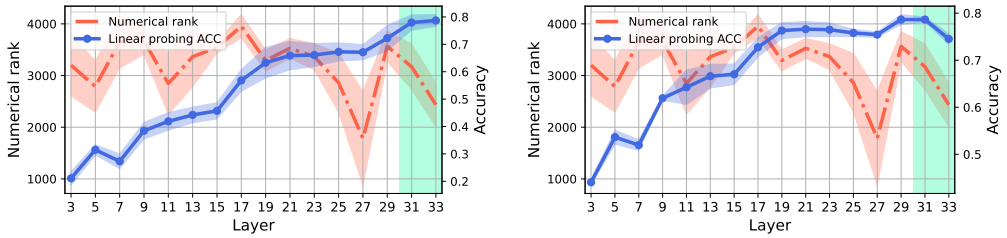
(a) In distribution (b) Out of distribution

Figure 23: In and out of distribution linear probing performance for ResNet-34 trained on CIFAR-10. The shaded area depicts the tunnel, the red dashed line depicts the numerical rank, and the blue curve depicts linear probing accuracy (in and out of distribution) respectively. Out-of-distribution performance is computed with random 10 class subsets of CIFAR-100.



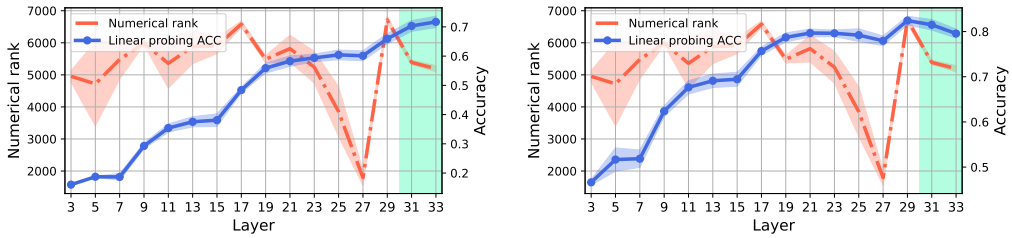
(a) In distribution (b) Out of distribution

Figure 24: In and out of distribution linear probing performance for ResNet-34 trained on a 30-class subset of CIFAR-100. The shaded area depicts the tunnel, the red dashed line depicts the numerical rank, and the blue curve depicts linear probing accuracy (in and out of distribution) respectively. Out-of-distribution performance is computed on CIFAR-10.



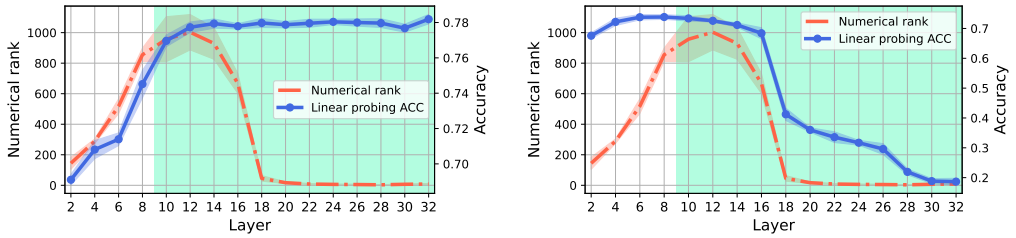
(a) In distribution (b) Out of distribution

Figure 25: In and out of distribution linear probing performance for ResNet-34 trained on a 50-class subset of CIFAR-100. The shaded area depicts the tunnel, the red dashed line depicts the numerical rank, and the blue curve depicts linear probing accuracy (in and out of distribution) respectively. Out-of-distribution performance is computed on CIFAR-10.



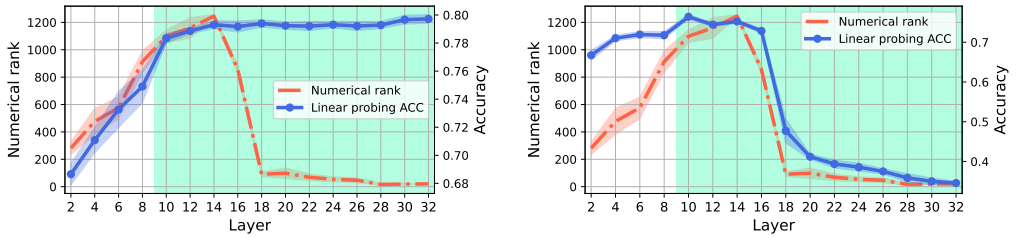
(a) In distribution (b) Out of distribution

Figure 26: In and out of distribution linear probing performance for ResNet-34 trained on CIFAR-100. The shaded area depicts the tunnel, the red dashed line depicts the numerical rank, and the blue curve depicts linear probing accuracy (in and out of distribution) respectively. Out-of-distribution performance is computed on CIFAR-10.



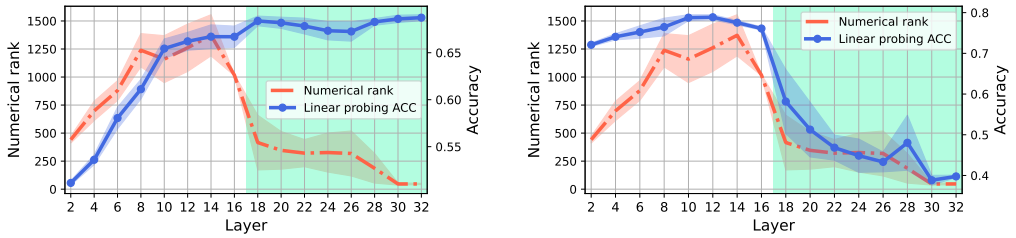
(a) In distribution (b) Out of distribution

Figure 27: In and out of distribution linear probing performance for ResNet-34 trained on a 3-class subset of CINIC-10. The shaded area depicts the tunnel, the red dashed line depicts the numerical rank, and the blue curve depicts linear probing accuracy (in and out of distribution) respectively. Out-of-distribution performance is computed on CIFAR-10.



