Robust Weight Imprinting: Insights from Neural Collapse and Proxy-Based Aggregation

Anonymous authors
Paper under double-blind review

Abstract

The capacity of foundation models allows for their application to new, unseen tasks. The adaptation to such tasks is called transfer learning. An efficient transfer learning method that circumvents parameter optimization is imprinting. It has been reinvented several times, but not systematically studied. In this work, we propose the general `IMPRINT` framework, identifying three main components: generation, normalization, and aggregation. Through the lens of this framework, we conduct an in-depth analysis and a comparison of the existing methods. Our findings reveal the benefits of representing novel data with multiple proxies in the generation step and show the importance of proper normalization. Beyond an extensive analytical grounding, our framework enables us to propose a novel variant of imprinting which outperforms previous work on transfer learning tasks by 6%. This variant determines proxies through clustering motivated by the neural collapse phenomenon – a connection that we draw for the first time. We publicly release our code at *(link removed for review)*.

1 Introduction

In machine learning applications, training models from scratch is often not viable due to limitations in data and compute. A popular solution is to apply transfer learning (Bengio, 2012; Yosinski et al., 2014) based on foundation models (`FM`s) (Bommasani et al., 2021) that are pre-trained on a large amount of data. To tune a `FM` to a novel task, e.g., for classification, a common procedure is to freeze the model parameters and replace the output layer with a new head. A particularly simple method for implementing such a new head was proposed by Qi et al. (2018) and coined *imprinting*.

Imprinting. In the original work by Qi et al. (2018), the last-layer weight vector of a novel class is set to the normalized average of its scaled embedding vectors, i.e., its class mean. These class means are representatives of the classes, which we denote as *proxies*. In general, we refer to imprinting as efficient learning methods without the need for cross-class statistics or gradient-based optimization. A plethora of studies have emerged surveying this technique by adding complexity and adaptability (Li et al., 2021; Yan et al., 2023; Passalis et al., 2020; Khan et al., 2021; Cristovao et al., 2022; Siam et al., 2019; Gidaris & Komodakis, 2018; Zhu et al., 2022). Despite its many adaptations, imprinting lacks a systematic comparison that unifies them. Understanding its variations could unlock greater efficiency and performance across many fields, making the method even more versatile and impactful.

We present `IMPRINT`, a framework that enables a comprehensive analysis of existing imprinting techniques. More precisely, we generalize prior work by decomposing imprinting into three principal steps (see fig. 1). During generation (`GEN`) of weights, the method selects representative data samples and generates one or more weight vectors (proxies) per class. Normalization (`NORM`) is crucial, as the generated weights need to be balanced. Aggregation (`AGG`) entails the computation of the final output, e.g., a class label. The computational efficiency of imprinting allows us to perform a large number of experiments. Through `IMPRINT`, we are able to propose a novel, best-performing imprinting strategy using multi-proxy weight imprinting in combination with L^2 normalization, outperforming previously studied methods, as depicted in fig. 2.

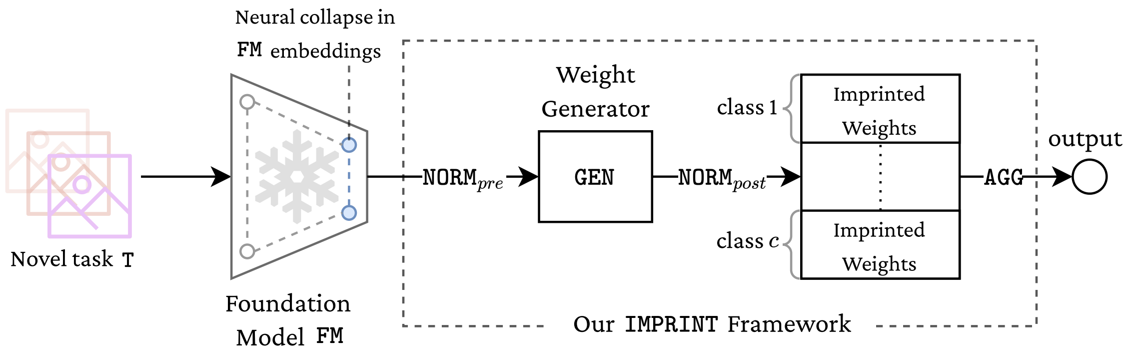


Figure 1: Overview of our `IMPRINT` framework. The foundation model `FM` is frozen and shows neural collapse. The weight generator (`GEN`) uses training data from a novel task `T` to consecutively generate one or more weight vectors (proxies) per class $1, \ldots, C$. The final output for the test data in `T` is computed by an aggregation (`AGG`) mechanism. Embeddings and generated weights are normalized according to $\texttt{NORM}_{pre}$ and $\texttt{NORM}_{post}$, respectively. During inference, embeddings are normalized according to $\texttt{NORM}_{inf}$ (not shown here).

Neural Collapse. When neural networks are trained to reach near-zero loss, their penultimate-layer embeddings collapse to the class means (Papyan et al., 2020; Zhu et al., 2021). We investigate this phenomenon, called neural collapse, as a potential explanation for when and why imprinting works. Our analysis proves that there exists a relationship between a measure of neural collapse and the success of imprinting. Since quantification of this phenomenon is possible in a post-hoc fashion over the `FM` features, we believe that these insights will contribute to further development of imprinting methods, as well as, training regimes of `FM`s that are more suitable for transfer learning via imprinting.

Contributions. In summary, our main results and contributions are:

- We deconstruct weight imprinting into the `IMPRINT` framework composed of generation, normalization, and aggregation, and discuss variations for each of them, while identifying prior work as special cases (section 3.1). To the best of our knowledge, we are the first to conduct a comprehensive analysis of imprinting to this scale (section 4).
- We present a new, outperforming imprinting method utilizing k-means clustering for weight generation (section 5.1) and show its benefits in certain few-shot scenarios (section 5.2).
- To the best of our knowledge, we are the first to identify a relationship between the degree of neural collapse and imprinting success (section 5.3).

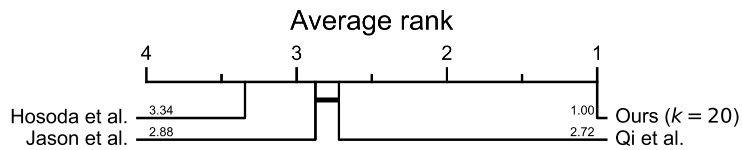
2 Related Work

Imprinting and Few-Shot Learning. Weight imprinting is implemented by setting the final layer weights for the novel classes to the scaled average of the embedding vectors of their training samples and was first introduced by Qi et al. (2018) for the few-shot learning scenario. The authors find that for up to 20 samples, using a combination of imprinting and fine-tuning outperforms other state-of-the-art methods, including nearest neighbor algorithms. In contract, in our work we do not limit the number of samples and perform no fine-tuning on the imprinted weights to maintain efficiency.

Imprinting has been applied to object detection (Li et al., 2021; Yan et al., 2023), multi-label classification (Khan et al., 2021), semantic segmentation (Siam et al., 2019), and combined with an attention mechanism to generate weights for the novel classes in a few-shot classification task (Gidaris & Komodakis, 2018).

Table 1 & Figure 2: Previously studied imprinting strategies are special cases within `IMPRINT`. We evaluate on 12 different classification tasks `Ts` derived from *MNIST*, *FashionMNIST*, and *CIFAR-10*, each with 10 classes or subsets thereof, and 4 pre-trained models (`resnet18`, `resnet50`, `vit_b_16`, `swin_b`). The proposed configuration ("Ours") derived from `IMPRINT` outperforms previous work across `FMs` and `Ts` by a large margin with statistical significance. Here, $k = 20$ is used, highlighting the gain of using multiple proxies per class.

Work	$\texttt{NORM}_{pre}$	`GEN`	$\texttt{NORM}_{post}$	$\texttt{NORM}_{inf}$	`AGG`	Avg. acc. %
Qi et al. (2018)	`L2`	`mean`	`L2`	`L2`	`max`	85.75
Hosoda et al. (2024)	`none`	`mean`	`quantile`	`none`	`max`	84.22
Janson et al. (2022)	`none`	`mean`	`none`	`none`	`1-nn`	85.59
Ours	`L2`	`k-means`	`L2`	`L2`	`max`	**91.11**



Hosoda et al. (2024) apply imprinting using quantile normalization to ensure statistical similarity between new and existing weights. We consider this as one normalization scheme in our framework. Zhang et al. (2021) apply imprinting in chest radiography for detection of COVID-19 and find that it yields better results than joint gradient descent training of all classes when only few samples are available. They speculate whether normalization is a constraint in their imprinting model.

Before the era of deep learning, Mensink et al. (2013) analyze the transferability of hand-crafted image features. They use a "nearest class multiple centroids" (NCMC) classifier with multiple proxies generated from a k-means clustering algorithm. In combination with metric learning, they compare favorably against the m-nearest neighbor algorithm. Our work, on the other hand, highlights efficient transfer learning provided by foundation models.

Transfer Learning. Using embedding vectors from pre-trained models is a simple and widely used transfer learning approach, established in the seminal works on computer vision (Donahue et al., 2014) and natural language processing (Devlin et al., 2019). Kornblith et al. (2019) show that pre-training performance of a model is highly correlated with the performance of the resulting embedding vectors in downstream tasks. In addition, Huh et al. (2016) provide insights into the required quality of pre-training data. Our work is orthogonal to these studies, since we focus on studying weight generation, normalization, and aggregation techniques applied later on for new task adaptation.

Continual Learning (CL). Although we investigate transfer learning scenarios, we review the imprinting applications and results from CL. Rebuffi et al. (2017) dynamically select a subset of examples for each class and update internal representations via gradient descent. They use a nearest mean classifier (NMC) with respect to the saved examples. Janson et al. (2022) use an NMC classifier as well and achieve good performance on CL benchmarks without any fine-tuning of the embeddings. However, they do not investigate the effect of normalization and using multiple proxies.

Findings of Prabhu et al. (2023) show that a simple, approximate m-nearest neighbor classifier outperforms existing methods in an Online CL setting when all data can be stored. In our work, however, we compare imprinting all data to a limited number of more representative proxies striving for efficiency.

