arXiv:2509.11711v1 [cs.CV] 15 Sep 2025

The Quest for Universal Master Key Filters in DS-CNNs

Zahra Babaiee*
Technische Universität Wien
Vienna, Austria
zahra.babaiee@tuwien.ac.at

Peyman M. Kiassari†
Technische Universität Wien
Vienna, Austria
peyman.kiasari@tuwien.ac.at

Daniela Rus
Massachusetts Institute of Technology
Cambridge, MA
rus@mit.edu

Radu Grosu
Technische Universität Wien
Vienna, Austria
radu.grosu@tuwien.ac.at

Abstract

A recent study has proposed the "Master Key Filters Hypothesis" for convolutional neural network filters. This paper extends this hypothesis by radically constraining its scope to a single set of just 8 universal filters that depthwise separable convolutional networks inherently converge to. While conventional DS-CNNs employ thousands of distinct trained filters, our analysis reveals these filters are predominantly linear shifts (ax+b) of our discovered universal set. Through systematic unsupervised search, we extracted these fundamental patterns across different architectures and datasets. Remarkably, networks initialized with these 8 unique frozen filters achieve over 80% ImageNet accuracy, and even outperform models with thousands of trainable parameters when applied to smaller datasets. The identified master key filters closely match Difference of Gaussians (DoGs), Gaussians, and their derivatives, structures that are not only fundamental to classical image processing but also strikingly similar to receptive fields in mammalian visual systems. Our findings provide compelling evidence that depthwise convolutional layers naturally gravitate toward this fundamental set of spatial operators regardless of task or architecture. This work offers new insights for understanding generalization and transfer learning through the universal language of these master key filters.

1 Introduction

Convolutional Neural Networks (CNNs) have significantly advanced computer vision through their hierarchical representations using trainable filters. As architectures evolved toward greater performance, models such as VGG [24], ResNet [7], and DenseNet [11] incorporated thousands of filters across their layers. This trend continued with the development of Depthwise Separable Convolutional Neural Networks (DS-CNNs) [10, 9], which separate spatial and channel-wise computations for improved efficiency. Contemporary architectures like the ConvNeXt family [20, 29] utilize DS-CNNs with up to 50,000 trainable spatial filters.

Recent research identified a notable pattern in trained depthwise convolutional kernels across various DS-CNN architectures [1]. Through analysis of trained filters using unsupervised clustering, they demonstrated that these patterns converge into distinct clusters resembling Difference of Gaussian

*Equal contribution.
†Equal contribution.

Preprint.

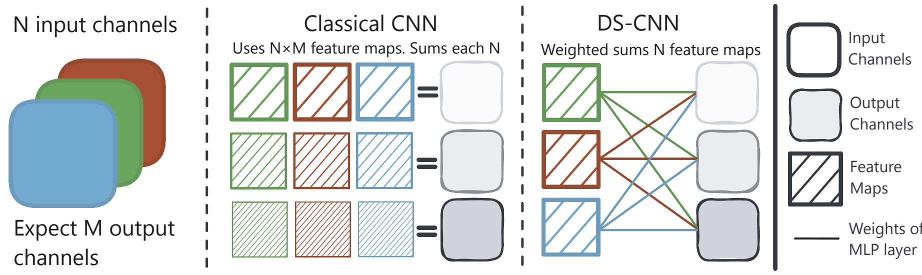

Figure 1: **Comparison of Classical CNN and DS-CNN architectures:** Left: Input with N channels. Center: In Classical CNNs, each output channel is produced by convolving a unique filter with each input channel, followed by summing the resulting feature maps. This results in N×M distinct filters and corresponding feature maps. Right: DS-CNN uses only N filters (one per input channel) to create N feature maps, then applies an N×M MLP layer to linearly combine these feature maps into M output channels. DS-CNNs represent a parameter-efficient *subset* of classical CNNs, reducing the number of required convolutional operations.

(DoG) functions and their derivatives. Their study classified over 95% and 90% of filters from ConvNextV2 and ConvNeXt models, respectively, into Gaussian-related clusters, indicating consistent patterns in filter learning.

Subsequently, another work proposed the "Master Key Filters Hypothesis," [2] proposing that there exist master key filter sets that are general for visual data, and that the depthwise filters in DS-CNNs tend to converge to these master key filters, regardless of the specific dataset, task, or architecture. This hypothesis challenges the conventional understanding that convolutional filters become increasingly specialized in deeper layers and suggests instead that a set of fundamental filters may underlie the performance of these networks. In this paper, we extend the "Master Key Filters Hypothesis" by radically constraining its scope through identification of a minimal fundamental master key filter set. While the original hypothesis posited the existence of general-purpose filter sets for visual data—potentially comprising numerous filters across multiple sets—our systematic unsupervised analysis across architectures and datasets reveals a remarkably compact representation. We demonstrate that DS-CNNs predominantly converge toward a basis of just 8 distinct filters, where a substantial proportion of learned filters approximate linear shifts of these fundamental kernels. Notably, networks restricted to this compact basis maintain performance integrity, suggesting these filters capture essential visual processing primitives rather than task-specific features. This finding significantly refines the original hypothesis by establishing both the cardinality and specific form of a universal filter basis for visual computing.

