Learning an Informed CNN Initialization for Image Classification

Jannek Ulm Alexander Bayer Leander Diaz-Bone Dennis Jüni

Abstract

We introduce a novel weight initialization technique for Convolutional Neural Networks, focusing on image classification using the CIFAR-100 dataset. Our weight initialization technique initializes the convolutional filters by drawing samples from a clustering computed on previously learned filters. Our findings show that this method significantly outperforms known initialization techniques, improving validation accuracy and leading to better generalization on unseen image datasets. This offers a new approach to CNN weight initialization, with potential applications in more complex models and datasets, which may enable more efficient networks and faster training.

1. Introduction

In computer vision, Convolutional Neural Networks (CNNs) have been pivotal, maintaining their relevance even with the emergence of vision transformers (Smith et al., 2023). A crucial aspect of optimizing CNN performance is the strategy used for weight initialization. Traditional approaches often use a uniform distribution, but as underscored by Narkhede et al. (2022), an informed approach to weight initialization can significantly enhance model performance, improve convergence speed, and may enable more efficient models with fewer filters.

Our project dives into the initialization of CNN parameters, particularly for image classification, using the CIFAR-100 dataset. We aim to understand the weight distribution in CNNs, such as ResNet-18, guiding the initialization of new models. The core objective is to develop an initialization method that not only improves performance but also offers robust generalization capabilities across varied, unseen natural image data distributions.

Prior research has revealed common structures in convolutional filters across various models and vision tasks, as identified by Gavrikov & Keuper (2022). The initial layers of CNNs often resemble Gabor filters, a finding by Rivas & Rai (2023) highlighting the performance benefits of using such filters for initialization. Additionally, Trockman & Kolter (2023) demonstrated the advantages of initializing weights by mimicking those of pre-learned transformers. Another significant study by Trockman et al. (2022) examined the covariance matrix of convolutional filters, revealing its structured nature and relationship with filter positioning. By sampling weights from a Gaussian distribution based on the empirically derived covariance matrix, they achieved improved model performance.

We aim to introduce a new CNN initialization strategy that performs better in handling diverse image datasets that have not been seen by the model before and hence provides a simple method for CNN initialization for various of image classification tasks.

2. Methods

The primary strategy of our novel initialization technique is to learn the similarities of CNN filters for image classification. To this end, we train several models to collect filters and use clustering techniques to find similarities. The details are described in the following sections.

2.1. Model

The model of choice for our experiments was ResNet-18 (He et al., 2015). It is a well-known convolutional neural network with 18 layers, skip connections and a final fully connected layer. Its selection was based on the compatibility with our computational resources [B] and good efficiency in image processing tasks. For all experiments, except the Full Filter Clustering (see 2.4), we used the ResNet-18 implementation provided by PyTorch (see [D] for further explanation).

2.2. Datasets

The dataset of choice was CIFAR-100 (Krizhevsky et al.), consisting of 60,000 32×32 RGB images across 100 classes. It is divided into 20 superclasses, and we split these into "clustering" and "validation" groups, each encompassing ten superclasses. Note that each group contains an independent training and validation split. Models were trained on pairs of superclasses to focus on specific semantic themes. We used all 45 pairs of combinations of the 10 "clustering" superclasses to generate the trained models for the clustering algorithms, resulting in 45 specialized classification models.

To further test the generalization of our initialization technique, we used the TinyImageNet dataset, a subset of the ImageNet classification dataset (Deng et al., 2009). It consists of 200 object classes, with 500 training and 50 validation images per class. All images have been downsampled to 64×64×3 pixels. We did not use the additionally provided bounding boxes for either training or validation.

2.3. Training

Uniform training settings were applied across models. Hyperparameters were optimized for maximum mean validation accuracy on a subset of "clustering" superclasses over multiple runs. More details regarding the specific parameters can be found in the appendix [C].