(a) In distribution (b) Out of distribution

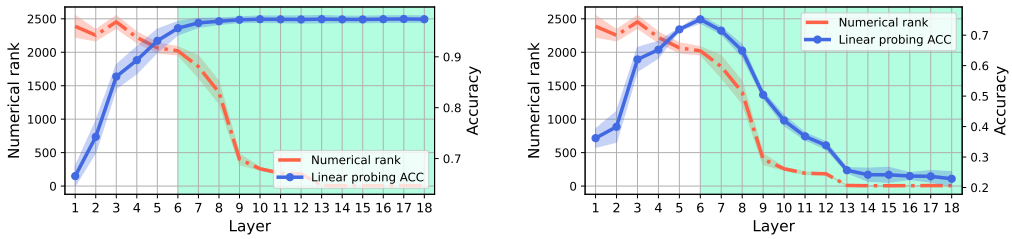
Figure 28: In and out of distribution linear probing performance for ResNet-34 trained on a 5-class subset of CINIC-10. The shaded area depicts the tunnel, the red dashed line depicts the numerical rank, and the blue curve depicts linear probing accuracy (in and out of distribution) respectively. Out-of-distribution performance is computed on CIFAR-10.



(a) In distribution (b) Out of distribution

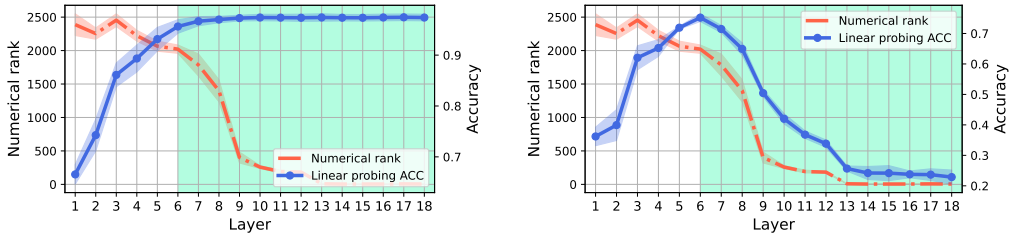
Figure 29: In and out of distribution linear probing performance for ResNet-34 trained on CINIC-10. The shaded area depicts the tunnel, the red dashed line depicts the numerical rank, and the blue curve depicts linear probing accuracy (in and out of distribution) respectively. Out-of-distribution performance is computed on CIFAR-10.

B.3.2 VGG-19



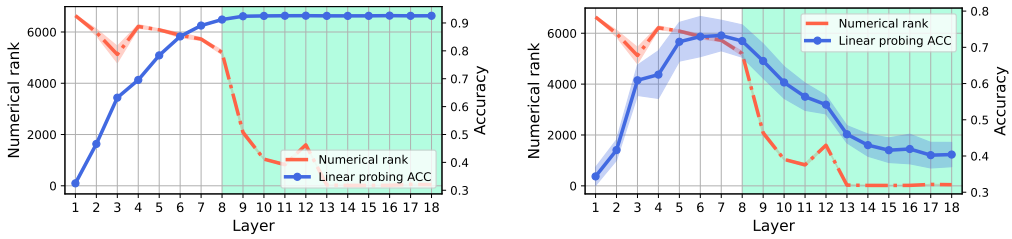
(a) In distribution (b) Out of distribution

Figure 30: In and out of distribution linear probing performance for VGG-19 trained on a 3-class subset of CIFAR-10. The shaded area depicts the tunnel, the red dashed line depicts the numerical rank, and the blue curve depicts linear probing accuracy (in and out of distribution) respectively. Out-of-distribution performance is computed with random 10 class subsets of CIFAR-100.



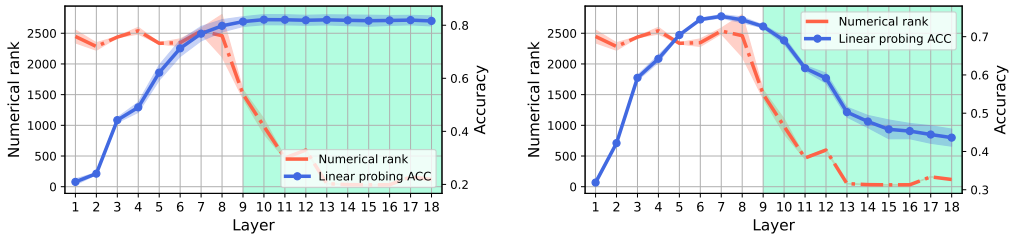
(a) In distribution (b) Out of distribution

Figure 31: In and out of distribution linear probing performance for VGG-19 trained on a 5-class subset of CIFAR-10. The shaded area depicts the tunnel, the red dashed line depicts the numerical rank, and the blue curve depicts linear probing accuracy (in and out of distribution) respectively. Out-of-distribution performance is computed with random 10 class subsets of CIFAR-100.



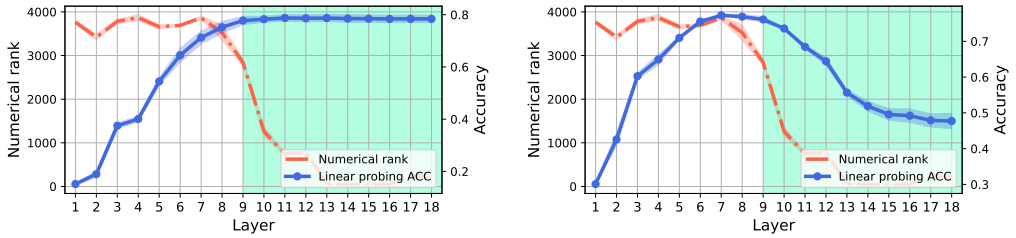
(a) In distribution (b) Out of distribution

Figure 32: In and out of distribution linear probing performance for VGG-19 trained on CIFAR-10. The shaded area depicts the tunnel, the red dashed line depicts the numerical rank, and the blue curve depicts linear probing accuracy (in and out of distribution) respectively. Out-of-distribution performance is computed with random 10 class subsets of CIFAR-100.