Neural Collapse (NC). The phenomenon of NC was identified by Papyan et al. (2020) and refers to the convergence of the last-layer weight vectors to class means. It was shown that, regardless of the loss function, optimizer, batch-normalization, or regularization, NC will eventually occur (provided the training data has a balanced distribution) (Zhu et al., 2021; Han et al., 2022; Kothapalli, 2023). Nevertheless, complete neural collapse is practically unrealistic (Tirer et al., 2023). In transfer learning, Galanti et al. (2022) show that NC occurs on new samples and classes from the same distribution as the pre-training dataset, highlighting the usability of foundational models in such scenarios. In our work, we expand the survey on NC by experimenting with out-of-distribution classes belonging to different datasets and linking their degree of collapse to the success of certain imprinting strategies.

3 Methods

3.1 `IMPRINT`

In order to find out how to best and efficiently set the classifier weights of a foundational model in new, previously unseen tasks `T`, we create the `IMPRINT` framework (see fig. 1) that generalizes previous work, specifically all the methods that work without access to cross-class statistics and gradient-based training. Thereby, we unify all the existing imprinting strategies described in section 2.

Overview. We analyze the effect of weight generation (`GEN`), normalizations ($\texttt{NORM} = \{\texttt{NORM}_{pre}, \texttt{NORM}_{post}, \texttt{NORM}_{inf}\}$), and aggregation (`AGG`). The `IMPRINT` framework depicted in fig. 1 consists of three main building blocks: a foundation model `FM`, a weight generator `GEN`, and extendable classifier weights that are imprinted. The `FM` remains frozen throughout the experiments. It receives data from `T` as inputs and produces embedding vectors. The training process generates weight vectors for each of the C classes in `T`. Hereby, embeddings from the `FM` are normalized before the generation (`GEN`) step according to $\texttt{NORM}_{pre}$. The generated weight vectors per class are referred to as *proxies*, prototypes, or representatives (Movshovitz-Attias et al., 2017; Snell et al., 2017; Yang et al., 2018; Yu et al., 2020). These proxies are normalized according to $\texttt{NORM}_{post}$. As in the work by Qi et al. (2018), we do not use bias values. To classify the test data in `T` during inference, it is first embedded by the `FM`, normalized according to $\texttt{NORM}_{inf}$, and finally aggregated by `AGG`, resulting in a predicted class label.

Generalization of Previous Methods. Previously proposed imprinting methods can be defined as special cases of our framework. We design `IMPRINT` such that every method can be defined by a single combination of `GEN`, `NORM`, and `AGG`, of which we inspect all possible combinations.

Weight Generation (`GEN`). The purpose of `GEN` is to determine how the embeddings of the training data in `T` are used to form the new weights. In contrast to Qi et al. (2018) which only incorporates one proxy per class (the mean), we add flexibility by allowing each class to have multiple proxies as Mensink et al. (2013). This enables non-linear classification. We denote the number of proxies as k, ranging between 1 and the number of samples. We investigate the following operations **conducted per class** to generate proxies:

- `all`: All embeddings (denoted as $k = \text{all}$).
- `k-random`: k random embeddings.
- `mean`: The mean of all embeddings.
- `k-means`: k-means cluster centers. $k = 1$ is the same as `mean`.
- `k-medoids`: k-medoids cluster centers.
- `k-cov-max` (covariance-maximization): Top k embeddings by covariance.
- `k-fps` (farthest-point sampling): Iteratively selecting k embeddings, such that it maximizes the distance from already selected ones (starting with a random sample).

We choose this diverse list of methods to cover a wide range of approaches, ranging from heuristics (e.g., `k-fps`) to more complex algorithms (e.g., `k-means`). Note that only `mean` and `k-means` generate proxies that are not included in the given samples. We also note an analogy to associative memory models, interpreting imprinting as a memory update process following a covariance rule, as detailed in appendix A.3.

Normalization (`NORM`). The main reason for applying normalization is to allow each embedding and weight vector to contribute equally on the same scale. The modes we allow are no normalization (`none`), L^2 normalization (`L2`), and quantile normalization (`quantile`).

L2 normalization can be applied to embeddings before GEN via $\texttt{NORM}_{pre}$, to the generated weights via $\texttt{NORM}_{post}$, and to embeddings in inference via $\texttt{NORM}_{inf}$. In any case, the vector is L^2-normalized by dividing it by its Euclidean length $\|\cdot\|_2$.

quantile normalization (Amaratunga & Cabrera, 2001; Bolstad et al., 2003) can only be applied to generated weights. This non-linear operation distributes weights equally. Recall that if more than one class is contained in T ($c > 1$), GEN is performed for each class consecutively, and the reference distribution changes accordingly. In particular, for the first class there is no reference distribution to map to. This is different from Hosoda et al. (2024), where new weights are matched to the distribution of the original classifier weights of the FM. Since we do not consider the classes used for pre-training the FM and especially do not assume access to their last-layer weights, this is not possible in our scenario.

Aggregation (AGG). There are various ways to use the generated weights (proxies) per class during inference, especially when $k > 1$. We focus on two different modes, max and m-nearest neighbor (m-nn). The former, max, computes the inner product of the input embedding and the imprinted weights and outputs the class label with the maximum activation. The latter, m-nn, uses the class weights as keys and the embeddings as values, and chooses the final winning output class via the m-nearest neighbor algorithm. The m-nn voting is weighted by the inverse of the distances to their nearest neighbor, turning it into weighted majority voting.

Note that max is the same as 1-nn in the case of L2 for $\texttt{NORM}_{post}$, since for any fixed embedding vector v and variable proxy w, the argmin of $\|v - w\|^2 = \|v\|^2 - 2\langle v, w\rangle + \|w\|^2$, calculated by 1-nn, is the same as the argmax of the inner product $\langle v, w\rangle$ calculated in max.

3.2 Quantifying Neural Collapse

Neural collapse (NC) (Papyan et al., 2020) characterizes the state of the features produced by a classification neural network after training to near zero training loss. Namely, the learned embeddings of each class converge, i.e., *collapse*, to their class means. These globally centered class means and classifier weights form a simplex equiangular tight frame (ETF) – a collection of equal length and maximally equiangular vectors, that maximize the between-class variability. This results in an optimal linearly separable state for classification. In fig. 3 (left), we illustrate the collapse of a FM on its pre-training data. The newly arrived data T from a different dataset is distributed more unevenly across the embedding space (right).

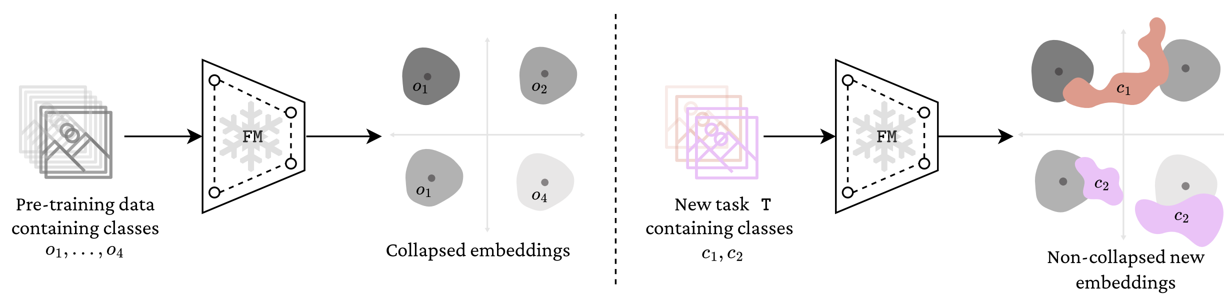


Figure 3: Left: The embeddings of the pre-training data, after being used to train the foundation model FM, show neural collapse, as each class $(o_1, \ldots, o_4)$ is evenly separated in space and accumulates around their respective class means. Right: For a novel task with classes c_1, c_2 (pink and brown) scatter around the collapsed pre-trained classes (gray).

Two important characteristics of NC are **variability collapse**, i.e., the within-class variability of the penultimate-layer embeddings collapses to zero, and **convergence to nearest-mean-classification**. We focus on variability collapse ($\mathcal{NC}_1$) as in Zhu et al. (2021):

$$\mathcal{NC}_1 = \frac{1}{C}\mathrm{trace}(\Sigma_W \Sigma_B^+), \tag{1}$$

where $\Sigma_W, \Sigma_B \in \mathbb{R}^{l \times l}$ are within- and between-class covariance matrices, respectively, l is the dimension of the embedding vector, C is the number of classes, and $^+$ symbolizes the pseudo-inverse. Based on the equation, an $\mathcal{NC}_1$ score closer to zero signifies a higher collapse. In contrast, an increase in multi-modality of data leads to a higher $\mathcal{NC}_1$ score (as analyzed in fig. 11). Note that this measurement is not independent of the embedding dimension l and the number of classes C. According to NC, imprinting the mean, as originally done by Qi et al. (2018), is best when $\mathcal{NC}_1$ is small. We claim that when the data is not fully collapsed (as is often the case in practice), the scale of $\mathcal{NC}_1$ could guide the proxy generation method, e.g., having multiple proxies $k > 1$ per class. We investigate this in section 5.3.

4 Experimental Setup

Foundation Models `FM`. We use `resnet18`, `resnet50` (He et al., 2016), `vit_b_16` (Dosovitskiy et al., 2021), and `swin_b` (Liu et al., 2021) as `FM`s, two CNN-based and two Transformer-based architectures. All four models are pre-trained on *ImageNet*-1K (ILSVRC 2012) (Deng et al., 2009). To generate the embeddings, we use PyTorch's *torchvision* models.

Tasks `T`. We analyze multi-class classification scenarios without separating base and new classes, instead focusing on all classes within a novel `T` at the same time. To investigate the effect of the number of samples given, we look at `n-shot` ($n \in \mathbb{N}$) scenarios. For that, we randomly pre-sample the training data of `T` to n samples per class – transitioning into the regime of few-shot learning.

To find out the best imprinting strategy within our `IMPRINT` framework, we focus on tasks `T` created from the datasets *MNIST* (Deng, 2012), *FashionMNIST* (Xiao et al., 2017), and *CIFAR-10* (Krizhevsky et al., 2009), each containing 10 classes. We mainly focus on the three `T` containing all ten classes. Furthermore, we look at smaller tasks only containing classes $\{0, 1, 2\}$, and the two tasks containing classes $\{1, 3, 5, 7, 9\}$ resp. $\{0, 2, 4, 6, 8\}$. This random selection of $3 \cdot 4 = 12$ tasks adds variation to our evaluations.