2.4. Clustering & Initialization

After having pre-trained several models on subsets of the CIFAR-100 dataset, we cluster filters from the pre-trained models per layer. We distinguish two different ways of clustering the filters. **Full Filter Clustering**, where we treat each filter as an individual, 3d (input channel × kernel size × kernel size) entity, clustering the entire filter. This allows us to align later layers of the pre-trained models (2.4.2). We then initialize the filters with the mean of the aligned filter weights. **Single Filter Clustering**, where we dissect each 3d filter into its n underlying 2d filters of the n input channels. The clustering is then performed on all 2d filters regardless of their input channel. This approach allows us to analyze and cluster more granular components of the filters. The models then were initialized per layer, that is, for each 3d filter block, its 2d sub-filters were independently sampled from the computed cluster centers according to the size of the cluster.

2.4.1. Clustering approach

We explored various clustering techniques to find the most effective and best-performing technique. Based on our testing results we chose to pursue KMeans as our main clustering algorithm due to its superiority in the results from the experiments.

Within the KMeans algorithm, the choice of the distance metric is crucial. Again, we tried multiple metrics, namely, the Euclidean distance, the cosine similarity, and the Euclidean distance of the magnitude of filters in the Fourier space (referred to as "Fourier distance"). Due to the lack of good results, we did not analyze cosine similarity further.

We performed the experiments with varying numbers of clusters, numbers of pre-trained models used for the clustering, and numbers of layers to initialize based on the clustering approach. If not all filters were initialized according to our clustering technique, the rest were randomly initialized.

2.4.2. Alignment algorithm

The following explains a novel approach to align multiple pre-trained models of the same architecture when using full filter clustering. If we sequentially cluster every layer and ensure that no two filters end up in the same cluster per pre-trained model, we can effectively align all pre-trained models by permuting the order of the filters in a layer. To maintain every model's integrity, we can permute the channels of every filter in the next layer identically. Figure 1 shows a simplified visualization of the alignment algorithm.

Intuitively, this should improve the following layer's clustering capability since the filters are correctly aligned. Note that it does not work with residual connections since they cannot be permuted similarly. One could still get around this by training in multiple steps with an additional layer frozen in every iteration. After considering the preliminary results, we decided not to proceed with testing the alignment algorithm with the actual ResNet-18 architecture. To evaluate the alignment algorithm, we modified ResNet-18 by removing the residual connections in the first block and using a simple greedy approach to cluster the filters. In the following, we refer to this modified ResNet-18 as Custom ResNet-18.

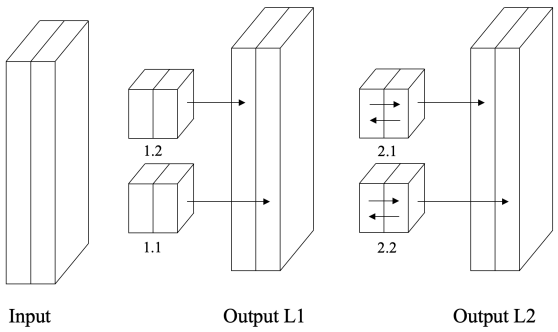

Figure 1. Visual representation of the alignment algorithm for a model with two layers, two convolutional filters each, and two input channels. In this case, the order of the filters in layer one was swapped, and the filters in layer two were adjusted accordingly.

2.5. Evaluation

For a fair comparison, all models underwent evaluation on the same set of 10 randomly selected superclass pairs from the "validation" segment of the CIFAR-100 dataset. This ensures comparability of results. Each model configuration was replicated and initialized five times with rerun clustering algorithms to evaluate the stability and repeatability of the clustering algorithm. The validation accuracies reported in our study are the mean values computed across these five

runs for all ten tuples, thereby providing a comprehensive and consistent measure of model performance.

To compare our methods, we compared the results with randomly initialized models, a random selection of the pre-trained models on the "clustering" dataset, and Gabor initialization (Rivas & Rai, 2023).