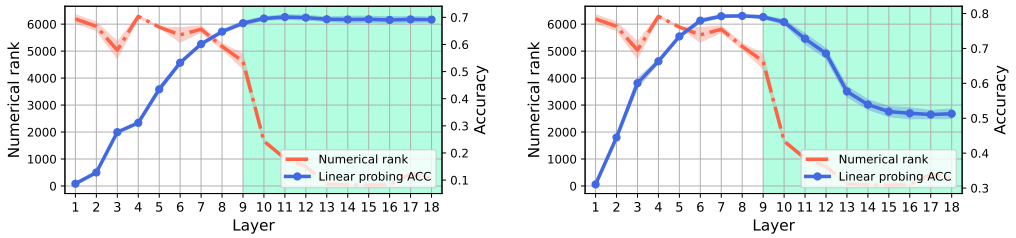
(a) In distribution (b) Out of distribution

Figure 33: In and out of distribution linear probing performance for VGG-19 trained on a 30-class subset of CIFAR-100. The shaded area depicts the tunnel, the red dashed line depicts the numerical rank, and the blue curve depicts linear probing accuracy (in and out of distribution) respectively. Out-of-distribution performance is computed on CIFAR-10.



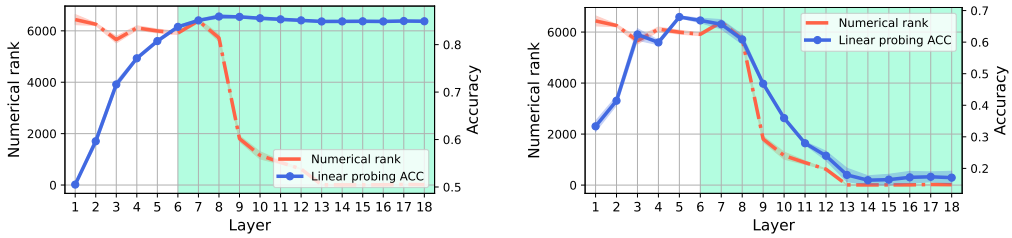
(a) In distribution (b) Out of distribution

Figure 34: In and out of distribution linear probing performance for VGG-19 trained on a 50-class subset of CIFAR-100. The shaded area depicts the tunnel, the red dashed line depicts the numerical rank, and the blue curve depicts linear probing accuracy (in and out of distribution) respectively. Out-of-distribution performance is computed on CIFAR-10.



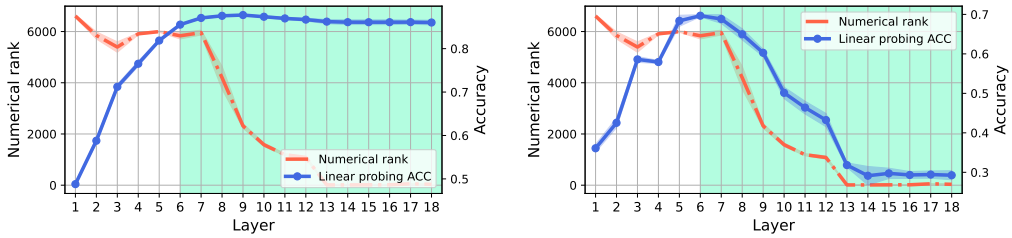
(a) In distribution (b) Out of distribution

Figure 35: In and out of distribution linear probing performance for VGG-19 trained on CIFAR-100. The shaded area depicts the tunnel, the red dashed line depicts the numerical rank, and the blue curve depicts linear probing accuracy (in and out of distribution) respectively. Out-of-distribution performance is computed on CIFAR-10.



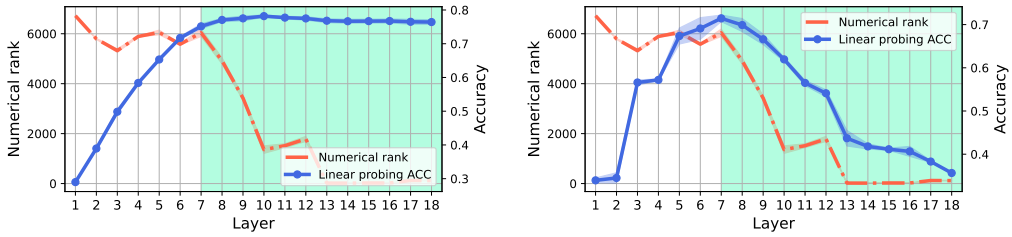
(a) In distribution (b) Out of distribution

Figure 36: In and out of distribution linear probing performance for VGG-19 trained on a 3-class subset of CINIC-10. The shaded area depicts the tunnel, the red dashed line depicts the numerical rank, and the blue curve depicts linear probing accuracy (in and out of distribution) respectively. Out-of-distribution performance is computed on CIFAR-10.



(a) In distribution (b) Out of distribution

Figure 37: In and out of distribution linear probing performance for VGG-19 trained on a 5-class subset of CINIC-10. The shaded area depicts the tunnel, the red dashed line depicts the numerical rank, and the blue curve depicts linear probing accuracy (in and out of distribution) respectively. Out-of-distribution performance is computed on CIFAR-10.

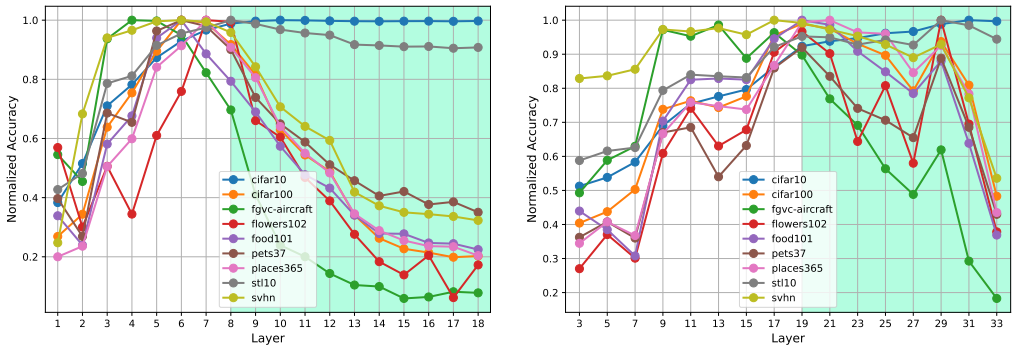


(a) In distribution (b) Out of distribution

Figure 38: In and out of distribution linear probing performance for VGG-19 trained on CINIC-10. The shaded area depicts the tunnel, the red dashed line depicts the numerical rank, and the blue curve depicts linear probing accuracy (in and out of distribution) respectively. Out-of-distribution performance is computed on CIFAR-10.