Our work also substantially refines the observations made in [1], which identified Gaussian-related patterns in DS-CNN filters without constraining their potential variability. While that study demonstrated the prevalence of Guassian-like filters, it allowed for an effectively infinite continuum of these structures with arbitrary standard deviations and noise characteristics. In contrast, we're narrowing it down as linear shift of a mere 8 fundamental filters. This significantly narrows the theoretical space.

These identified filters correspond to mathematical forms matching Difference of Gaussians (DoGs), Gaussians, and their derivatives, which are established components in scale-space theory [16] and share structural similarities with receptive fields observed in mammalian visual systems [30, 31]. Networks initialized with these 8 filters achieve over 80% ImageNet accuracy and demonstrate superior performance compared to models with thousands of trainable parameters when applied to smaller datasets.

Our findings provide empirical support for the "Master Key Filters Hypothesis" and suggest potential applications in efficient network design and transfer learning, while contributing to the understanding of generalizable patterns in visual processing systems.

2 Related Work

Depthwise Convolutional Filters. Depthwise Convolutions (DCs) have revolutionized the design of Convolutional Neural Networks (CNNs) by using only one feature map per input channel, leading to the development of lightweight and performant architectures such as MobileNet [10], EfficientNet [26], and ConvNeXt [20]. As illustrated in Figure 1, classical CNNs utilize $c_{in} \times c_{out}$ completely separate filters, creating independent feature maps for each input-output channel combination. Each output channel is computed as:

$$Y_i = \sum_{j=1}^{c_{in}} K_{i,j} * X_j, \text{for } (i,j) \in [c_{out}] \times [c_{in}]$$

where $X \in \mathbb{R}^{H \times W \times c_{in}}$ is the input tensor with c_{in} channels, Y_i is the i-th output channel, $K_{i,j}$ is the convolutional kernel for the i-th output channel and j-th input channel, and $[n]$ denotes the set $\{1, 2, ..., n\}$.

In contrast, DS-CNNs force the model to use only c_{in} feature maps (one per input channel) followed by linearly combining these feature maps using pointwise convolutions or MLP layers. It is important to emphasize that DS-CNNs combine the feature maps resulting from the depthwise convolutions rather than linearly combining the kernels directly:

$$Y_i = \sum_{j=1}^{c_{in}} W_{i,j} \cdot K_j * X_j, \text{for } (i,j) \in [c_{out}] \times [c_{in}]$$

Although DS-CNNs are restricted-CNNs, but this restriction significantly reduces the parameter count while maintaining competitive performance.

Recent studies have revealed striking properties of depthwise convolutional filters in these networks. Trockman et al. [28] observed that learned filters in their DS-CNN model ConvMixer exhibit highly structured covariance matrices. Furthermore, Babaiee et al. [1] discovered that trained depthwise convolutional kernels across all layers of DS-CNNs converge into a few main clusters, each resembling the difference of Gaussian (DoG) functions and their first and second-order derivatives. The authors were able to classify the majority of the filters from state-of-the-art DS-CNN models. Building on this work, recent work. [2] introduced the "Master Key Filters Hypothesis," which proposes that depthwise filters in DS-CNNs exhibit generality across domains, architectures, and layer depths, challenging the conventional view that deeper layers become increasingly specialized. Our work builds upon these findings by investigating the potential of using a limited number of unique filters in DS-CNNs, exploiting the observed clusterability in depthwise convolutional kernels, and moving towards finding the master key filters.

Filter Diversity. Filter pruning and compression techniques have been widely explored to reduce the computational complexity and memory footprint of CNNs [8]. Structured pruning in CNNs is typically achieved by removing redundant filters [15]. These methods highlight the importance of "feature-map" diversity in CNNs, as removing redundant or less informative filters can lead to more efficient models without significant performance degradation. In contrast, our work is not attempting to reduce the number of parameters or computational cost, but rather investigating the role of "filter" diversity in DS-CNNs and challenging the assumption that a large number of unique filters is necessary for optimal performance. By discovering that a small set of carefully chosen filters can effectively replace a large number of learned filters in DS-CNNs, we are shedding light on the inherent limited diversity present in the learned depthwise filters.

Scale-Space Theory. Scale-space theory, which examines signals across different scales, was initially developed in the mid-1980s [12] and has since become a fundamental concept in signal processing, particularly within the field of computer vision. Lindeberg [16] introduced a computational framework for visual receptive fields that exploits symmetry properties across space and time. This framework is compelling for two reasons [19]: First, it offers a normative perspective on visual processing that closely aligns with the hierarchical stages observed in the visual systems of higher mammals[17]. Second, it provides a provable approach to capture the natural transformations of images over space and time [18]. Gaussian derivatives are the sole kernels satisfying isotropy (rotational invariance) and non-creativity (with respect to the causality principle) in scale-space theory [16]. Remarkably, our work reveals that the master key filters required for efficient performance in DS-CNNs consists