3. Results

3.1. Single Filter Clustering Comparison

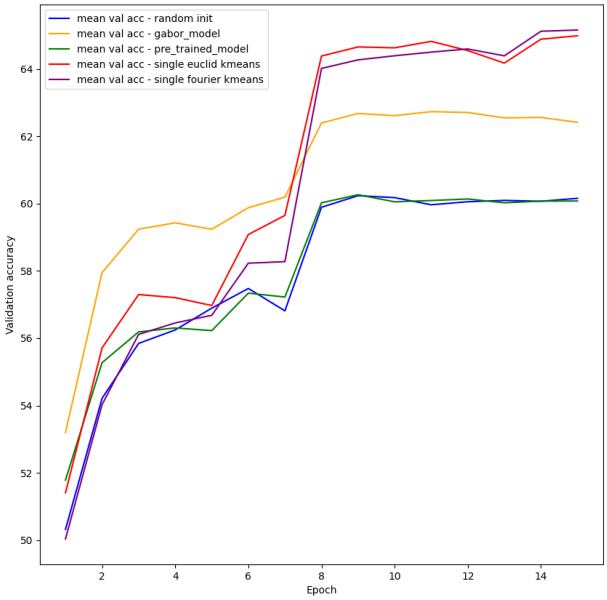

Figure 2. Mean validation accuracy plot over the training for the single-filter clustering initialization methods, including the uniform initialization, Gabor initialization, and a pre-trained model as baselines. The results show the optimal hyperparameter selection.

Figure 2 shows that our clustering initializations with Euclidean and Fourier distances perform similarly well, with the clustering using the Fourier distance yielding a marginally better accuracy. Both approaches performed significantly better than all of the baselines, improving the accuracy compared to the uniformly random initialized model by more than 4% and compared to the Gabor initialization by more than 2% on average. The Gabor filter initialization proved especially beneficial during the first few epochs of training and finally achieved higher accuracy compared to the other baselines; this corresponds to the findings of previous work. While we initialized only the first layer with Gabor filters compared to initializing the whole model for the other methods, we observed that initializing more than one layer with Gabor filters reduced the performance. This result can be found in the appendix in Figure 7.

The validation accuracies for clustering the whole filters combined with aligning the different networks did not yield preferable results compared to the baselines. The results can be found in Figure 6 in the appendix. A visualization of the learned filters can be found in the appendix in Figure 10.

3.2. Effects of Clustering Hyperparameter Selection

In the next experiments, we evaluated the effect of different hyperparameter choices on the classification accuracies. Firstly, we investigated how the number of clusters that we used to cluster the filters influenced the result. Figure 3 shows that, interestingly, as long as we choose more than one cluster, we achieve similar accuracies. This is surprising, as we achieve high accuracy for models initialized with as few as two different filters. Choosing only one cluster, on the other hand, slows down convergence and leads to a worse accuracy.

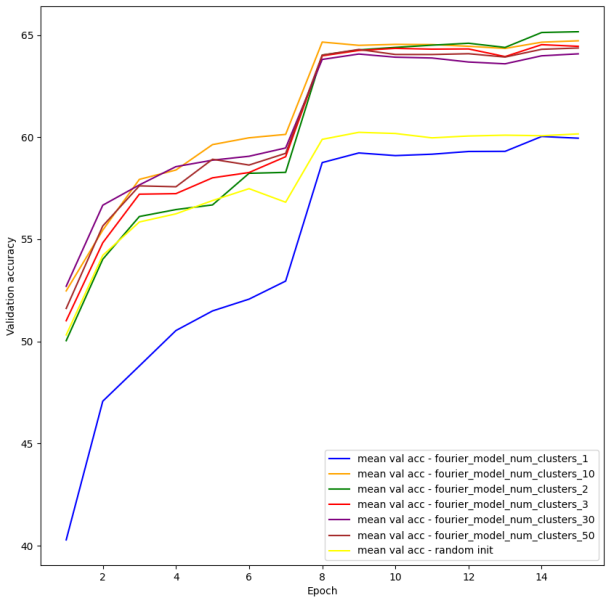

Figure 3. Mean validation accuracy plot over the training for the single-filter clustering initialization methods with varying numbers of clusters used for the initialization. All layers were initialized.

Secondly, we studied the effect of initializing only parts of the model. Figure 4 shows the effect of initializing only the first k-layers using our clustering approaches. The plot shows that the accuracies of the models increase if we initialize more layers of the models. This shows that initialization beyond the first layer, considered in previous work, can increase performance significantly. Notably, if we only initialize the first layer of the model, the Gabor filter initialization outperforms our clustering methods. Figure 8 visualizes this behavior.