C Out of distribution generalization - extended results

In this experiment, we aim to determine if the tunnel consistently decreases the performance of models on out-of-distribution (OOD) datasets. To achieve this, we trained VGG-19 and ResNet34 models on CIFAR-10 and conducted linear probing on various OOD datasets. The results, depicted in Figure 39, are consistent across both the tested models and the datasets used. Notably, in all cases except for the training dataset (CIFAR-10), we observe a decline in performance starting from the beginning of the tunnel and continuing to degrade further. In the case of ResNet-34, there is a spike in performance at the 29^{th} layer, which aligns with the findings in the main paper. Interestingly, the dataset that exhibits the least deterioration is STL-10. This dataset consists of 10 classes, 9 of which overlap with classes found in CIFAR-10. However, the images in STL-10 are sampled from the ImageNet dataset. These results suggest that models can generalize well to OOD data that share semantic similarities with the in-distribution data. Note that the linear probing performance was normalized for better presentation.



(a) VGG-19 (b) ResNet-34

Figure 39: Out of distribution normalized linear probing performance for different datasets. The shaded area depicts the tunnel, different colors depict the linear probing performance on given dataset. Note that all the results are normalized for clarity of presentation.

The following experiment complements the analysis presented in the main paper, aiming to further explore the degradation of out-of-distribution performance caused by the tunnel effect. In this particular setup, the network is trained using CIFAR-10, and linear probes are trained and evaluated using subsets of CIFAR-100 with varying numbers of classes. The results, depicted in Figure 40, consistently demonstrate that regardless of the number of classes used to train the linear probes, the tunnel effect consistently leads to a decline in their performance. These findings confirm our observations from the main paper, indicating that the tunnel effect is a prevalent characteristic of the model rather than a peculiar artifact of the dataset or training setup.

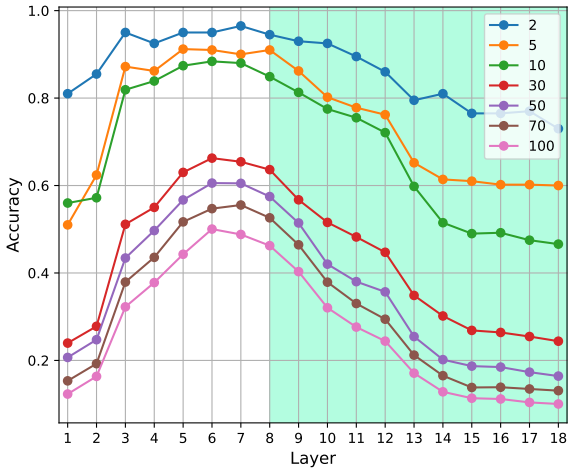


Figure 40: Source task with a fixed number of classes results in a tunnel consistently degrading the OOD performance for a different number of classes. VGG-19 is trained on CIFAR-10 and linear probes are trained on different subsets of CIFAR-100 with different numbers of classes. The tunnel is marked with a shaded color.

D Exploring the effects of task incremental learning on extractor and tunnel – extended results

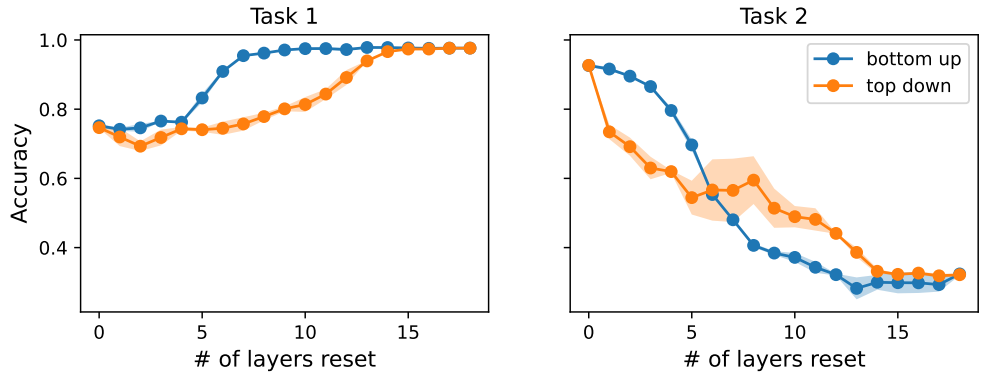


Figure 41: Substituting layer experiment. VGG-19 trained on the sequence of two tasks on split-CIFAR10. First task 3 class, second task 7 classes.

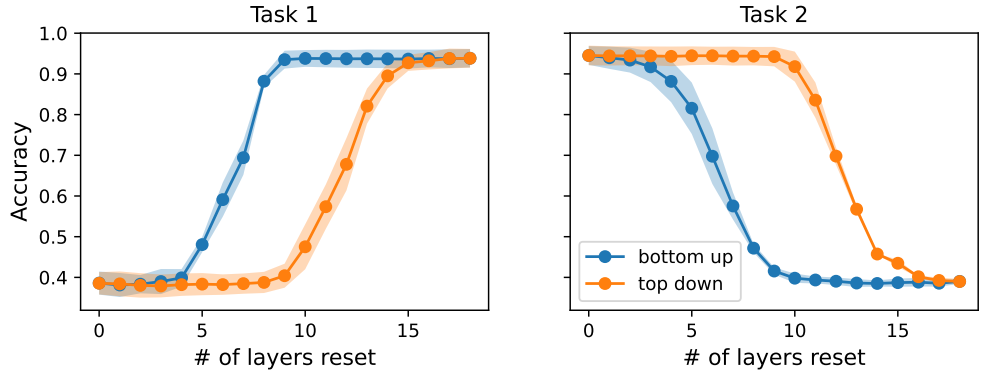


Figure 42: Substituting layer experiment. VGG-19 trained on the sequence of two tasks on split-CIFAR10. First task 5 class, second task 5 classes.