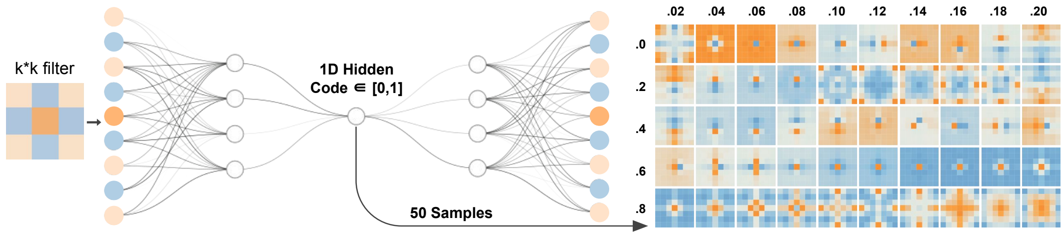

Figure 2: **Visualization of selecting candidate filters using autoencoder-based dimensionality reduction**. Left: An autoencoder compresses filters into a 1D hidden code. Right: Heatmap of 50 uniformly sampled candidate filters from 1D hidden code, generated by the decoder part of the autoencoder. These samples serve as the initial pool for our search for the master key filters.

of only 8 distinct filters: Gaussians, Difference of Gaussians (DoG, which can be approximated by the Laplacian of Gaussians), and first derivatives of Gaussians. This finding establishes a strong connection between the principles of scale-space theory and the design of CNN architectures.

3 Do We Need Thousands of Distinct Filters?

In this section, we investigate whether employing thousands of unique filters is essential for maintaining the performance of DS-CNNs. In particular, we explore what is the impact on the performance of the network, when we replace the trained filters with a minimal set of distinct filter variations.

3.1 The Quest for Master Key Filters

In order to explore the possibility of reducing the number of distinct filters in DS-CNNs, we sought to distill the filters of trained models into a compact set. We collected filters from publicly available trained models of various sizes and employed an autoencoder to learn a compressed representation of these filters. The autoencoder was trained to encode each filter into a single dimension, following a similar procedure as the one described in [1]. This approach allowed us to capture the essential characteristics of the filters while significantly reducing their dimensionality.

To create a comprehensive filter set, we collected all the depthwise filters from every layer of the networks in our model bank. Each filter had a consistent size of 7×7. To ensure uniformity, we normalized the filters by first centering them and then by scaling their length to 1. This normalized filter set finally served as the training data for our autoencoder.

The autoencoder architecture comprises two primary components: an encoder and a decoder. The encoder consists of four intermediate layers, each followed by a leaky rectified linear unit (Leaky ReLU) activation function. These layers progressively compress and abstract the input filter representations. The final layer of the encoder, known as the code layer, employs a sigmoid activation function to map the compressed filter representations to values within the range [0, 1]. This mapping ensures that the encoded filters are bounded and compatible with the subsequent decoding process.

The decoder on the other hand, is responsible for reconstructing the original normalized filters from the encoded representations. It mirrors the structure of the encoder, with four intermediate layers that gradually upsample and expand the encoded features. The final layer of the decoder utilizes a hyperbolic tangent (tanh) activation function, which allows for accurate reconstruction of the normalized filters within [-1, 1]. This choice of activation function ensures that reconstructed filters maintain their centering and scale, aligning with the characteristics of the original normalized filters.

By training the autoencoder on the diverse set of normalized filters, we aim to learn a compact and meaningful representation of the filter space. The encoder captures the essential features and patterns present in the filters, while the decoder enables the reconstruction of filters from their encoded representations. This architecture facilitates the exploration of filter variations and the potential for reducing the number of distinct filters required in depthwise separable convolutional neural networks.

After training the autoencoder, we performed uniform sampling from the code layer. We took various numbers of samples, 50, 25, and 10, from the [0,1] interval to generate distinct filter sets. Using the decoder, we transformed these codes back into filter reconstructions.

A depthwise filter for the c-th channel can be denoted as F_c, where c is the index of the channel. When flattened, F_c can be represented as a vector $\mathbf{f}_c \in \mathbb{R}^{k^2}$, where $k \times k$ is the spatial dimension of the filter. The matrix F composed of these flattened vectors is $F = [\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_C]^T \in \mathbb{R}^{C \times k^2}$.

For each depthwise filter F_c learned by the ConvNeXtV2-tiny model, we then conducted a linear approximation with respect to the decoded filters, by identifying the scalar coefficients a and b, which minimized the Euclidean distance between the corresponding flattened filter vector f_c, and the linear combination $af'_c + b$, where f'_c represents a flattened decoded-filter sample. The original filter was then substituted with the optimal linear combination $af'_c + b$ that exhibited the smallest distance to the original, thus preserving the filter's functional characteristics while reducing model complexity.

To solve for scalars a and b that minimize the distance between vectors f_c and $af'_c + b$, we use linear regression. Here, the goal is to determine the coefficients a and b for two vectors x and y such that by having $\tilde{y} = ax + b$ the length of the vector $y - \tilde{y}$ is minimized. This problem has a well-known solution.

$$a = \frac{n\sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} y_i}{n\sum_{i=1}^{n} x_i^2 - (\sum_{i=1}^{n} x_i)^2} \quad b = \frac{\sum_{i=1}^{n} y_i \sum_{i=1}^{n} x_i^2 - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} x_i y_i}{n\sum_{i=1}^{n} x_i^2 - (\sum_{i=1}^{n} x_i)^2} \tag{1}$$

Calculating Equations (1) can be computationally intensive, especially when dealing with hundreds of thousands of filters. To reduce computational complexity, we can use a normalization trick. Since any linear shift of x does not alter the optimal $\tilde{y}$, we normalize x using the transformation $\hat{x} = \frac{x-\bar{x}}{||x-\bar{x}||}$. With this normalization, $\sum_{i=1}^{n} \hat{x}_i = 0$ and $\sum_{i=1}^{n} \hat{x}_i^2 = 1$, allowing us to simplify Equation(1).