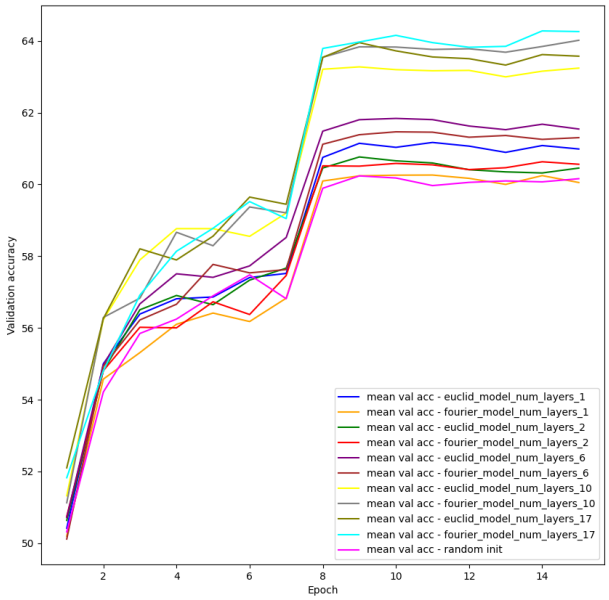

Figure 4. Mean validation accuracy plot over the training for the Fourier KMeans single-filter clustering initialization methods with varying numbers of initialized layers. Other hyperparameters where chosen optimally.

The last hyperparameter of the clustering we investigated was the number of models used to collect filters. Changing the number of models did not significantly affect the validation accuracy. Consequently, only a small number of models is required to learn the filters we use for the clustering, making our method computationally efficient. The results are in Figure 5 in the appendix.

3.3. Evaluation on different Datasets

Lastly, we evaluate the capabilities of our initialization to generalize to a different dataset. In contrast to the last experiments, where we evaluated the initialization on unseen classes, we now test the generalization on the Tiny ImageNet dataset, which was not used to learn the filters. Table 1 and figure 9 show that our initialization approaches still improve the validation performance compared to the random initialization and compared to a pre-trained model on ten classes of CIFAR-100. This indicates that our initialization strategy can generalize to new and unseen data, which was the objective of this work.

4. Discussion

Our paper introduces a novel weight initialization technique for CNNs that enhances top-1 accuracy by approximately 5% on previously unseen image classes. Notably, our initialization technique also improves the accuracy on a different dataset. Compared to the most commonly used random initialization or the known improvement by using Gabor filters in the receptive layer, our approach performs significantly better. While we get consistently better results, it does require multiple pre-trained models to get to our initialization. Compared to random initialization or initialization with Gabor filters, this is an overhead in initialization but saves computational costs for further downstream model training. Another interesting insight is that a low number of clusters improves the accuracy of a model initialized with our technique. Our research contributes to a deeper understanding of weight initialization in CNNs.

Table 1. Mean validation accuracy during training on the Tiny ImageNet dataset. The clustered model was clustered with Fourier distance, using 6 pre-trained models on CIFAR-100, 3 clusters, and by initializing all 17 layers.

MODEL	EPOCH 5	EPOCH 15
RANDOMLY INITIALIZED	28.246	30.924
PRE-TRAINED ON CIFAR-100	28.338	30.268
CLUSTERED INITIALIZATION	**31.1**	**35.374**

Unfortunately, our alignment algorithm did not lead to any improvement when compared to a randomly initialized model. One possible explanation is the curse of dimensionality for clustering the 3d filters.

Further research should analyze the performance of our initialization technique on larger models and different datasets or even different tasks. Another interesting study could focus on gaining a deeper understanding of the sampled weights and considering different sampling and initialization strategies.

5. Summary

In this work, we introduced a novel weight initialization strategy for CNNs, used for image classification. We were able to show the superior generalization performance of our method compared to several baselines, such as random initialization and initialization using Gabor filters. Furthermore, we investigated the effect of different hyperparameter settings, leading to a better understanding of our method as well as initialization of CNNs.