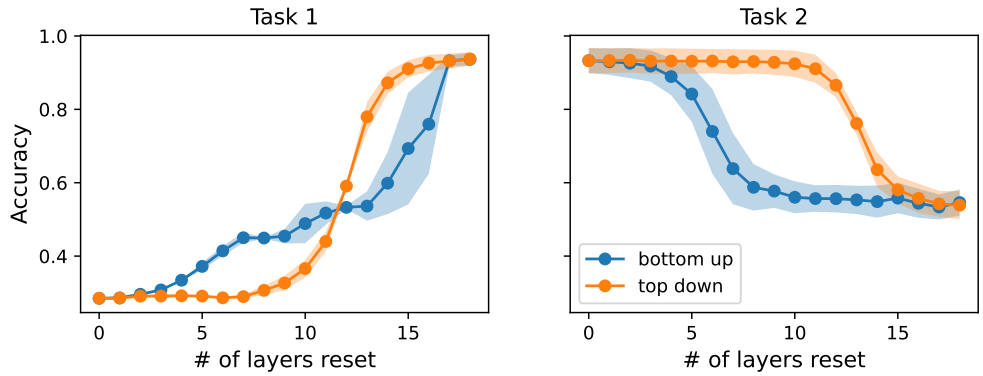


Figure 43: Substituting layer experiment. VGG-19 trained on the sequence of two tasks on split-CIFAR10. First task 7 class, second task 3 classes.

In this section we discuss in greater details experiment from section 4.1. First, we focus on examining reset layers experiment in case of sequence of tasks with different number of classes in section D.1. Next, we discuss the the discrepancies between our results and results presented in work [55].

D.1 Different number of classes in source and target tasks.

In this experiment, our aim is to gain a better understanding of tunnel immunity to catastrophic forgetting. Specifically, we are interested in exploring scenarios where the number of classes differs in each task. To analyze this scenario, we conducted three experiments using the VGG-19 network. We trained the network on sequences of two tasks, each composed of CIFAR-10 classes with different splits: $(3, 7)$, $(5, 5)$, and $(7, 3)$.

During training, we saved the model after completing the first and second tasks, denoted as M_1 and M_2 respectively. When we refer to $M_1^{1:x} + M_2^{x+1:n}$, we mean that the network consists of the first x layers with parameters from after completing the first task, combined with the remaining $n - x$ layers from the network after completing the second task.

Here, instead of a table we present the results using plots, see Figure 41 for the reference. The y-axis values represent the accuracy of the model when substituting a certain number of layers, denoted as x. The blue plot represents the situation where we substitute layers starting from the bottom ($M_1^{1:x} + M_2^{x+1:n}$), while the orange plot represents the opposite scenario ($M_2^{1:x} + M_1^{x+1:n}$). Please note the change in subscripts.

In Figure 42, we observe that when the tasks have an equal number of classes, the tunnel is preserved perfectly. Specifically, substituting 10 layers from the top down does not affect the performance on the second task, and substituting more than 8 layers does not yield any improvement on the first task.

Conversely, in Figure 41, substituting more than 7 layers from the bottom up does not lead to any improvement in the second task. Additionally, substituting any layers from the top down actually harms the performance on the second task. This suggests that while the network encountered more classes in the second task, it built upon the existing tunnel, maintaining its performance on the first task.

In the opposite scenario, where the second task involves fewer classes, a reverse situation is observed. Substituting any layers from the top down negatively impacts the performance on the first task, while substituting 10 layers from the top down does not affect the performance on the second task. This suggests that the network successfully reused a portion of the tunnel from the first task while discarding the unnecessary part.

D.2 On the primary source of catastrophic forgetting on split-CIFAR10 task.

There is an ongoing discussion surrounding the layers responsible for driving the phenomenon of forgetting. In a study [55], authors claim that "Higher layers are the primary source of catastrophic forgetting on split CIFAR-10 task." However, our findings present a different perspective compared to the conclusions drawn in that research. Specifically, the results presented in Section 4.1 and Section D.1 indicate that there exist continual learning scenarios where the deeper layers do not contribute to catastrophic forgetting. Instead, we show that in certain scenarios the earlier layers are responsible for performance degradation, while the deeper layers remain unaffected due to their task-agnostic nature. This insight is of particular significance because many studies have built upon the assumption that mainly deeper layers are responsible for catastrophic forgetting, potentially leading to inadequate or inefficient continual learning mechanisms [13, 52].

It is important to note that the tunnel hypothesis effect holds for overparameterized networks. In contrast, the authors of [55] evaluated their claims using the VGG-13 network, with the width of the layers reduced by a factor of four. This discrepancy plays a crucial role in tunnel formation, as it reduces the model's capacity. Figures 44- 49 illustrate the disparity between these models in the reset experiment.

From this comparison, the main conclusion emerges that the question of "which layers are the primary source of catastrophic forgetting?" is nuanced and contingent upon multiple factors.

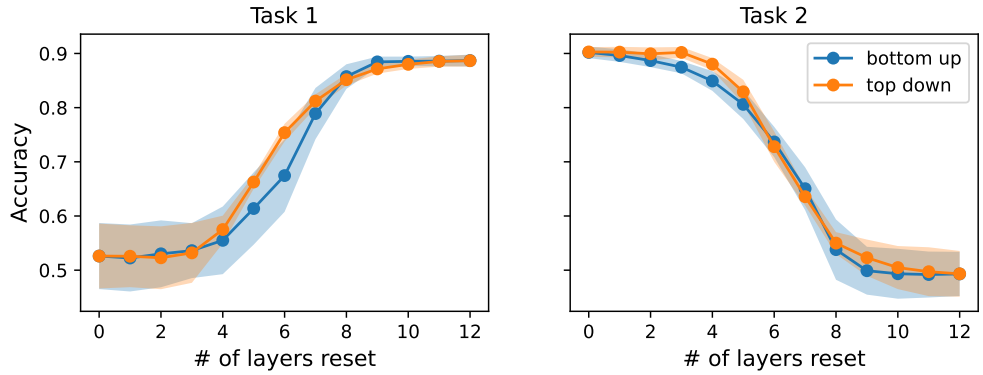


Figure 44: Substituting layer experiment. VGG-13, width factor = 0.25, trained on the sequence of two tasks on split-CIFAR10.