Neural Collapse. In the analysis of neural collapse (NC), we also look at the `FM`s' pre-training data (*ImageNet*). As its test set is not available, we use its validation set in $\mathcal{NC}_1$ computations. Furthermore, for *ImageNet*, we relabel data by combining multiple classes into one label to simulate multi-modal class distributions for an in-depth NC analysis. These tasks are called "d in 1", $d = 1, \ldots, 10$, each containing 10 different labels. More precisely, we take 100 random classes from *ImageNet* and sequentially map the first d to label 1, the second d to label 2, etc., until we reach 10 distinct labels. See fig. 4 for a simplified illustration.

Moreover, we construct a new out-of-distribution, non-collapsed dataset by merging all classes from four digit datasets: *MNIST*, *MNIST-M* (Ganin et al., 2016), *SVHN* (Netzer et al., 2011), and *USPS* (Hull, 1994). The resulting dataset, which we refer to as *CombiDigits*, exhibits reduced collapse due to the greater distributional diversity within each class. To ensure scale invariance in covariance-based NC measurements, all embeddings are L^2-normalized before computing $\mathcal{NC}_1$.

Scale. In total, we run approximately 300 000 experiments, varying across the imprinting components, foundation models, tasks, and seeds. This is feasible with minimal effort as imprinting is a highly efficient method that operates without relying on gradient descent or other non-linear optimization techniques.

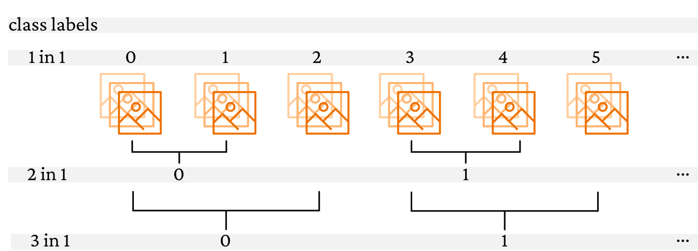


Figure 4: Combining multiple classes into one to create tasks with multi-modal class distributions. Simplified example for "d in 1", $d = 1, 2, 3$, with only six (instead of 100) original classes.

Evaluation. Throughout our experiments, the median accuracy on the test set for three different seeds is reported, if not otherwise specified. In sections 5.1 and 5.2, we investigate the imprinting performance by varying the `FM` (4) and `T` (12). We then sort the combinations by their final accuracy. There are $4 \cdot 12 = 48$ potentially different ranks for each of the combinations. We show the average rank, average accuracy, and statistical significance in ranking (dis-)agreements through critical difference (CD) diagrams as presented by Demšar (2006). The code used to generate these diagrams is inspired by Ismail Fawaz et al. (2019). In the CD diagrams, a thick horizontal line indicates a group of combinations that are not significantly different from each other in terms of ranking accuracy. We consider differences significant if $p < 0.05$.

In experiments with neural collapse (section 5.3), we investigate the four `FM`s on 100 random *ImageNet* tasks with remapped labels (as explained above) and the four tasks containing all of *MNIST*, *FashionMNIST*, *CIFAR-10*, and *CombiDigits*, respectively.

5 Results

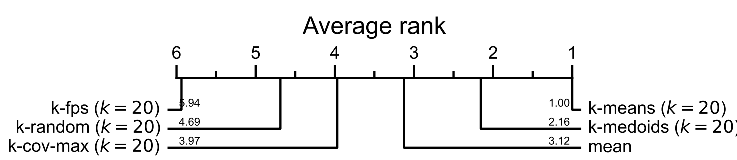
Our main experimental insights are: **1.** Our `IMPRINT` framework generalizes previous methods, and we find a new superior imprinting strategy (section 5.1). **2.** We show that our novel strategy is even beneficial in few-shot scenarios with as little as 50 samples per class (section 5.2). **3.** We identify a correlation between imprinting success utilizing multiple proxies and measures of neural collapse (section 5.3).

5.1 Best Imprinting Strategy

We provide a comparison between existing memory-constrained methods used for imprinting on foundation models in fig. 2, namely, the work by Qi et al. (2018); Hosoda et al. (2024); Janson et al. (2022), as well as a novel, best-performing configuration ("Ours") that results from `IMPRINT`. We investigate the impact of using the unconstrained `m-nn` aggregation on `all` data afterward. We focus on $k = 20$ and find that our method, consisting of `k-means` weight generation, `L2` normalizations, and `max` aggregation, outperforms all previously studied approaches by a margin of 6% on average with statistical significance. Next, we analyze each of the components of `IMPRINT` separately.

Table 2 & Figure 5: Benchmarking `GEN` mechanism for $k \leq 20$ across `FM`s and `T`s. Best `NORM` combination for each row used implicitly. `AGG` is fixed to `max`. CD diagram proves that `k-means` weight generation is significantly better than all other methods.

`GEN`	k	Avg. acc. %
`k-means`	20	**91.11**
`k-medoids`	20	87.32
`mean`	1	85.77
`k-cov-max`	20	84.15
`k-random`	20	82.76
`k-fps`	20	64.18



Weight Generation (`GEN`). To assess the impact of `GEN`, we first focus on the `max` aggregation and do not fix `NORM`, but simply show the run with the best `NORM` combination, if not otherwise specified. The `m-nn` aggregation and different values for `NORM` are analyzed later in this section.

Initially, we limit the number of generated proxies to $k \leq 20$. Results in fig. 5 show how `k-means`, using as many proxies as possible (in this case, 20) outperforms by 4% on average accuracy compared to all the other `GEN` methods. The CD diagram illustrates its statistical significance in ranking. Furthermore, while `k-medoids` with 20 proxies is computationally expensive, it is not much better than `mean`, and covariance maximization, furthest-point sampling and random selection show even weaker performances. We find similar results for $k \leq 5$, where `k-means` outperforms the other methods as well (see fig. A.1).

As the number of proxies k increases, `k-means` continues to be the best `GEN` method. An example for `resnet18` and *CIFAR-10* can be found in fig. 6. All methods converge towards the point of imprinting (saving) `all`

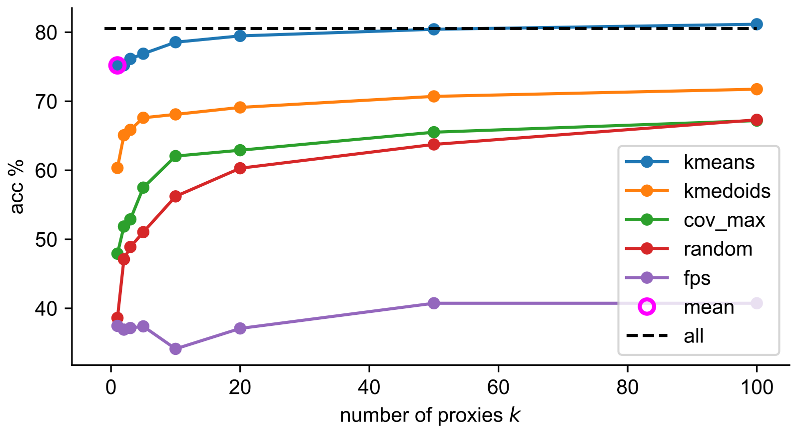


Figure 6: Benchmarking different `GEN` methods with `resnet18` on all of *CIFAR-10* shows superiority of `k-means` proxies. All combinations employ `L2` for all `NORM` and `max` as `AGG`.

data (k = all), even surpassing it in the case of `k-means`. Due to its superior performance, we mainly focus on `k-means` in the remainder of the analysis.

Normalization (`NORM`). We compare all the different `NORM` methods, focusing on `k-means` as `GEN`. For $k = 1$ and varying $\texttt{NORM}_{post}$ (while taking best values for $\texttt{NORM}_{pre}$ and $\texttt{NORM}_{inf}$ implicitly), fig. 7 shows that `L2` is the best choice for weight normalization. `quantile` and `none` normalization both perform worse.

Table 3 & Figure 7: Benchmarking $\texttt{NORM}_{post}$ mechanism across `FM`s and `T`s. The best $\texttt{NORM}_{pre}$ and $\texttt{NORM}_{inf}$ combinations for each row are used implicitly. `GEN` is fixed to `mean` (that is, $k = 1$) and `AGG` is fixed to `max`. The CD diagram shows the statistical significance of `L2` as the best weight normalization $\texttt{NORM}_{post}$.

$\texttt{NORM}_{post}$	Avg. acc. %
`L2`	**85.77**
`quantile`	84.22
`none`	82.21

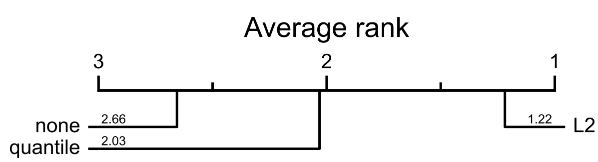


Table 4: Benchmarking $\texttt{NORM}_{pre}$ and $\texttt{NORM}_{inf}$ mechanisms across `FM`s and `T`s. $\texttt{NORM}_{post}$ is fixed to `L2`, `GEN` to `mean`, and `AGG` to `max`. No statistically significant differences were found.

$\texttt{NORM}_{pre}$	$\texttt{NORM}_{inf}$	Avg. acc.
`none`	`L2`	85.77
`none`	`none`	85.77
`L2`	`L2`	85.75

Keeping `L2` for $\texttt{NORM}_{post}$ fixed, we find no statistical differences between the different combinations of $\texttt{NORM}_{pre}$ and $\texttt{NORM}_{inf}$ (see table 4). For larger values of k, the differences among $\texttt{NORM}_{post}$ become even more pronounced. However, the performance of $\texttt{NORM}_{pre}$ and $\texttt{NORM}_{inf}$ remains statistically indifferent for `L2` weight normalization (see fig. A.2 for all combinations at once with $k = 1$, and fig. A.3 for $k = 20$).

We restrict all the subsequent experiments to using `L2` across all `NORM` following Qi et al. (2018). This combination of normalizations is chosen to specifically model cosine similarity within `max` aggregation.

Aggregation (`AGG`). In addition to `max`, we study the effect of using m-nearest neighbor (`m-nn`) as an aggregation method. Recall that `max` is a special case of `m-nn` when $m = 1$ (as $\texttt{NORM}_{post}$ is set to `L2`). We investigate different values for $m \in \{1, 3, 5, 20, 50\}$.

Table 5 & Figure 8: Benchmarking `AGG` mechanism across `FM`s and `T`s. `GEN` is fixed to `all` ($k = \text{all}$), that is, imprinting (saving) all data to weights. `L2` normalization is used for all `NORM`. The CD diagram shows statistical significance of `3-nn`, `5-nn`, and `20-nn` over `max` aggregation.