References

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009. doi: 10.1109/CVPR.2009.5206848.

Gavrikov, P. and Keuper, J. CNN filter DB: An empirical investigation of trained convolutional filters. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, jun 2022. doi: 10.1109/cvpr52688.2022.01848. URL https://doi.org/10.1109%2Fcvpr52688.2022.01848.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. URL http://arxiv.org/abs/1512.03385.

Krizhevsky, A., Nair, V., and Hinton, G. Cifar-100 (canadian institute for advanced research). URL http://www.cs.toronto.edu/~kriz/cifar.html.

Narkhede, M. V., Bartakke, P. P., and Sutaone, M. S. A review on weight initialization strategies for neural networks. *Artif. Intell. Rev.*, 55(1):291–322, jan 2022. ISSN 0269-2821. doi: 10.1007/s10462-021-10033-z. URL https://doi.org/10.1007/s10462-021-10033-z.

Rivas, P. and Rai, M. Gabor filters as initializers for convolutional neural networks: A study on inductive bias and performance on image classification. In *LatinX in AI Workshop at ICML 2023 (Regular Deadline)*, 2023. URL https://openreview.net/forum?id=4c4ABljkC1.

Smith, S. L., Brock, A., Berrada, L., and De, S. Convnets match vision transformers at scale, 2023.

Trockman, A. and Kolter, J. Z. Mimetic initialization of self-attention layers, 2023.

Trockman, A., Willmott, D., and Kolter, J. Z. Understanding the covariance structure of convolutional filters, 2022.

A. Code

The code for this project containing all the experiments, plots, and tracked parameters for each training run can be found on GitHub. Due to size limitations we could not upload the 45 trained models, but they can easily be recomputed by running the corresponding Python scripts. https://github.com/janulm/deep-learning-advanced-initialization

B. Computational Resources

All experiments were performed on a system with an Nvidia RTX 3090, AMD Ryzen9 5900X, and 64 GB of RAM. Training one model on the sub dataset, which consists of two superclasses, for 15 epochs took 10s.

C. Hyperparameters

The following settings were used to train all models: The optimizer of choices was SGD with learning rate set to 0.01, momentum set to 0.9, and weight decay set to $1e^{-4}$ with a batch size of 128, for 15 epochs. Input images were normalized. If the validation accuracy did not increase for five consecutive epochs, the learning rate was halved.

D. Model choice

The tests for the CIFAR-10 datasets in the original ResNet paper (He et al., 2015) were performed using a slightly different version than the implementation provided by PyTorch. Since the implementation by PyTorch focuses on general image classification tasks where generally larger images are used than in CIFAR-100, the initial downsampling is a lot more aggressive than the one in the original paper. However, since our goal was not to improve the maximum accuracy of a model and we also wanted to test our models on larger images, we still chose the PyTorch implementation. This also explains why our maximally achieved accuracy on a simple dataset such as CIFAR-100 never really exceeds 65%.

E. Additional Figures

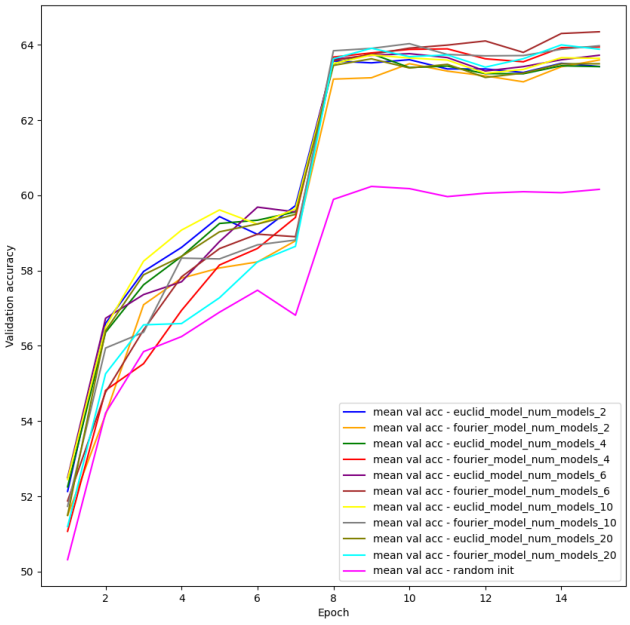

Figure 5. Mean validation accuracy plot over the training for a different number of models used to produce the clustering, comparing both Fourier and Euclidean clustering approaches.