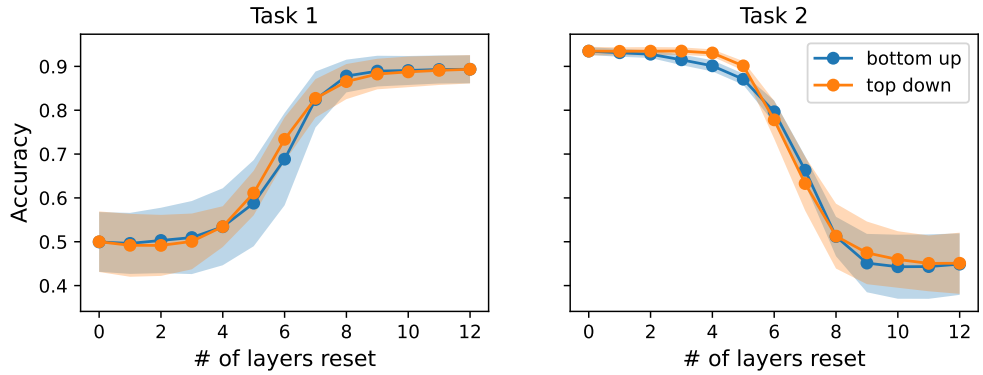


Figure 45: Substituting layer experiment. VGG-13, width factor = 1, trained on the sequence of two tasks on split-CIFAR10.

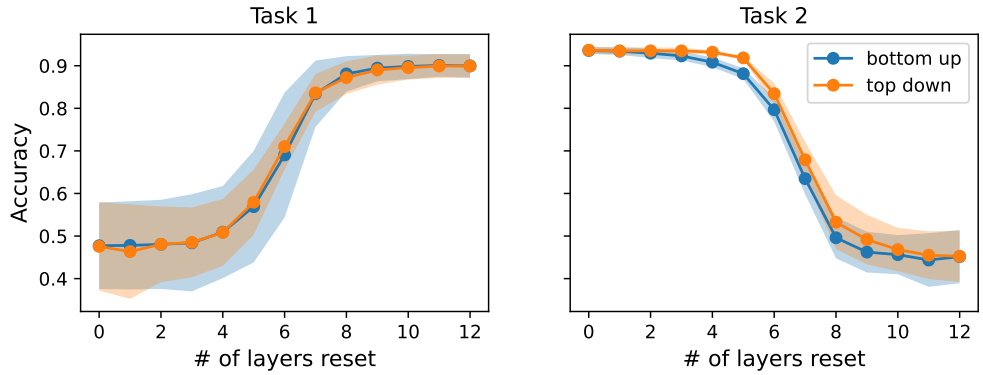


Figure 46: Substituting layer experiment. VGG-13, width factor = 2, trained on the sequence of two tasks on split-CIFAR10.

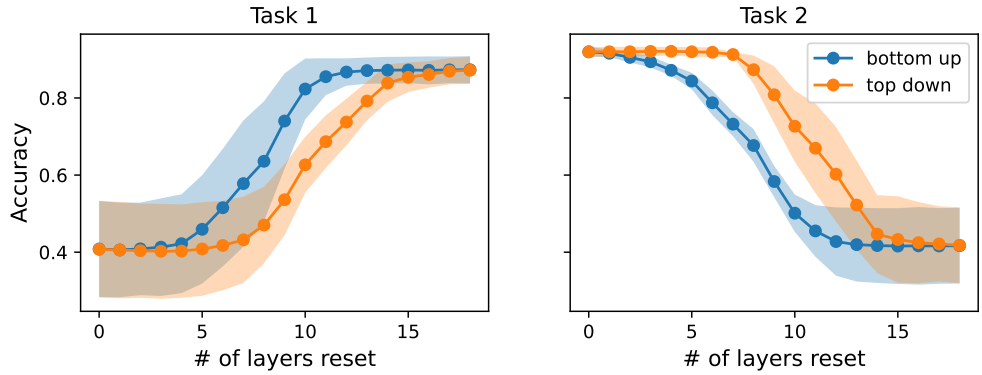


Figure 47: Substituting layer experiment. VGG-19, width factor = 0.25, trained on the sequence of two tasks on split-CIFAR10.

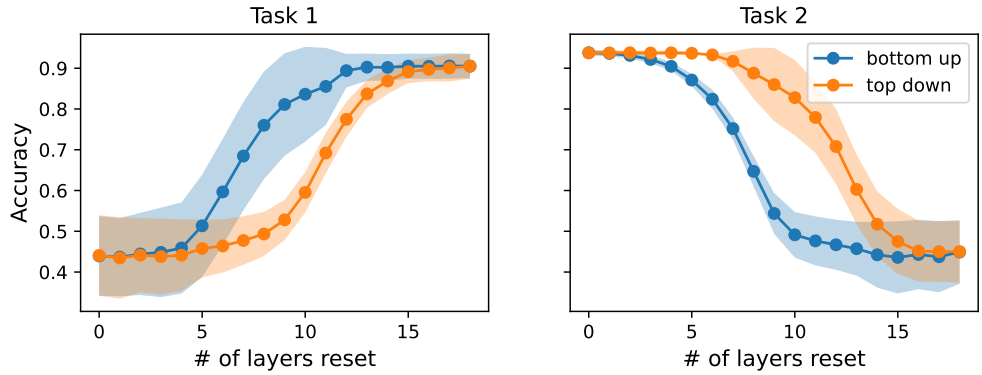


Figure 48: Substituting layer experiment. VGG-19, width factor = 1, trained on the sequence of two tasks on split-CIFAR10.