`AGG`	Avg. acc. %
`5-nn`	**94.24**
`3-nn`	94.09
`20-nn`	93.96
`1-nn`	93.50
`max`	93.50
`50-nn`	93.28

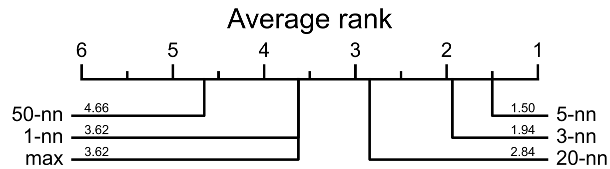
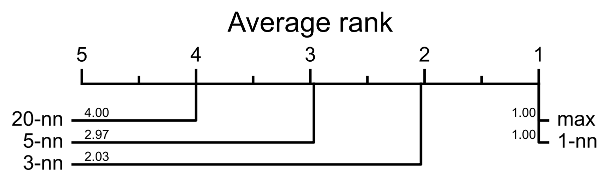


Table 6 & Figure 9: Benchmarking `AGG` mechanism across `FM`s and `T`s. `GEN` is fixed to `k-means` with $k = 20$. `L2` normalization is used for all `NORM`. The CD diagram shows that `max` is the best-performing aggregation method.

`AGG`	Avg. acc. %
`1-nn`	**91.11**
`max`	**91.11**
`3-nn`	90.58
`5-nn`	89.99
`20-nn`	86.53



When `all` data is imprinted, fig. 8 shows that using `m-nn` aggregation for $m \in \{3, 5, 20\}$ is slightly better than `max`. With $k = 20$ and `k-means` as `GEN`, `max` (=`1-nn`) aggregation becomes the top performing combination (see fig. 9). Furthermore, the reduction of proxies (from `all` (≈ 6000 per class) to $k = 20$) leads to a decrease in accuracy of around 3%.

Learned Weights and Multiple Proxies. Employing gradient-based methods – such as setting weights via the analytical least-squares initialization proposed by Harun & Kanan (2025), which uses data *and labels* from all classes *jointly* – can be seen as an upper bound for the unsupervised weight imprinting approach discussed here. We extend this method by combining it with `k-means`, forming `k-least-squares`, and observe in appendix A.5 that using multiple proxies per class instead of a single weight vector can still improve performance, particularly on datasets with high $\mathcal{NC}_1$.

5.2 Few-Shot Scenario

We analyze the `n-shot` scenario with our method (`k-means` as `GEN`, `L2` as `NORM`, and `max` as `AGG`). Furthermore, we focus on the large tasks `T` containing all ten classes of *MNIST*, *FashionMNIST*, resp. *CIFAR-10* at once. In this scenario, due to sampling only a few examples n, we average over five (instead of three) different seeds.

From the results in fig. 10, we find that as the number of samples n increases, `k-means` starts to outperform `mean` imprinting. The use of more proxies k further amplifies this performance gain. The shift occurs at roughly 50 samples per class for *MNIST* and *FashionMNIST*, while for *CIFAR-10*, $k > 1$ becomes prominently better at around 200 samples per class (see fig. A.4 for a display of all `FM`s focused on $10 \leq n \leq 400$).

5.3 Neural Collapse and Number of Proxies

Figure 11 depicts the neural collapse measurement $\mathcal{NC}_1$ (see eq. (1)) for the four tasks containing all of *MNIST*, *FashionMNIST*, *CIFAR-10*, resp. *CombiDigits*, as well as the 100 random *ImageNet* tasks with remapped labels as explained in section 4. We can see that *ImageNet* has a close-to-zero $\mathcal{NC}_1$ score, which increases linearly when adding more classes to each label (i.e., increasing multi-modality). As for other datasets, *CIFAR-10* is generally more collapsed according to its low value of $\mathcal{NC}_1$. We hypothesize that this is due to the similarity of its categories to those appearing in *ImageNet*. By design, the synthetic *CombiDigits*

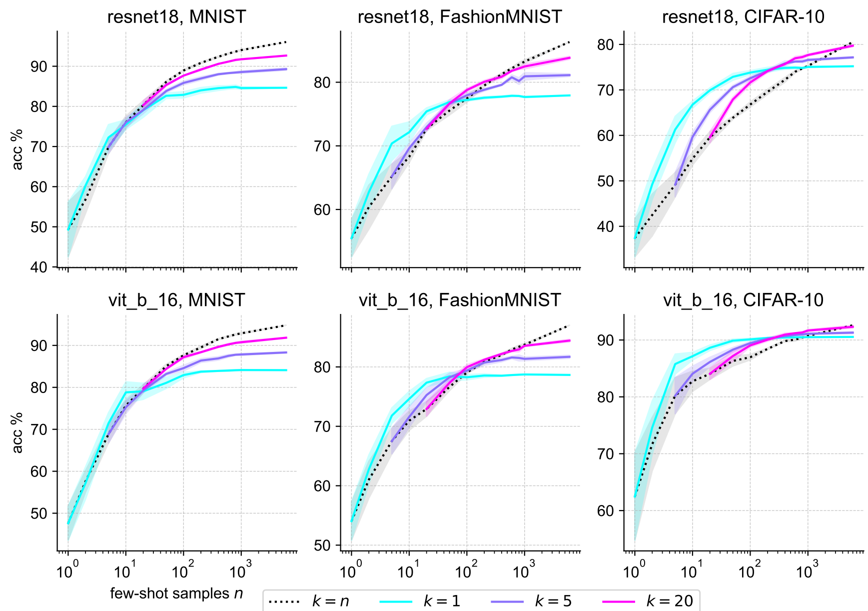


Figure 10: `k-means` with different values for k in `n-shot` scenarios. 95% CIs are shown in shaded colors. Other variables are fixed to our previously described best method of using `L2` normalizations and `max` as `AGG`. Note that only data for the meaningful case of $k \leq n$ is shown. For *MNIST* and *FashionMNIST*, `mean` is not the best strategy anymore starting at only roughly 50 samples.

dataset, introduced in section 4, has a very high $\mathcal{NC}_1$. Apart from that, $\mathcal{NC}_1$ of the *ImageNet* data for the Transformer-based architectures are much lower and therefore they are more collapsed compared to the CNN-inspired `FM`s. Architectural differences are further investigated in appendix A.4.

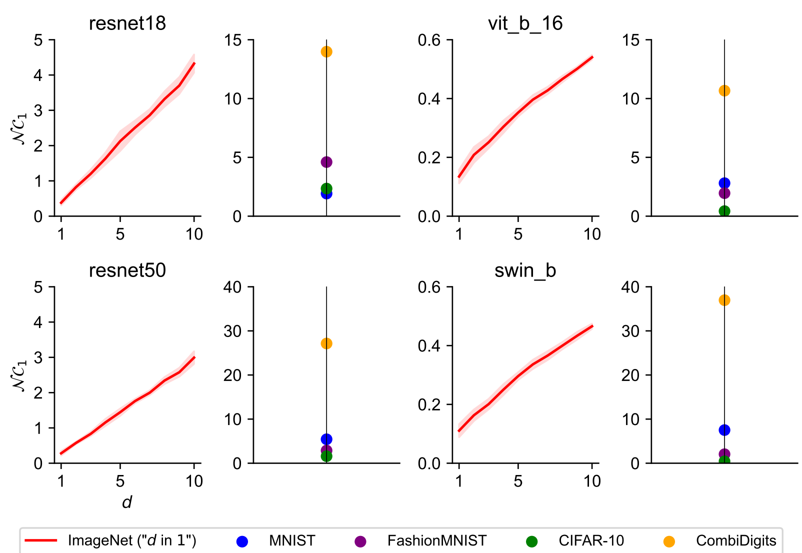


Figure 11: Average $\mathcal{NC}_1$ and 95% CIs of ten random *ImageNet* label remappings ("d in 1") for every $d = 1, \ldots, 10$. The $\mathcal{NC}_1$ for the tasks containing all of *MNIST*, *FashionMNIST*, *CIFAR-10*, resp. *CombiDigits* at once are depicted as dots. When compared to $\mathcal{NC}_1$ on the pre-training set (*ImageNet*), these sets are less collapsed on the Transformer-based architectures. A clear linear relationship across d can be inferred for all `FM`s, i.e., increased multi-modality implies less collapse. *CombiDigits* achieves a high $\mathcal{NC}_1$ score across all `FM`s.

For the same data, fig. 12 depicts accuracy over a varying number of proxies k inferred from `k-means`. A prominent peak at $k = d$ can be inferred for every `FM` and reflects that d class proxies lead to the best result for d-modal class distributions. The fact that *CIFAR-10* has the lowest $\mathcal{NC}_1$ (see fig. 11) is reflected by flat curves over k. This confirms that the $\mathcal{NC}_1$ score is a significant indicator of multi-modality. Namely, a higher $\mathcal{NC}_1$ score indicates the benefits of using a higher number of proxies. This effect is particularly evident in the imprinting results on *CombiDigits*, where using $k = 3$ proxies per class leads to a substantial accuracy improvement.

Furthermore, increasing k for the *ImageNet* sets has a much larger effect on the CNN-based `FM`s. We argue that this is because of their higher values of $\mathcal{NC}_1$, indicating less neural collapse. In appendix A.4, we analyze architectural differences and training setups to explain these variations.