$$a = \frac{n\sum_{i=1}^{n} x_i y_i}{n} = \langle x, y \rangle \quad b = \frac{\sum_{i=1}^{n} y_i}{n} = \bar{y} \tag{2}$$

Consequentially, Given the vectors $\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_n$ as the rows of matrix $\hat{X}$ and the vectors $y_1, y_2, \ldots, y_m$ as the columns of matrix Y, we introduce the vector y_{mean}, which contains the means $\bar{y}_1, \bar{y}_2, \ldots, \bar{y}_m$. Using these, we can calculate the coefficients a_{ij} and b_{ij} for each pair of x_i and y_j through matrix multiplication.

$$A = \hat{X}Y \quad B = y_{\text{mean}}\mathbf{1}^\top \tag{3}$$

For each layer, with the set of depthwise filter vectors matrix F and the sample filter vectors matrix F', we calculate the coefficients as above to find the closest linear approximation.

We chose linear shifts for approximating the original filters because they preserve the heatmap and essential characteristics of the filters. By applying a linear shift to a filter sample, we maintain the spatial structure and relative importance of different regions within the filter.

In order to evaluate the impact of replacing the original filters with their linear approximations derived from the sampled filter set, we assessed the performance of the modified models on the test set. In Table 1 we present the accuracy of models with varying sizes from the ConvNeXtv2 and Hornet [23] families, along with their accuracy after their filter replacement. Quite remarkably, when replacing all the filters of the models with approximations based on only 50 sampled filters, the model performance remains robust, even without any fine-tuning. This resilience is particularly evident for larger model sizes. In the case of ConvNeXtv2 Huge, replacing nearly 50K filters with just 50 sampled filters results in less than a two percent accuracy drop, without any fine-tuning.

As expected, reducing the number of sampled filters leads to a larger accuracy gap, and the ConvNeXtv2 models struggle to perform well when using an small set of only 10 filters. However, it is important to note that the filter samples used in this experiment were obtained through uniform sampling from the code layer of the autoencoder. This immediately raises the following question: *Is there a more strategically selected set of filter samples which can yield a better performance?*

To elucidate this question, we focused on the ConvNeXt-v2-Tiny model and conducted a greedy search on a set of 50 filter samples. We began with the 50 uniformly sampled filters and iteratively removed filters one by one. Figure 2 illustrates the 50 filter samples used in this search. At each

Table 1: **Performance comparison when thousands of trained filters are replaced with linear shifts (ax+b) from candidate filters.** *Without **any** fine-tuning* (For trained model see Table 2), models with just 8 selected filters from our greedy search on *ConvNeXtv2 Tiny* maintain remarkably high accuracy (e.g., only 3.5% drop for ConvNeXtv2 Huge despite reducing from 50k to 8 unique filter patterns) and even on different architecture, HorNet. Concidering models' high sensitivity to filter alterations, this is evidence that DS-CNN filters predominantly converge to these filters.

	ConvNeXt					Hornet	
ConvNeXtv2 Models	**Pico**	**Tiny**	**Base**	**Large**	**Huge**	**Tiny**	**Small**
Number of Filters	2 944	6 624	18 048	27 072	49 632	11 488	17 232
Original Acc	80.3%	83.0%	84.9%	85.8%	86.3%	82.3%	83.5%
Acc with 50 candidates	75.0%	75.4%	80.5%	83.2%	84.0%	79.4%	81.3%
Acc with 25 candidates	72.0%	66.9%	72.8%	79.6%	80.4%	78.3%	80.9%
Acc with 10 candidates	23.4%	1.0%	1.4%	3.0%	2.0%	66.3%	70.5%
Acc with 8 (greedy search)	73.1%	76.7%	79.3%	81.2%	82.8%	76.0%	78.1%
Acc with 8 random filters	0.11%	0.10%	0.10%	0.12%	0.09%	0.96%	1.0%

iteration, we evaluated the model accuracy after removing each individual filter and eliminated the one whose removal resulted in the smallest accuracy drop. This process was repeated for all filters.

The accuracy plot during the greedy search, as shown in Figure 3, reveals an interesting trend. The model's accuracy remains relatively stable until the last 10 filters are removed, with the curve exhibiting a distinct elbow around 8 samples. This observation suggests that a small subset of only 10 filters is playing a crucial role in maintaining the performance of the model. To further refine the search, we selected the 10 best-performing filters from the previous step and expanded our search space by sampling 4 additional filters around each of these 10 filters. This local exploration allows us to fine-tune the selection of filters and capture any potential variations that may enhance performance.

We then conducted a second round of the greedy search using this expanded set of filters. The search converged to a set of 8 filters located just after the curve elbow. These 8 filters, as shown in Figure 4, represent a highly informative subset that can effectively replace the original large set of filters, while minimally impacting the model's accuracy.