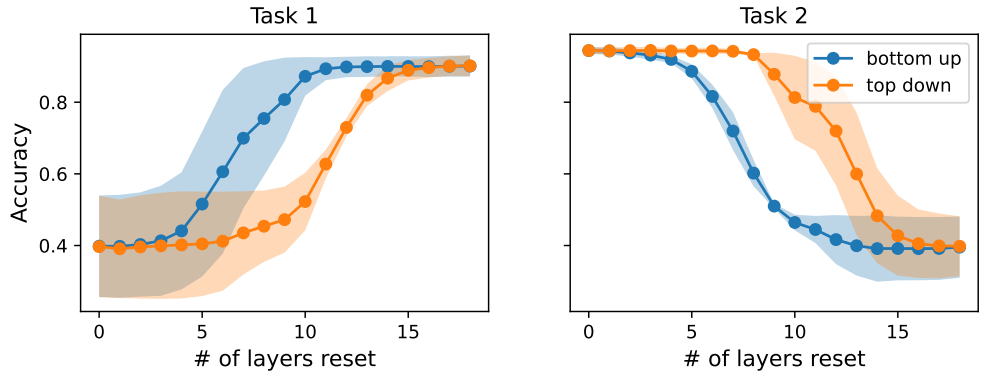


Figure 49: Substituting layer experiment. VGG-19, width factor = 2, trained on the sequence of two tasks on split-CIFAR10.

E CKA similarity

The Centered Kernel Alignment (CKA) similarity is a measure commonly used in machine learning and neuroscience to quantify the similarity between two representations or feature spaces. It provides a way to assess the similarity of representations learned by different models or layers, even when the representations may have different dimensionalities. CKA is invariant to orthogonal transformations, such as isotropic scaling, permutations, reflections and rotations. This invariance property is particularly valuable when comparing representations that have undergone different preprocessing or normalization steps. By accounting for the underlying relationships between representations while being insensitive to orthogonal transformations, CKA enables a more meaningful and reliable assessment of similarity, aiding in tasks such as model comparison, representation learning, and understanding the neural code. For computational reasons, we use a modification of CKA index given by the following formula.

CKA similarity: Let $\mathbf{X}_i \in \mathbb{R}^{m \times p_1}$, $\mathbf{Y}_i \in \mathbb{R}^{m \times p_2}$ be the representations matrices from i^{th} minibatch of two layers of m samples and p_1 and p_2 number of features respectively. Similarly to [45], we estimate CKA index by averaging over k mini-batches:

$$s\text{CKA} = \frac{\frac{1}{k}\sum_{i=1}^{k} \text{HSIC}\left(\mathbf{X}_i\mathbf{X}_i^\top, \mathbf{Y}_i\mathbf{Y}_i^\top\right)}{\sqrt{\frac{1}{k}\sum_{i=1}^{k} \text{HSIC}\left(\mathbf{X}_i\mathbf{X}_i^\top, \mathbf{X}_i\mathbf{X}_i^\top\right)}\sqrt{\frac{1}{k}\sum_{i=1}^{k} \text{HSIC}\left(\mathbf{Y}_i\mathbf{Y}_i^\top, \mathbf{Y}_i\mathbf{Y}_i^\top\right)}}, \tag{1}$$

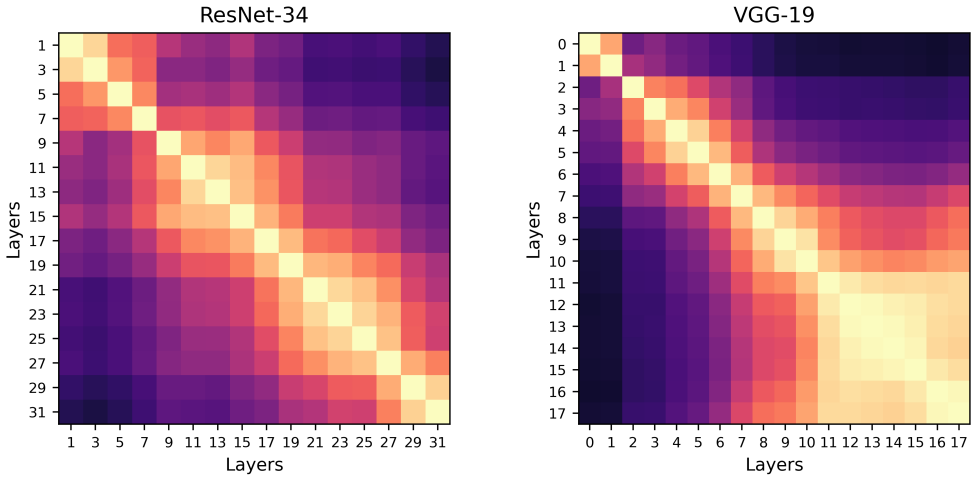
where HSIC is an unbiased estimate or of the HSIC score [45]:

$$\text{HSIC}(\mathbf{K}, \mathbf{L}) = \frac{1}{n(n-3)}\left(\text{tr}(\tilde{\mathbf{K}}\tilde{\mathbf{L}}) + \frac{\mathbf{1}^\top\tilde{\mathbf{K}}\mathbf{1}\mathbf{1}^\top\tilde{\mathbf{L}}\mathbf{1}}{(n-1)(n-2)} - \frac{2}{n-2}\mathbf{1}^\top\tilde{\mathbf{K}}\tilde{\mathbf{L}}\mathbf{1}\right), \tag{2}$$

where $\tilde{\mathbf{L}} = \mathbf{L} - diag(\mathbf{L})$.

CKA is a normalized similarity index, hence value 1 means that representations matrices are identical.

In Figure 50, we present the plots of representations similarity for ResNet-18 and VGG-19 networks.



(a) ResNet-34 (b) VGG-19

Figure 50: CKA-similarity across network's layers for ResNet-34 and VGG-19.

F Inter and Intra class variance

Understanding the concepts of inter-class and intra-class variance is particularly important in the context of deep neural network representations analysis for classification tasks. In this context,

inter-class variance refers to the variability between different classes or categories of data. On the one hand, they capture the representation of the linear separability of a given task [4]. On the other hand, intra-class variance is an indicator of representations transferability [22, 71].