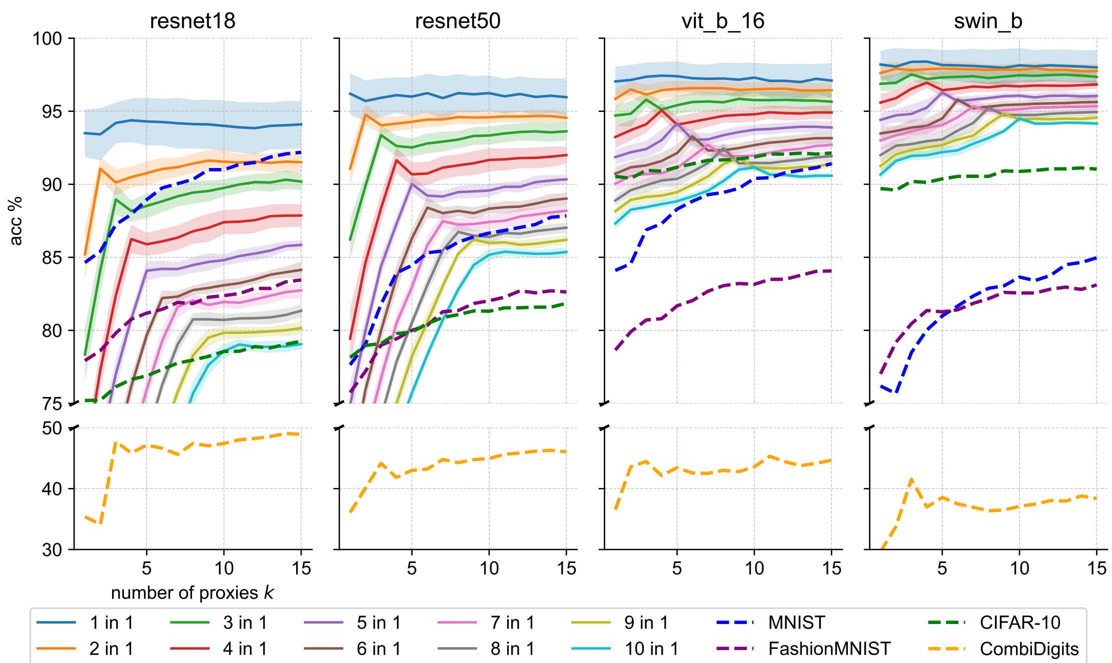


Figure 12: Averaged accuracy of ten random *ImageNet* label remappings ("d in 1") for every $d = 1, \ldots, 10$ over number of proxies k used in `k-means`. 95% CIs are shown in shaded colors. `L2` for all `NORM` and `max` as aggregation are used. Accuracies of the tasks containing all of *MNIST*, *FashionMNIST*, *CIFAR-10*, resp. *CombiDigits* at once are shown in dotted lines. In all four plots, peaks in accuracy at $k = d$ can be inferred. This confirms the connection between the effect of using multiple proxies and the collapse of the data.

6 Conclusion

We present a new framework, `IMPRINT`, to analyze three main components relevant to weight imprinting, namely, weight generation, normalization, and aggregation. Within `IMPRINT`, state-of-the-art imprinting strategies become special cases. This allows for a comprehensive analysis of different approaches through systematic experiments and leads us to generalize to a new, best-performing imprinting variant. That is, using `k-means` weight generation with `L2` normalizations and `max` aggregation, which outperforms all previously studied methods (see fig. 2).

`k-means` generates better weights than `mean`. In particular, we find that the `mean` weight generation (`GEN`) method, despite its prominence in previous work, falls short compared to `k-means` – even when the number of proxies k is very small. Remarkably, with as little as 50 samples per class, `k-means` can already outperform the original imprinting method proposed by Qi et al. (2018), highlighting its advantage in few-shot scenarios.

`L2` weight normalization is essential for strong performance. The `max` aggregation directly scales with the magnitude of the weights. Normalization ($\texttt{NORM}_{post}$) ensures that all class weights contribute equally to the output, making it crucial for a well-performing imprinting method. Nearest neighbor (`1-nn`) aggregation is not as affected by the lack of normalization, since it uses Euclidean distance. Although still part of common procedure, normalizations for embeddings ($\texttt{NORM}_{pre}$ and $\texttt{NORM}_{inf}$) appear to have minimal impact on performance.

With `max` aggregation, there is no need to store `all` data. While nearest neighbor (`m-nn`) aggregation (`AGG`) performs well when all data is saved (e.g., when there are no storage constraints), `max` aggregation with limited number of representative proxies (e.g., through k-means) is an efficient alternative without a substantial loss in performance.

Neural collapse proves the efficacy of imprinting. During training, the last-layer weights of a `FM` collapse to their respective class means. This proves the success of `mean` imprinting on known classes. New, out-of-distribution data, however, often shows less collapse, making it beneficial to imprint more than one proxy. This is particularly evident on the synthetic, multi-modal, and out-of-distribution dataset *CombiDigits*, where multiple proxies per class substantially improve performance.

Limitations. Our experiments are limited to foundation models for image classification and do not cover other data modalities such as audio, text, or sensor inputs. While `IMPRINT` is agnostic to modality, empirical validation of our results outside of vision is needed. Although imprinting alone provides an efficient solution to transfer learning, we do not investigate the benefit of combining it with optimization methods such as gradient-based learning.

Future Work. The usage of both weight and activation sparsity as Shen et al. (2023) could change the within- and between-class variability in favor of using a higher number of proxies. Besides only using the penultimate layer embeddings for generating the classifier weights, an interesting area of study could be extracting embeddings from previous layers of the `FM` for this purpose. Recently, the study by Marczak et al. (2025) showed that adding a multi-layer perceptron projector between the penultimate and classification layers results in representations that are more transferable. Another avenue of research is to thoroughly analyze imprinting the weights of other layers as well (Siam et al., 2019).

References

Dhammika Amaratunga and Javier Cabrera. Analysis of data from viral dna microchips. *Journal of the American Statistical Association*, 96(456):1161–1170, 2001.

James A Anderson. A simple neural network generating an interactive memory. *Mathematical biosciences*, 14(3-4): 197–220, 1972.

James A Anderson, Jack W Silverstein, Stephen A Ritz, and Randall S Jones. Distinctive features, categorical perception, and probability learning: Some applications of a neural model. *Psychological review*, 84(5):413, 1977.

Maximilian Beck, Korbinian Pöppel, Markus Spanring, Andreas Auer, Oleksandra Prudnikova, Michael Kopp, Günter Klambauer, Johannes Brandstetter, and Sepp Hochreiter. xlstm: Extended long short-term memory. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL `http://papers.nips.cc/paper_files/paper/2024/hash/c2ce2f2701c10a2b2f2ea0bfa43cfaa3-Abstract-Conference.html`.

Yoshua Bengio. Deep learning of representations for unsupervised and transfer learning. In *Proceedings of ICML workshop on unsupervised and transfer learning*, pp. 17–36. JMLR Workshop and Conference Proceedings, 2012.

Benjamin M Bolstad, Rafael A Irizarry, Magnus Åstrand, and Terence P. Speed. A comparison of normalization methods for high density oligonucleotide array data based on variance and bias. *Bioinformatics*, 19(2):185–193, 2003.

Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

Paulino Cristovao, Hidemoto Nakada, Yusuke Tanimura, and Hideki Asoh. Few shot model based on weight imprinting with multiple projection head. In *2022 16th International Conference on Ubiquitous Information Management and Communication (IMCOM)*, pp. 1–7. IEEE, 2022.

Peter Dayan and David J Willshaw. Optimising synaptic learning rules in linear associative memories. *Biological cybernetics*, 65(4):253–265, 1991.

Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine learning research*, 7:1–30, 2006.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255, 2009.

Li Deng. The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE signal processing magazine*, 29(6):141–142, 2012.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL-HLT*, pp. 4171–4186, 2019.

Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition. In *ICML*, pp. 647–655, 2014.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

Tomer Galanti, András György, and Marcus Hutter. On the role of neural collapse in transfer learning. In *International Conference on Learning Representations*, 2022.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario March, and Victor Lempitsky. Domain-adversarial training of neural networks. *Journal of machine learning research*, 17(59):1–35, 2016.

Spyros Gidaris and Nikos Komodakis. Dynamic few-shot visual learning without forgetting. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4367–4375, 2018.

X. Y. Han, Vardan Papyan, and David L. Donoho. Neural collapse under MSE loss: Proximity to and dynamics on the central path. In *The Tenth International Conference on Learning Representations, ICLR*, 2022.

Md Yousuf Harun and Christopher Kanan. A good start matters: Enhancing continual learning with data-driven weight initialization. *arXiv preprint arXiv:2503.06385*, 2025.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Kazufumi Hosoda, Keigo Nishida, Shigeto Seno, Tomohiro Mashita, Hideki Kashioka, and Izumi Ohzawa. A single fast hebbian-like process enabling one-shot class addition in deep neural networks without backbone modification. *Frontiers in Neuroscience*, 18:1344114, 2024.

Minyoung Huh, Pulkit Agrawal, and Alexei A. Efros. What makes ImageNet good for transfer learning?, 2016. arXiv:1608.08614.

Jonathan J. Hull. A database for handwritten text recognition research. *IEEE Transactions on pattern analysis and machine intelligence*, 16(5):550–554, 1994.

Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Deep learning for time series classification: a review. *Data mining and knowledge discovery*, 33(4):917–963, 2019.

Paul Janson, Wenxuan Zhang, Rahaf Aljundi, and Mohamed Elhoseiny. A simple baseline that questions the use of pretrained-models in continual learning. In *NeurIPS 2022 Workshop on Distribution Shifts: Connecting Methods and Applications*, 2022.

Mina Khan, P Srivatsa, Advait Rane, Shriram Chenniappa, Asadali Hazariwala, and Pattie Maes. Personalizing pre-trained models. *arXiv preprint arXiv:2106.01499*, 2021.

Teuvo Kohonen. Correlation matrix memories. *IEEE transactions on computers*, 100(4):353–359, 2009.

Simon Kornblith, Jonathon Shlens, and Quoc V. Le. Do Better ImageNet Models Transfer Better? In *CVPR*, pp. 2656–2666, Long Beach, CA, USA, 2019. IEEE.

Vignesh Kothapalli. Neural collapse: A review on modelling principles and generalization. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856.

Alex Krizhevsky, Geoffrey Hinton, et al. CIFAR-10. `https://www.cs.toronto.edu/~kriz/cifar.html`, 2009.

Yiting Li, Haiyue Zhu, Jun Ma, Sichao Tian, Chek Sing Teo, Cheng Xiang, Prahlad Vadakkepa, and Tong Heng Lee. Classification weight imprinting for data efficient object detection. In *2021 IEEE 30th International Symposium on Industrial Electronics (ISIE)*, pp. 1–5. IEEE, 2021.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 10012–10022, 2021.

Daniel Marczak, Sebastian Cygert, Tomasz Trzciński, and Bartłomiej Twardowski. Revisiting supervision for continual representation learning. In *European Conference on Computer Vision*, pp. 181–197. Springer, 2025.

Thomas Mensink, Jakob Verbeek, Florent Perronnin, and Gabriela Csurka. Distance-based image classification: Generalizing to new classes at near-zero cost. *IEEE transactions on pattern analysis and machine intelligence*, 35 (11):2624–2637, 2013.

Yair Movshovitz-Attias, Alexander Toshev, Thomas K Leung, Sergey Ioffe, and Saurabh Singh. No fuss distance metric learning using proxies. In *Proceedings of the IEEE international conference on computer vision*, pp. 360–368, 2017.