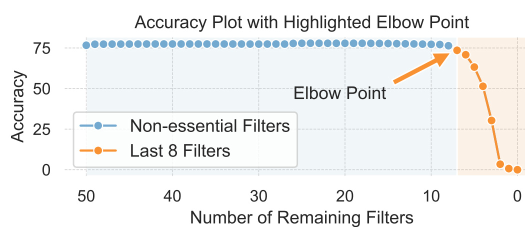

Figure 3: **Our systematic greedy search for the essential filters**. While the removal of most filters did not noticeably change accuracy, 8 of them were essential, consistently in all models we tested.

The last row of Table 1 showcases the accuracy of the models, when their filters are replaced by the 8 filter transformations, obtained from the greedy search. Remarkably, the results demonstrate that the ConvNeXtv2 models accuracy achieved with these 8 filters, surpasses even the performance of the 25-filters-sample set. This finding underscores the effectiveness of the greedy search approach in identifying a highly discriminative subset of filters. Moreover, it highlights the potential for replacing a large number of filters, up to 50K in the case of ConvNeXt v2 models, with just 8 strategically selected filters, while maintaining an acceptable performance.

To validate the generalizability of these 8 filter samples, we extended our experiments to the ConvNeXt-v2-Pico model, which represents a different model size. In this case also, we arrived at a similar set of 8 filter samples, indicating the robustness and transferability of our findings across different model architectures. The consistency of the 8 filter samples across different model sizes suggests that these filters capture fundamental and generalizable patterns in the data. It hints at the existence of a set of universal filters that can effectively represent the essential information required for accurate classification.

The remarkable performance maintained when replacing thousands of trained filters with just our 8 master key filters cannot be coincidental. Noting models' high sensitivity to filter alterations (evidenced by the catastrophic performance drop with 8 random filters), strongly indicates that DS-

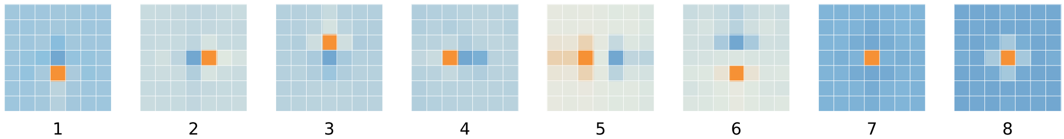

Figure 4: **Heatmap visualization of the eight universal filters discovered through systematic greedy search on the ConvNeXtv2 tiny model.** Our empirical analysis demonstrates that DS-CNN filters predominantly converge to linear shifts (ax+b) of one of these eight filters, regardless of architecture or dataset. Filters 1-4 display central difference operator characteristics, and filters 5-8 correspond to established mathematical image processing fundamentals (See Figure 5).

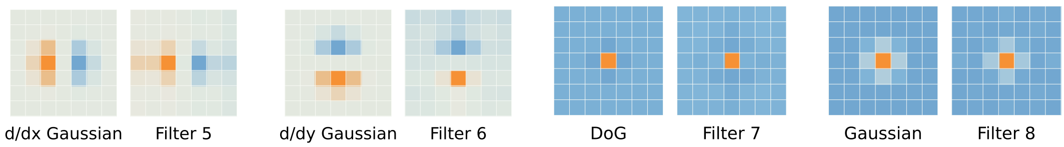

Figure 5: **Correspondence between the empirically discovered filters (5-8) and their theoretical mathematical counterparts.** Left column shows the idealized mathematical forms: first derivatives of Gaussians in x and y directions (filters 5-6), Difference of Gaussians (filter 7), and Gaussian function (filter 8). Right column shows our discovered filters that closely approximate these operators, demonstrating the network's natural convergence toward established visual processing primitives.

CNN filters predominantly converge to fundamental patterns during training. Even more compelling, these 8 filters (discovered exclusively from ConvNextV2 Tiny) transfer seamlessly to architecturally distinct models like Hornet. This fascinating cross-architecture generalization suggests that DS-CNNs naturally gravitate toward a mathematically well-defined universal filter code that captures fundamental visual processing operations.

3.2 Understanding the Eight Filters

This subsection investigates into the functional characteristics of the eight filters identified through our systematic greedy search, and which have proven to be very effective across various datasets. By analyzing these filters, we aim to understand their resemblance to traditional image-processing operators and their potential roles in effective feature extraction within the network.

Filters 1-4: These 4 filters in Figure exhibit characteristics reminiscent of central difference operators, commonly used for approximating Gaussian derivatives discretely. The arrangement and weights of these filters mimic the theoretical models used in edge detection and texture analysis.

Filters 5-6: These 2 filters strongly resemble 1st order Gaussian derivatives along the x and y axis. These filter contribute more pronounced spatial smoothing than previous filters. This characteristic enables these filters to capture broader and more varied textural information from the input images, potentially allowing for a better generalization across different visual contexts.

Filters 7: This filter resemble a 2-D discrete analogue of the Difference of Gaussians (DoG) due to its positive center with slightly negative surround. The DoG filter, often approximated by the Laplacian of the Gaussian in digital image processing, is crucial for blob detection and bar pattern recognition in images. These filter likely contribute to the model's ability to differentiate areas of rapid intensity change, enhancing edge and contour detection.

Filter 8: This filter closely aligns with a very fine-scaled Gaussian kernel. In image processing, Gaussian kernels are smoothing filters used to reduce noise and detail. This results in a blurred image that preserves edges better than uniform filters. Gaussian filters are mathematically proven to be the only function for scale-space representation.