Let $\mathbf{X}_j \in \mathbb{R}^{t_j \times p}$ be the representations matrix for samples from j^{th} class. $\frac{1}{C}\sum_{j=1}^{C}\left(\frac{1}{t_j}\sum_{x_i \in \mathbf{X}_j}\|\mathbf{f}_i - \mu\left(\mathbf{X}_j\right)\|^2\right)$ is the intra-class variance, where $\mathbf{f}_i$ is a representation of sample x_i, $\mu\left(\mathbf{X}_j\right)$ is the mean representation of representations matrix $\mathbf{X}_j$, and C is the number of classes. Then $\frac{1}{C(C-1)}\sum_{j=1}^{C}\sum_{k=1,k\neq j}^{C}\|\mu\left(\mathbf{X}_j\right) - \mu\left(\mathbf{X}_k\right)\|^2$ is the inter-class variance.

G Tunnel development

In this section, we provide a more detailed analysis of the evolution of numerical rank in VGG-19 dataset. In this experiment, we save the checkpoint of the network every epoch and calculate its numerical rank. The results are depicted in Figure 51.

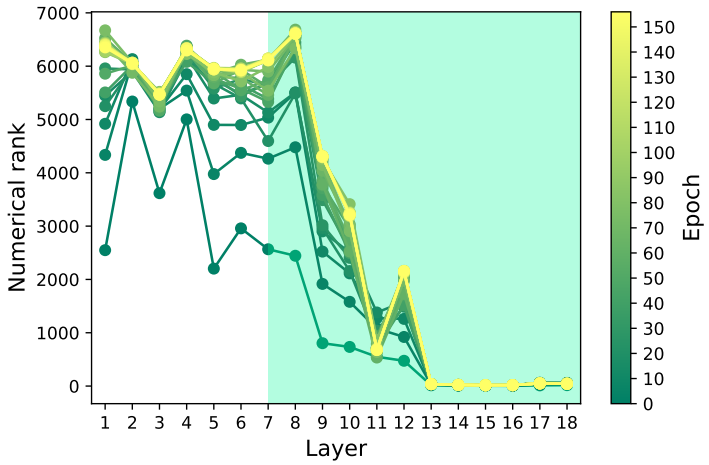
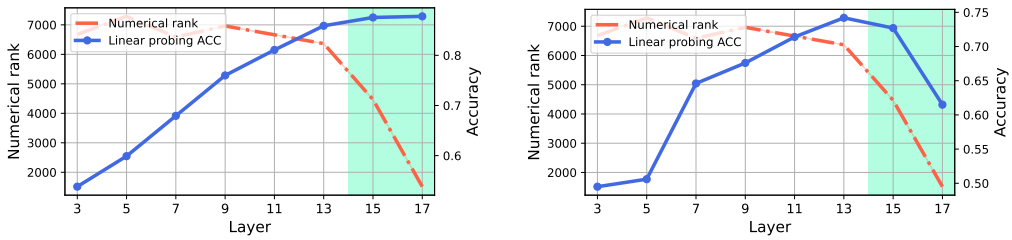


Figure 51: Evolution of numerical rank of VGG-19 representations throughout the training on CIFAR-10.

Initially, during the early epochs, the rank collapses primarily in the deeper layers. Throughout the training process, two distinct patterns can be observed. Firstly, the numerical rank of representations from the earlier layers tends to increase. Secondly, the numerical rank of representations from the deeper layers decreases. Interestingly, the place of this transition aligns with the beginning of the tunnel in the network. Once the numerical rank in deeper layers collapsed in the first gradient steps, as shown in Figure 6, it remained collapsed throughout the whole training.

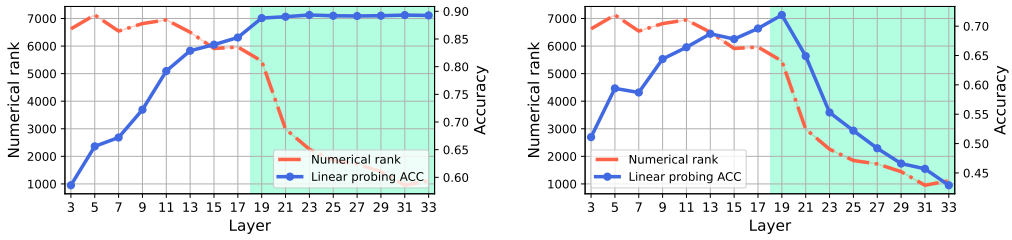
H ResNets without skip connections

In this section, we delve into the impact of skip connections on the formation of the tunnel effect. To investigate this relationship, we trained ResNet models (ResNet-18 and ResNet-34) without skip connections on CIFAR-10 and conducted the same analysis used in the main paper. Specifically, we examined the linear probing performance for both in-distribution and out-of-distribution data and estimated the representations' numerical rank. The results, depicted in Figure 52 and Figure 53, highlight the significance of skip connections in the formation of the tunnel effect. Firstly, in the absence of skip connections (Plainnets), the tunnel effect is slightly more pronounced, with the model's performance saturating two layers earlier than standard ResNet networks. Secondly, the rank of the representations exhibits a more predictable pattern without the spike at 29^{th} layer. This suggests that the spike in the numerical rank and in OOD performance is related to the skip connections. Interestingly, the numerical rank in both networks is higher than in the case of VGGs. The reason for this difference needs a further investigation. Lastly, the presence or absence of skip connections does not alter the degradation of out-of-distribution performance. However, in the absence of skip connections, the deterioration is more severe, aligning with the observation that it correlates with the numerical rank of the representations.



(a) In distribution (b) Out of distribution

Figure 52: In and out of distribution linear probing performance for ResNet-18 without skip connections trained on CIFAR-10. The shaded area depicts the tunnel, the red dashed line depicts the numerical rank and the blue curve depicts linear probing accuracy (in and out of distribution) respectively. Out-of-distribution performance is computed with random 10 class subsets of CIFAR-100.



(a) In distribution (b) Out of distribution

Figure 53: In and out of distribution linear probing performance for ResNet-34 without skip connections trained on CIFAR-10. The shaded area depicts the tunnel, the red dashed line depicts the numerical rank and the blue curve depicts linear probing accuracy (in and out of distribution) respectively. Out-of-distribution performance is computed with random 10 class subsets of CIFAR-100.