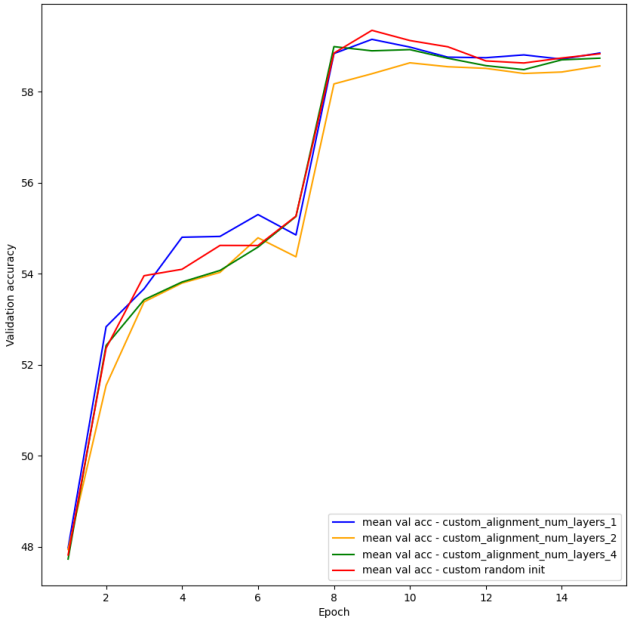

Figure 6. Mean validation accuracy plot over the training for the clustering of full filters with alignment compared to the uniformly random initialization on the modified ResNet-18.

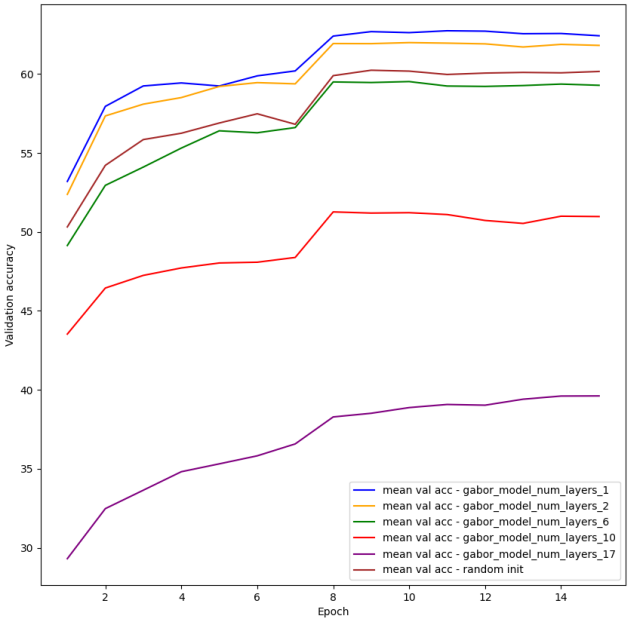

Figure 7. Mean validation accuracy plot over the training for different number of layers being initialized with Gabor Filters.