Kaoru Nakano. Associatron-a model of associative memory. *IEEE Transactions on systems, man, and cybernetics*, pp. 380–388, 2007.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Baolin Wu, Andrew Y Ng, et al. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, volume 2011, pp. 4. Granada, 2011.

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

Nikolaos Passalis, Alexandros Iosifidis, Moncef Gabbouj, and Anastasios Tefas. Hypersphere-based weight imprinting for few-shot learning on embedded devices. *IEEE Transactions on Neural Networks and Learning Systems*, 32(2): 925–930, 2020.

Ameya Prabhu, Zhipeng Cai, Puneet Dokania, Philip Torr, Vladlen Koltun, and Ozan Sener. Online continual learning without the storage constraint. *arXiv preprint arXiv:2305.09253*, 2023.

Hang Qi, Matthew Brown, and David G Lowe. Low-shot learning with imprinted weights. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5822–5830, 2018.

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

Terrence J Sejnowski. Storing covariance with nonlinearly interacting neurons. *Journal of mathematical biology*, 4(4): 303–321, 1977.

Yang Shen, Sanjoy Dasgupta, and Saket Navlakha. Reducing catastrophic forgetting with associative learning: a lesson from fruit flies. *Neural Computation*, 35(11):1797–1819, 2023.

Mennatullah Siam, Boris N Oreshkin, and Martin Jagersand. Amp: Adaptive masked proxies for few-shot segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5249–5258, 2019.

Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. *Advances in neural information processing systems*, 30, 2017.

Tom Tirer, Haoxiang Huang, and Jonathan Niles-Weed. Perturbation analysis of neural collapse. In *International Conference on Machine Learning*, pp. 34301–34329. PMLR, 2023.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Dingtian Yan, Jitao Huang, Hai Sun, and Fuqiang Ding. Few-shot object detection with weight imprinting. *Cognitive Computation*, 15(5):1725–1735, 2023.

Hong-Ming Yang, Xu-Yao Zhang, Fei Yin, and Cheng-Lin Liu. Robust classification with convolutional prototype learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3474–3482, 2018.

Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? *Advances in neural information processing systems*, 27, 2014.

Lu Yu, Bartlomiej Twardowski, Xialei Liu, Luis Herranz, Kai Wang, Yongmei Cheng, Shangling Jui, and Joost van de Weijer. Semantic drift compensation for class-incremental learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 6982–6991, 2020.

Jianxing Zhang, Pengcheng Xi, Ashkan Ebadi, Hilda Azimi, Stéphane Tremblay, and Alexander Wong. Covid-19 detection from chest x-ray images using imprinted weights approach. *arXiv preprint arXiv:2105.01710*, 2021.

Haiyue Zhu, Xiong Li, Wenjie Chen, Xiaocong Li, Jun Ma, Chek Sing Teo, Tat Joo Teo, and Wei Lin. Weight imprinting classification-based force grasping with a variable-stiffness robotic gripper. *IEEE Transactions on Automation Science and Engineering*, 19(2):969–981, 2022.

Zhihui Zhu, Tianyu Ding, Jinxin Zhou, Xiao Li, Chong You, Jeremias Sulam, and Qing Qu. A geometric analysis of neural collapse with unconstrained features. *Advances in Neural Information Processing Systems*, 34:29820–29834, 2021.

A Appendix

A.1 Additional Results

We provide additional tables and critical difference (CD) diagrams that expand on the results of section 5.

Table A.1 & Figure A.1: Benchmarking `GEN` mechanism for $k \leq 5$ across `FM`s and `T`s. Best `NORM` combination for each row is used implicitly. `AGG` is fixed to `max`. CD diagram depicts statistical significance of `k-means` as `GEN`. See fig. 5 for $k \leq 20$.

`GEN`	k	Avg. acc. %
`k-means`	5	**88.79**
`mean`	1	85.77
`k-medoids`	5	84.84
`k-cov-max`	5	82.47
`k-random`	5	75.83
`k-fps`	5	62.10

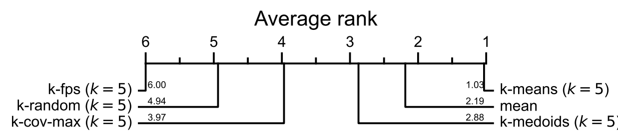


Table A.2 & Figure A.2: Benchmarking `NORM` across `FM`s and `T`s shows crucial effect of `L2` normalization. `GEN` is fixed to `mean` and `AGG` to `max`. CD diagram depicting statistical significance of `L2` for $\texttt{NORM}_{post}$. Combinations are listed as "$\texttt{NORM}_{inf}$ & $\texttt{NORM}_{pre}$ & $\texttt{NORM}_{post}$".

$\texttt{NORM}_{inf}$	$\texttt{NORM}_{pre}$	$\texttt{NORM}_{post}$	Avg. acc. %
`L2`	`none`	`L2`	**85.77**
`none`	`none`	`L2`	**85.77**
`L2`	`L2`	`L2`	**85.75**
`L2`	`none`	`quantile`	84.22
`none`	`none`	`quantile`	84.22
`L2`	`L2`	`quantile`	84.16
`L2`	`L2`	`none`	82.21
`L2`	`none`	`none`	62.47
`none`	`none`	`none`	62.47

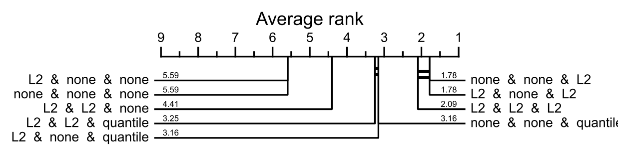
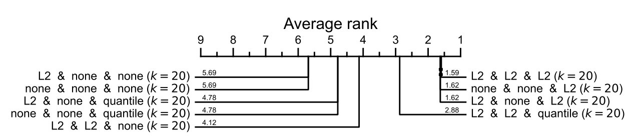


Table A.3 & Figure A.3: Benchmarking `NORM` across `FM`s and `T`s. `GEN` is fixed to `k-means` with $k = 20$ and `AGG` to `max`. CD diagram depicting statistical significance of `L2` for $\texttt{NORM}_{post}$. Combinations are listed as "$\texttt{NORM}_{inf}$ & $\texttt{NORM}_{pre}$ & $\texttt{NORM}_{post}$."

$\texttt{NORM}_{inf}$	$\texttt{NORM}_{pre}$	$\texttt{NORM}_{post}$	Avg. acc. %
`L2`	`L2`	`L2`	**91.11**
`L2`	`none`	`L2`	**91.08**
`none`	`none`	`L2`	**91.09**
`L2`	`L2`	`quantile`	90.70
`L2`	`L2`	`none`	89.68
`L2`	`none`	`quantile`	78.51
`none`	`none`	`quantile`	78.51
`L2`	`none`	`none`	68.18
`none`	`none`	`none`	68.18



A.2 Datasets

We briefly describe the datasets used in our experiments.

***ImageNet* (Deng et al., 2009).** We use the ILSVRC 2012 version (commonly called *ImageNet*-1K) containing 1 000 classes and 1.2M training images. Since the test set is not publicly available, we use the

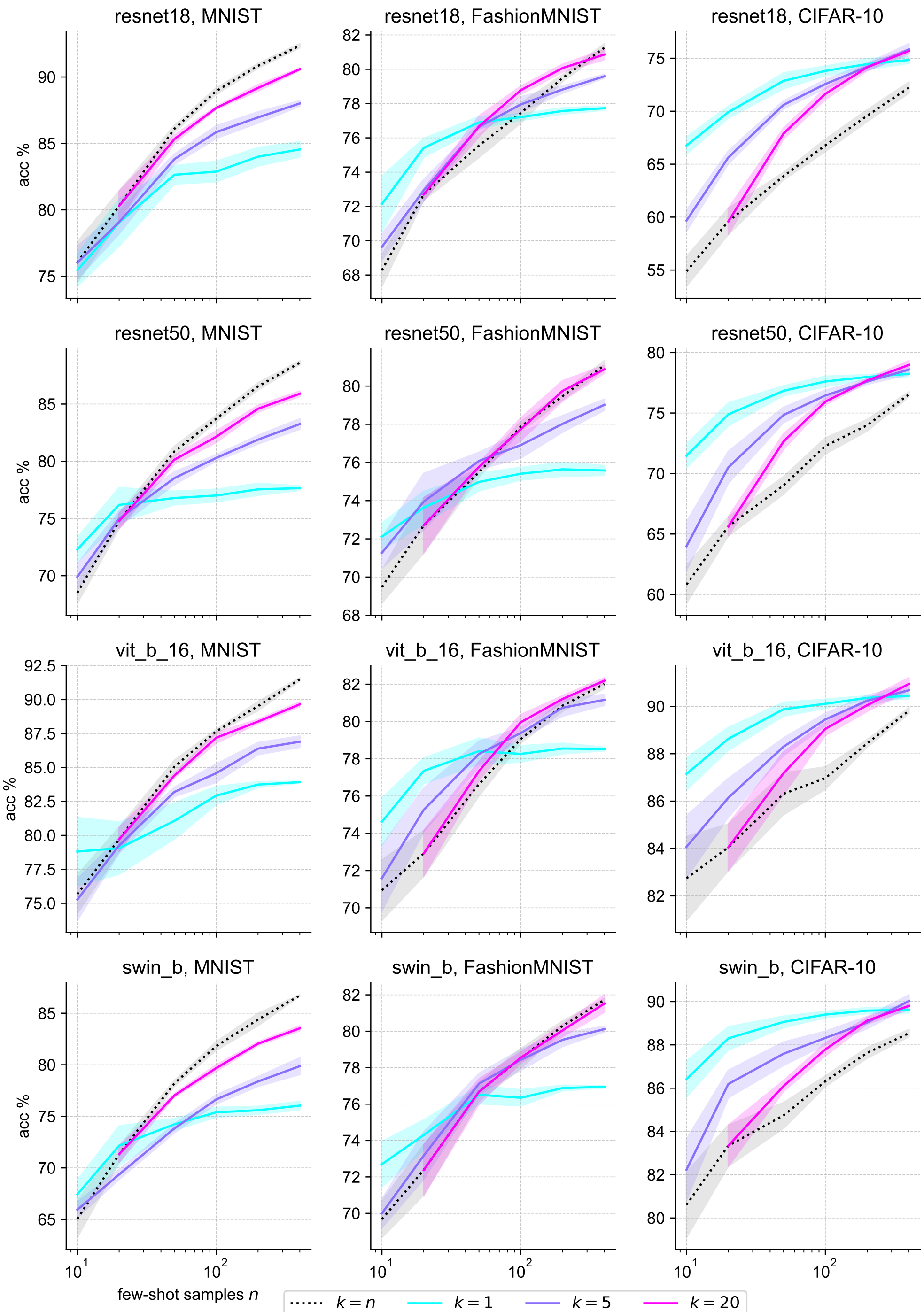


Figure A.4: `k-means` with different values for k in `n-shot` scenarios with focus on $10 \leq n \leq 400$. 95% confidence intervals shown in shaded colors. Other variables are fixed to our previously described best method of using `L2` for `NORM` and `max` as `AGG`. Note that only data for the meaningful case of $k \leq n$ is shown. It can be inferred that, across `FMs`, for *MNIST* and *FashionMNIST*, `mean` is not the best strategy anymore starting at only roughly 50 samples. See fig. 10 for more values of n.

validation set (50 000 images) as a stand-in. For neural collapse investigations, we construct tasks by randomly grouping d classes into one label, producing "d in 1" tasks as explained in section 4.