Filters formal definition: For completeness, we provide below the formal definition of the continuous functions corresponding to the 2D Gaussian, the 2D derivative of the Gaussian along the x and the y axis, respectively, and the 2D difference of Gaussians (DoG, Laplacian, Mexican hat):

$$\text{Gaussian:} \quad G(x,y) \quad = \quad e^{-(x^2+y^2)/2\sigma^2} \qquad \Delta\text{Gaussian:} \quad \text{DoG}(x,y) = G_1(x,y) - G_2(x,y)$$

Table 2: **ImageNet Top-1 accuracy comparison between conventional trainings and our 8-filter constraint.** Models restricted to using only our 8 unique filters (plus learnable bias terms) achieve comparable accuracy to their fully-trained counterparts. The consistent performance across different architectures (ConvNeXtv2 and Hornet) demonstrates the universality of these fundamental filters.

	ConvNeXtv2				Hornet
Models	**Pico**	**Tiny**	**Base**	**Large**	**Tiny**
Number of Original Filters	2 944	6 624	18 048	27 072	11 488
Original model with FCMAE[1]	80.3%	82.9%	84.9%	85.8%	—
Original model	79.7%	82.5%	84.3%	84.5%	82.3%
Our 8 unique filters + bias	80.2%	82.7%	84.6%	85.4%	81.8%

[1] FCMAE (Fully Convolutional Masked Autoencoder Framework) is a heavy pretraining.

These formulations were used to construct the last four filters as depicted in Figure 5, with the exception of the DoG, for which its approximation, the Ricker wavelet, was used for simplicity. The reconstructed filters bear a strong resemblance to those discovered through our encoding and greedy search methods, validating our hypothesise. in function approximation and emphasizing the practical relevance of traditional image processing theories in modern deep learning architectures.

Functional Approximation and Construction: We reconstructed the lower four filters using theoretical formulas typically associated with these image processing techniques, depicted in Figure 5. The reconstructed filters bear a strong resemblance to those discovered through our encoding and greedy search methods, validating our approach in filter selection and emphasizing the practical relevance of traditional image processing theories in modern deep learning architectures.

4 Experiments

So far, we've identified 8 unique filters that, when used as linear approximations to replace the filters of trained models, maintain relatively stable accuracy despite this dramatic change in model parameters. These findings naturally lead to a key question: *Can models be successfully trained from initialization with just these 8 filter types kept frozen throughout training?* In this section, we present experimental evaluations on ImageNet and additional datasets to investigate this question.

4.1 ImageNet

The results in Table 1 demonstrate that model accuracy remains stable despite significantly reduced filter diversity. In these experiments, model filters are linear shifts of one of the 8 identified filters, mathematically expressed as $a(x + b)$. Given the architecture of DS-CNNs, the coefficient a can be transferred to the fully-connected layers following depthwise convolutions (the following pointwise layer), effectively simplifying the filters to $x + b$, where b acts as a learnable bias. This insight motivated us to train models from scratch using only these 8 fixed filters with learnable biases.

Training with Only 8 Frozen Filters

To investigate the effectiveness of our 8 candidate filters, we trained ConvNeXtv2 models from scratch, initializing each layer's filters with these 8 filters while allowing only the bias terms to be trainable. We followed the same 300-epoch training pipeline described in the original paper [29], with the critical difference that all convolutional filters remained frozen throughout training. Table 2 presents our results. Remarkably, the ConvNeXtv2 Tiny model with only 8 types of filters achieved an accuracy of 82.7%, merely 0.2% lower than the model trained with 6,624 trainable filters and FCMAE pretraining. Similarly, the smaller ConvNeXtv2 Pico model with 8 types of frozen filters reached 80.2% accuracy, just 0.1% below the model with 2,944 trainable filters.

To validate the generalizability of our findings across different architectures, we conducted an additional experiment with the Hornet model [23]—a DS-CNN with substantially different structure than the ConvNext family. The Hornet Tiny model with only our 8 filters achieved 81.8% accuracy compared to 82.3% for the original model, representing only a 0.5% drop. Notably, these 8 filters were derived exclusively from ConvNext models through greedy search on the ConvNeXtv2 Tiny model, yet transferred effectively to Hornet without modification.

It is worth emphasizing that despite our significant architectural modification, these experiments used the original training hyperparameters for each model. A dedicated hyperparameter search optimized for this fixed-filter approach could potentially enhance results further.

Table 3: **Cross-dataset evaluation** demonstrating the superiority of our universal filter approach on smaller datasets. The ConvNeXt Femto model restricted to using only 8 unique frozen filters outperforms both models trained from scratch and those using transferred ImageNet filters, on Oxford Pets and Oxford Flowers. We evaluate multiple model sizes (Atto, Femto, Pico, Tiny) for Flowers and Pets datasets to verify that our observed advantage is consistent across architectures of varying capacity and not merely an artifact of dataset size limitations.

Dataset	CIFAR10	STL-10	Oxford Flowers				Oxford Pets			
# Training Set Size	50000	5000	2040				3680			
Filters \ Model	Femto	Femto	Atto	Femto	Pico	Tiny	Atto	Femto	Pico	Tiny
Original (normal traning)	96.9	80.4	63.3	66.0	60.2	75.7	38.4	36.3	40.1	65.4
ImageNet Transferring	**97.1**	**83.2**	72.2	73.2	74.8	81.8	58.5	56.0	66.3	80.1
Our 8 Unique Filters	96.3	83.1	**77.8**	**77.7**	**77.2**	**85.1**	**66.5**	**66.4**	**72.8**	**81.8**

4.2 Other Datasets

To investigate the generalizability of our findings, we extend our experiments to other datasets and compare the performance of the ConvNeXt Femto across various settings.