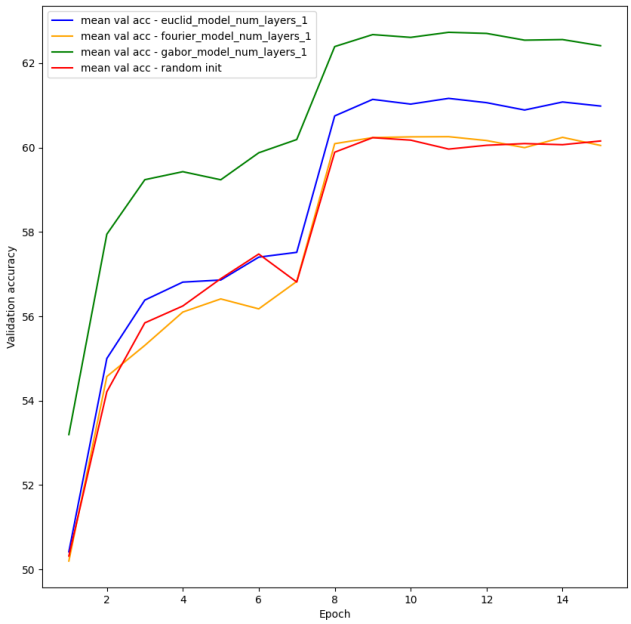

Figure 8. Mean validation accuracy plot over the training only initializing the first layer of the network using different methods.

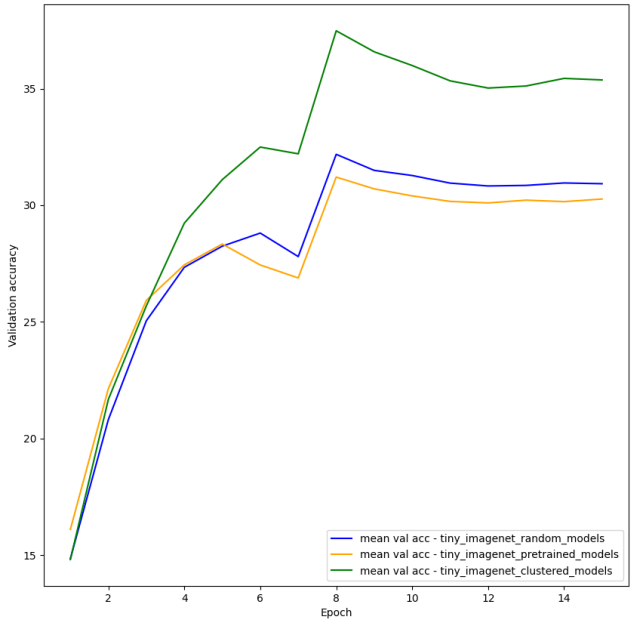

Figure 9. Mean validation accuracy plot over the training for the single-filter clustering initialization methods on the Tiny ImageNet dataset. The clustered model was clustered with Fourier distance, using 6 pre-trained models, 3 clusters, and by initializing all 17 layers.

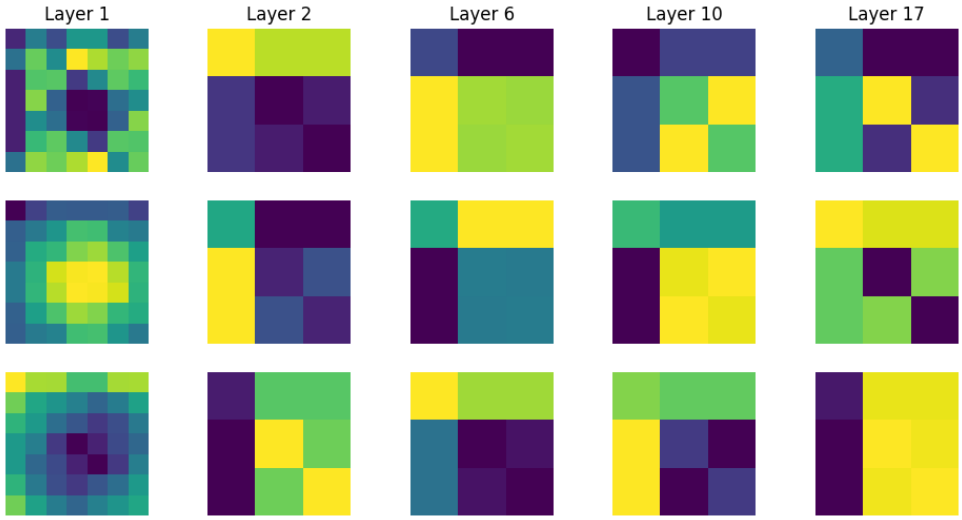

Figure 10. Visualization of the filters of a clustered model with 3 clusters, using 6 pre-trained models, and by initializing all 17 layers.