***MNIST* (Deng, 2012).** A benchmark dataset of handwritten digits (0–9), consisting of 60 000 training and 10 000 test grayscale images of size 28×28.

***FashionMNIST* (Xiao et al., 2017).** A drop-in replacement for *MNIST* with the same format and number of samples, containing grayscale images of fashion items across 10 classes.

***CIFAR-10* (Krizhevsky et al., 2009).** A dataset of 32×32 RGB images covering 10 object classes, with 50 000 training and 10 000 test samples.

***MNIST-M* (Ganin et al., 2016), *SVHN* (Netzer et al., 2011), *USPS* (Hull, 1994).** Digit classification datasets each containing digits 0-9 with domain-specific visual characteristics. *MNIST-M* applies color and texture transformations to MNIST digits, yielding 60 000 training and 10 000 RGB images of size 28×28. *SVHN* consists of digit crops from house numbers in Google Street View, totaling 73 257 training and 26 032 test images of 32×32 in RGB. *USPS* contains scanned and normalized handwritten digits in 16×16 grayscale, split into 7 219 training and 2 007 test images.

***CombiDigits* (Ours).** A synthetic dataset constructed by merging all classes from *MNIST*, *MNIST-M*, *SVHN*, and *USPS*. Each class label corresponds to a digit (0–9) and fig. A.5 shows example images of class "6". The dataset includes significant visual heterogeneity across sources and thus simulates multi-modal, non-collapsed class distributions.

Figure A.5: Twenty sample images from class "6" of the *CombiDigits* dataset. Each row corresponds to one of the four source datasets: *MNIST*, *MNIST-M*, *SVHN*, and *USPS* (top to bottom), illustrating the intra-class heterogeneity across domains.

A.3 Imprinting as Memory

We revisit the classical idea of Bidirectional Associative Memories (BAMs) (Kohonen, 2009; Anderson, 1972; Nakano, 2007; Anderson et al., 1977) and the associated update rule (Sejnowski, 1977; Dayan & Willshaw, 1991) for storing key-value pairs as in

$$\mathbf{W} \leftarrow \mathbf{W} + \mathbf{v}\mathbf{k}^\top,$$

where $\mathbf{W} \in \mathbb{R}^{l \times l}$ is a matrix and $\mathbf{k}, \mathbf{v} \in \mathbb{R}^d$ are key and value vectors, respectively, to be stored therein.

Throughout this work, imprinting can be interpreted as inserting such key-value associations into the classifier weights. More precisely, the generation (`GEN`) component extends the linear classification head by setting $\mathbf{v}$ as a one-hot vector representing the class, and $\mathbf{k}$ as the corresponding proxy previously computed. While we focus on this simple linear setting, imprinting is not limited to it, as discussed in section 2.

Notably, this form of direct memory update has seen renewed attention in modern architectures beyond standard query-key-value attention. In particular, the xLSTM model (Beck et al., 2024) implements this mechanism within its mLSTM memory blocks, where the matrix memory cell is updated by gated key-value associations, closely following this classical covariance rule, indicating a broader resurgence of associative memory principles in contemporary sequence modeling.

A.4 Differences between Foundation Models

While an in-depth comparison of foundation models is beyond the scope of this work, we believe it is important to highlight key observations. In particular, fig. 11 shows significantly lower $\mathcal{NC}_1$ scores for `vit_b_16` and `swin_b` on their pre-training *ImageNet* data compared to the `resnet` models. We hypothesize that this difference is primarily due to model size and training regimes. The Transformer-based architectures (`vit_b_16` and `swin_b`) have a considerably higher parameter count ($\approx$ 87M) than the `resnet` models (11.7M and 25.6M, respectively). Additionally, `vit_b_16` and `swin_b` were trained for three times as many epochs (300 vs. 90) while using a substantially lower learning rate (0.003 and 0.01 vs. 0.1). Notably, the embedding dimensions of these models are comparable, meaning that the observed differences in $\mathcal{NC}_1$ scores cannot be attributed to differences in representation dimensionality. Instead, we argue that the combination of larger model size, extended training duration, and lower learning rates likely contributes to greater overfitting, leading to more pronounced collapse. As discussed in section 5.3, this enables the Transformer-based `FMs` to handle the *ImageNet* tasks with remapped labels more effectively and to achieve significantly better performance on the similarly distributed *CIFAR-10* dataset.

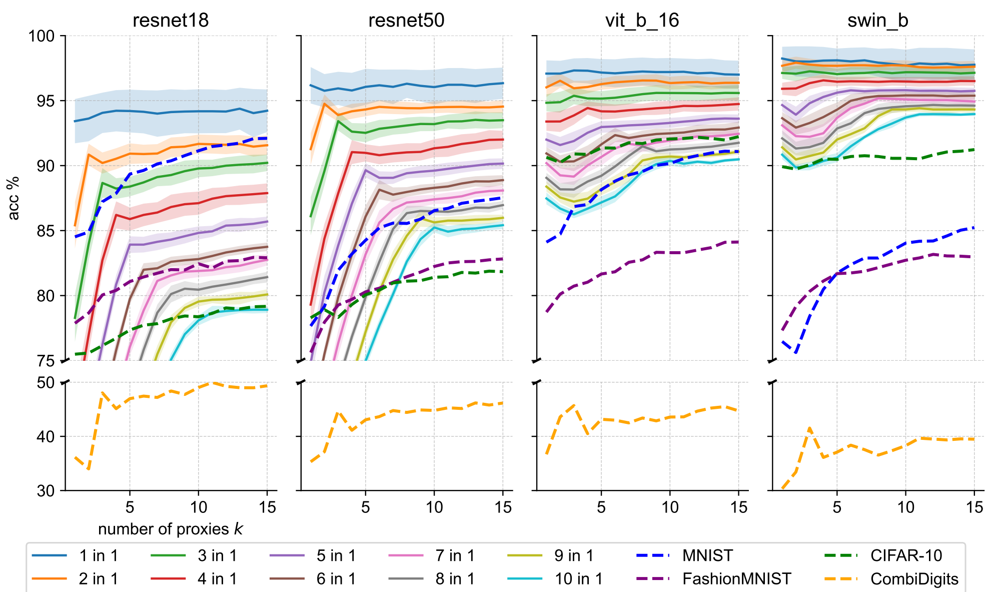


Figure A.6: Averaged accuracy of ten random *ImageNet* label remappings ("d in 1") for every $d = 1, \ldots, 10$ over number of proxies k used for `k-means` in `GEN`. 95% confidence intervals are shown in shaded colors. We set $\texttt{NORM}_{post}$ and $\texttt{NORM}_{inf}$ to `L2`, and $\texttt{NORM}_{pre}$ to `none`. Accuracies of the tasks containing all of *MNIST*, *FashionMNIST*, and *CIFAR-10* at once are shown in dotted lines. Besides the prominent peaks in accuracy at $k = d$ (as already observed in fig. 12), a consistent dip between $k = 1$ and $k = d$ appears in Transformer-based models.

Figure A.6, similar to fig. 12, illustrates the impact of varying the number of proxies on imprinting accuracy across different foundation models (`FMs`). The key difference in this figure is the use of `none` for $\texttt{NORM}_{pre}$ instead of `L2`. This seemingly minor change reveals a striking contrast between CNN- and Transformer-based architectures: a distinct and consistent dip between $k = 1$ and $k = d$ appears in Transformer-based models, whereas this dip is absent in fig. 12, where `L2` is used as $\texttt{NORM}_{pre}$, and does not occur at all in the `resnet` models. We hypothesize that this difference arises from the distinct embedding distributions of CNN and Transformer architectures (see, e.g., Hosoda et al. (2024, Figure S2)).

A.5 Learned Weights and Multiple Proxies

Harun & Kanan (2025) recently studied the initialization of classifier weights for novel categories in a Continual Learning (CL) setting using the least squares algorithm, comparing it to random initialization and class mean imprinting across various loss functions.

In contrast, our work centers on imprinting-based approaches, which avoid gradient-based optimization and cross-class statistics, operating instead on a per-class basis with immediate availability of data. Nonetheless, we include the least-squares method as a supervised oracle baseline – explicitly not an imprinting method – fine-tuned for classification accuracy.

We evaluate its performance on the same tasks as defined in section 4. Additionally, we introduce a multi-proxy extension, `k-least-squares`, which combines least squares with `k-means` clustering. This generalization allows each class to be represented by multiple weight vectors. As in the case of weight generation `GEN` in imprinting (where `k-means` outperforms `mean`), we find that using multiple proxies per class improves performance in settings with high $\mathcal{NC}_1$ as well, indicating that the benefits of multi-prototype representations persist even in this non-imprinting, supervised context.

All experiments use no normalization (`none` as `NORM`), based on ablations confirming that additional normalization impairs performance. This matches expectations, since least squares outputs are already calibrated, and normalization distorts them.