Datasets and Settings. We evaluate the low filter variety on four datasets: CIFAR-10 [13], Flowers [21], Pets [22], and STL-10 [5]. These datasets have smaller scales compared to ImageNet, with the size of training sets ranging from 2040 to 50000 samples. We use the ConvNeXt Femto model as our base architecture for all datasets, and additionally use ConvNeXt Atto, Pico, and Tiny for the Flowers and Pets datasets. For a fair comparison, we train the model on all datasets for 300 epochs, following the training parameters from the ConvNeXt paper [20], and keep the training settings consistent across all runs and datasets. For each dataset, we first train the model to obtain baseline accuracy. We then evaluate two filter initialization strategies: (1) depthwise filters transferred from a ConvNeXt model pretrained on ImageNet, and (2) eight frozen filter types trained from scratch. Table 3 shows the resulting accuracies for each setting.

Results. The results demonstrate the effectiveness of using the 8 frozen filters across different datasets. Notably, the performance improvement becomes more pronounced as the dataset size decreases. For the Flowers and Pets datasets, the frozen 8 filter types achieve remarkable improvements of up to 11% and 34.5%, respectively, compared to the baseline model. Interestingly, on these smaller datasets, the eight frozen filter types even outperform the transferred filters from the model trained on ImageNet. To further evaluate and verify the performance of our 8 filters on these datasets, we used other sizes of the ConvNeXt model, the results of which showed consistent superior performance on all sizes.

This observation suggests that the carefully selected filter types capture fundamental patterns that are highly relevant to the task at hand, even when the dataset size is limited. This finding has significant implications for scenarios where training data is scarce or computational resources are limited.

5 Conclusions

This paper extends the "Master Key Filters Hypothesis" by identifying a set of just 8 filters. While conventional DS-CNNs employ thousands of distinct trained filters, our analysis reveals these filters predominantly converge to linear shifts (ax+b) of one of the filters in our discovered set. This finding significantly narrows the theoretical space proposed in previous work. The discovered filters closely match established mathematical forms: Difference of Gaussians, Gaussians, and their derivatives, creating a bridge between classical computer vision theory and modern deep learning practice. This correspondence to structures found in both scale-space theory and mammalian visual systems suggests DS-CNNs inherently rediscover optimal operators aligned with natural image statistics.

Our systematic experiments demonstrate that networks initialized with these 8 frozen filters achieve over 80% ImageNet accuracy. Particularly noteworthy is the superior performance of our filters on smaller datasets, where models initialized with our filters outperform even ImageNet transfer learning. This suggests that these filters encode fundamental visual processing primitives that transcend specific datasets and visual domains, offering a novel approach to transfer learning.

Future Work may explore direct optimization approaches to refine our master key filter set. While our 8 filters demonstrate impressive performance, systematic optimization might further improve their effectiveness or reduce their number. The observed constraint on filter diversity invites us to rethink the fundamental principles governing these architectures, potentially leading to insights about the complementary roles of depthwise spatial filters and pointwise channel mixing layers and opening opportunities for novel architecture designs.

References

[1] Zahra Babaiee, Peyman Kiasari, Daniela Rus, and Radu Grosu. Unveiling the unseen: Identifiable clusters in trained depthwise convolutional kernels. In *The Twelfth International Conference on Learning Representations*, 2023.

[2] Zahra Babaiee, Peyman M. Kiasari, Daniela Rus, and Radu Grosu. The master key filters hypothesis: Deep filters are general. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2025.

[3] Hangbo Bao, Li Dong, Songhao Piao, and Furu Wei. Beit: Bert pre-training of image transformers. In *International Conference on Learning Representations*, 2022.

[4] Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. Electra: Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations*, 2020.

[5] Adam Coates, Andrew Y. Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 215–223, 2011.

[6] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 702–703, 2020.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '16, pages 770–778. IEEE, June 2016.

[8] Torsten Hoefler, Dan Alistarh, Tal Ben-Nun, Nikoli Dryden, and Alexandra Peste. Sparsity in deep learning: Pruning and growth for efficient inference and training in neural networks, 2021.

[9] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, Quoc V. Le, and Hartwig Adam. Searching for mobilenetv3. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1314–1324. IEEE, 2019.

[10] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *CoRR*, abs/1704.04861, 2017.

[11] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4700–4708. IEEE, 2017.

[12] Jan J Koenderink. The structure of images. *Biological cybernetics*, 50(5):363–370, 1984.

[13] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical Report TR-2009, University of Toronto, Toronto, Canada, 2009.

[14] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Fractalnet: Ultra-deep neural networks without residuals. In *International Conference on Learning Representations*, 2017.

[15] Lucas Liebenwein, Cenk Baykal, Harry Lang, Dan Feldman, and Daniela Rus. Provable filter pruning for efficient neural networks, 2020.

[16] Tony Lindeberg. *Scale-space theory in computer vision*, volume 256. Springer Science & Business Media, 2013.