Least Squares Weights. For all data contained in task `T`, define $\mathbf{Z} \in \mathbb{R}^{l \times N}$ as the collection of all feature vectors of the N training samples in `T`, i.e., $\mathbf{z}_{i,c} \in \mathbb{R}^l$ is the feature vector of the i-th sample in the c-th class. From this, we can define the feature global mean $\boldsymbol{\mu}_G = \mathbb{E}_{i,c}[\mathbf{z}_{i,c}]$, feature class means $\boldsymbol{\mu}_c = \mathbb{E}_i[\mathbf{z}_{i,c}]$ for $c = 1, \ldots, C$, and feature within-class covariance $\boldsymbol{\Sigma}_W = \mathbb{E}_{i,c}[(\mathbf{z}_{i,c} - \boldsymbol{\mu}_c)(\mathbf{z}_{i,c} - \boldsymbol{\mu}_c)^\top]$. Finally, we define the total covariance matrix $\boldsymbol{\Sigma}_T$ and class-means matrix $\mathbf{M}$ as

$$\mathbf{M} = [\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_C] \in \mathbb{R}^{l \times C}, \qquad \boldsymbol{\Sigma}_T = \mathbb{E}_{i,c}[(\mathbf{z}_{i,c} - \boldsymbol{\mu}_G)(\mathbf{z}_{i,c} - \boldsymbol{\mu}_G)^\top] \in \mathbb{R}^{l \times l}.$$

From these feature statistics, we can obtain the `least-squares` weights via

$$\mathbf{W}_{LS} = \frac{1}{C}\mathbf{M}^\top(\boldsymbol{\Sigma}_T + \boldsymbol{\mu}_G\boldsymbol{\mu}_G^\top + \lambda\mathbf{I})^{-1}. \tag{A.1}$$

Here, $\mathbf{I}$ is the identity matrix and λ is the weight decay. We set λ to match the value used during the original training of each respective model. Table A.4 lists the specific λ values used for all models considered in our experiments.

Table A.4: Weight decay values λ used for each model.

Model	λ
`resnet18`, `resnet50` (He et al., 2016)	0.0001
`vit_b_16` (Dosovitskiy et al., 2021)	0.1
`swin_b` (Liu et al., 2021)	0.05

Equation (A.1) reveals that `least-squares` shares structural similarities with `mean` imprinting, but applies a more sophisticated normalization scheme that includes both scaling and rotation. Crucially, it relies on cross-class statistics computed across the entire dataset, and thus is incompatible with imprinting scenarios, which operate by directly constructing class weights from the data of a single class without supervision or access to other classes.

Combining `k-means` and `least-squares` into `k-least-squares`. To allow for multiple proxies per class, we propose `k-least-squares`, a generalization of standard least squares (`least-squares`) that integrates clustering. Instead of computing a single weight vector per class using all class samples, we first partition each class's feature set $\mathbf{Z}_c$ into k clusters using k-means. Each cluster is treated as a separate proxy class,

Algorithm 1 `k-least-squares`

Input: Class data $\{\mathbf{Z}_c\}_{c=1}^{C}$, number of proxies k
Output: Weights $\{\mathbf{W}_c\}_{c=1}^{C}$ with shape $[k, l]$ per class

if $k = 1$ **then**
  return standard least squares weights $\mathbf{W}_{LS}$ for each class (see eq. (A.1))
end if
for each class c **do**
  Cluster $\mathbf{Z}_c$ into k clusters via k-means
  Assign each cluster j to proxy class (c, j)
  Collect proxy samples $\{\mathbf{Z}_{(c,j)}\}_{j=1}^{k}$
end for
Compute least squares weights $\mathbf{w}_{(c,j)}$ jointly for all proxy classes (c, j)
Assemble $\mathbf{W}_c = [\mathbf{w}_{(c,1)}, \ldots, \mathbf{w}_{(c,k)}]$ for each original class c
return $\{\mathbf{W}_c\}_{c=1}^{C}$

effectively expanding the classification task to $C \times k$ proxy classes. We then solve a single regularized least squares problem over all proxy classes at once, assigning a distinct target vector to each proxy. The resulting weights are grouped by their original class to yield k weight vectors per class. The complete procedure is given in algorithm 1.

The results in fig. A.7 extend previous findings from `mean` and `k-means` for the CNN-based `FM`s, as accuracy increases with increasing proxy count k, peaks at $k = d$, and reveals a distinct dip between $k = 1$ and $k = d$, confirming the weakness of using only one proxy per class. For Transformer-based `FM`s, by contrast, accuracy is often highest at $k = 1$, improves slightly near $k = d$, but lacks a pronounced peak. Beyond $k > d$, accuracy generally declines for `k-least-squares`, indicating diminishing returns from excessive proxy splitting. In contrast, `k-means GEN` maintains stable performance even as k grows, demonstrating robustness to increasing proxy counts.

Comparing Results. We now compare the performance of `k-means` and `k-least-squares` as proxy generation methods, noting again that `k-least-squares` is not an imprinting scheme and is significantly less efficient than the imprinting methods presented in section 3.1.

Upon proper comparison of the best performing `k-means` and `k-least-squares` configurations, as depicted in fig. A.8, we observe that on these synthetic *ImageNet* tasks, `k-least-squares` does not consistently offer a substantial improvement. In fact, it is only on `resnet50`, in a highly multi-modal setting ($d = 10$), that `k-least-squares` reaches better accuracy. For the Transformer-based models, `k-means` performs better or comparably.

While the Transformer-based models generally achieve superior performance overall (note the different y-axis scales for CNN and Transformer models in the figure), a consistent trend across all models is observed: with increasing multi-modality (increasing d), using more than one proxy, whether through `k-means` or `k-least-squares`, begins to outperform the single-proxy methods (`mean` and `least-squares`) Still, it is striking how effective single weights with `least-squares` can be. However, it is also noteworthy how easily `k-means` can compensate for its limitations, particularly considering that `least-squares` is not an actual imprinting scheme, while `k-means` is.

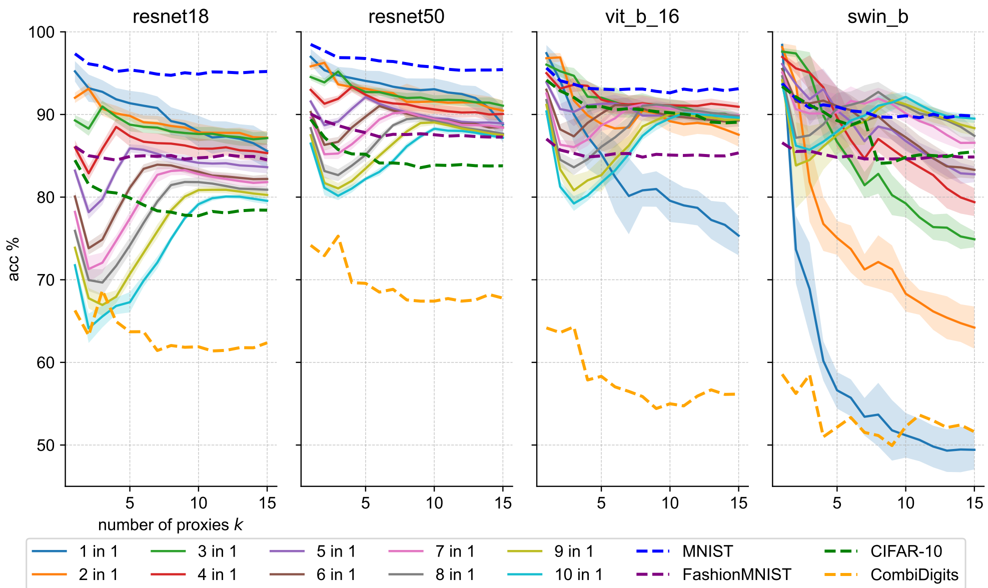


Figure A.7: Averaged accuracy of ten random relabelled *ImageNet* tasks ("d in 1") for every $d = 1, \ldots, 10$ over number of proxies k used for `k-least-squares` in `GEN`. 95% confidence intervals are shown in shaded colors. All `NORM` are set to `none`. Accuracies of the tasks containing all of *MNIST*, *FashionMNIST*, *CIFAR-10*, resp. *CombiDigits* at once are shown in dotted lines. As in figs. 12 and A.6, the CNN-based `FMs` exhibit a clear accuracy peak at $k = d$, with a consistent drop for intermediate values $1 < k < d$. For the Transformer-based `FMs`, however, accuracy is typically highest at $k = 1$ and only improves slightly around $k = d$ but does not show a pronounced peak. For $k > d$ and all `FMs`, performance declines again, indicating that excessive proxy splitting harms generalization. This degradation does not occur in `k-means GEN`, which remains robust even with large k.

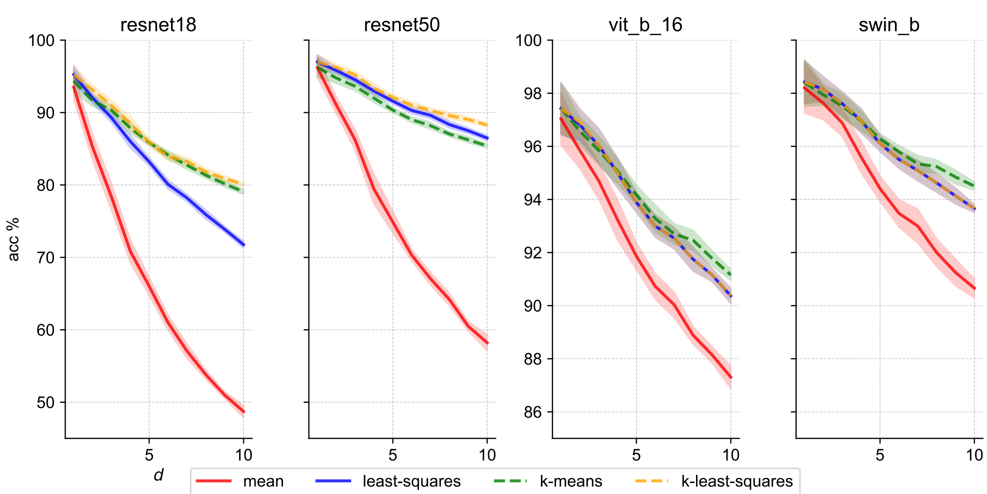


Figure A.8: Averaged accuracy of ten random *ImageNet* label remappings ("d in 1") for every $d = 1, \ldots, 10$ for `mean` and `least-squares`, and optimal `k-means` and `k-least-squares` ($k \in \{0, \ldots, 15\}$) weights. 95% confidence intervals are shown in shaded colors. Note the differing y-axis scales for CNN-based (`resnet18`, `resnet50`) and Transformer-based (`vit_b_16`, `swin_b`) models. The figure illustrates that with increasing multi-modality d, multi-proxy methods (`k-means`, `k-least-squares`) generally outperform single-proxy methods (`mean`, `least-squares`), and that the `k-means` imprinting scheme is competitive with `least-squares` approaches.