[17] Tony Lindeberg. Normative theory of visual receptive fields. *Heliyon*, 7(1):e05897, 2021.

[18] Tony Lindeberg. Covariance properties under natural image transformations for the generalised gaussian derivative model for visual receptive fields. *Frontiers in Computational Neuroscience*, 17, June 2023.

[19] Tony Lindeberg. Approximation properties relative to continuous scale space for hybrid discretizations of gaussian derivative operators, 2024.

[20] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[21] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *2008 Sixth Indian Conference on Computer Vision, Graphics and Image Processing*, pages 722–729, 2008.

[22] Omkar M. Parkhi, Andrea Vedaldi, Andrew Zisserman, and C. V. Jawahar. Cats and dogs. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3498–3505, 2012.

[23] Yongming Rao, Wenliang Zhao, Yansong Tang, Jie Zhou, Ser-Lam Lim, and Jiwen Lu. Hornet: Efficient high-order spatial interactions with recursive gated convolutions. *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[24] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[25] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2818–2826, 2016.

[26] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. *CoRR*, abs/1905.11946, 2019.

[27] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers and distillation through attention. In *International Conference on Machine Learning*, pages 10347–10357, 2021.

[28] Asher Trockman, Devin Willmott, and J Zico Kolter. Understanding the covariance structure of convolutional filters. In *The Eleventh International Conference on Learning Representations*, 2023.

[29] Sanghyun Woo, Shoubhik Debnath, Ronghang Hu, Xinlei Chen, Zhuang Liu, In So Kweon, and Saining Xie. Convnext v2: Co-designing and scaling convnets with masked autoencoders. *arXiv preprint arXiv:2301.00808*, 2023.

[30] R.A. Young, R.M. Lesperance, and W.W. Meyer. The gaussian derivative model for spatial-temporal vision: I. cortical model. *Spatial vision*, 14(3-4):261–319, 2001.

[31] Richard A. Young. The gaussian derivative model for spatial vision: I. retinal mechanisms. *Spatial Vision*, 2(4):273 – 293, 1987.

[32] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6023–6032, 2019.

[33] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018.

A Technical Appendices and Supplementary Material

A.1 Experimental Settings

Table 4: Training (t) and fine-tuning (ft) hyperparameters used in Section 4.2 experiments for ConvNeXtv2 Tiny model, taken from [29].

config	value
optimizer	AdamW
base learning rate	8e-4
weight decay	0.05
optimizer momentum	$\beta_1, \beta_2 = 0.9, 0.999$
layer-wise lr decay [4, 3]	0.9
batch size	1024
learning rate schedule	cosine decay
warmup epochs	(t) 40, (ft) 3
training epochs	(t) 300, (ft) 100
augmentation	RandAug (9, 0.5) [6]
label smoothing [25]	0.1
mixup [33]	0.8
cutmix [32]	1.0
drop path [14]	0.2
head init [27]	0.001
ema	0.9999

Table 5: Training (t) and fine-tuning (ft) hyperparameters used in Section 4.2 experiments for ConvNeXtv2 Pico model, taken from [29].

config	value
optimizer	AdamW
base learning rate	2e-4
weight decay	0.3
optimizer momentum	$\beta_1, \beta_2 = 0.9, 0.999$
layer-wise lr decay [4, 3]	0.9
batch size	1024
learning rate schedule	cosine decay
warmup epochs	0
training epochs	(t) 600, (ft) 100
augmentation	RandAug (9, 0.5) [6]
label smoothing [25]	0.2
mixup [33]	0.3
cutmix [32]	0.3
drop path [14]	0.0
head init [27]	0.001
ema	0.9999

A.2 8 Master Key Filters

In Figure 4 we provide the full numerical values of the 8 discovered universal filters, each of size 7×7. These filters were derived from a greedy search over encoded depthwise filters, as described in Section 3.1, and used in all experimental evaluations (Sections 4.1 and 4.2).

A.3 Experimental Compute Resources

We used 2 NVIDIA TITAN RTX GPUs for experiments on other datasets. For ImageNet training and fine-tuning we used 8 NVIDIA TITAN RTX GPUs.

Table 6: Information of Datasets used in the study and sample sizes, in training set size descending order.

Dataset	**Classes**	**Train Samples**	**Test Samples**
ImageNet	1000	1.2 million	50,000
CIFAR-10 [13]	10	50,000	10,000
STL-10 [5]	10	5,000	8,000
Oxford-IIIT Pets [22]	37	3,680	3,369
Oxford 102 Flowers [21]	102	2,040	6,149

Table 7: Training hyperparameters used in Section 4.2 experiments. The setting is taken from ConvNeXt [20].

config	**value**
optimizer	AdamW
base learning rate	4e-3
weight decay	0.05
optimizer momentum	$\beta_1, \beta_2 = 0.9, 0.999$
batch size	4096
training epochs	300
learning rate schedule	cosine decay
warmup epochs	50
warmup schedule	linear
layer-wise lr decay	None
randaugment	(9, 0.5)
mixup	0.8
cutmix	1.0
random erasing	0.25
label smoothing	0.1
layer scale	1e-6
head init scale	None
gradient clip	None

(a) Filter 1

-0.01	-0.02	-0.01	-0.00	-0.01	-0.02	-0.01
-0.02	-0.02	-0.00	0.00	-0.01	-0.02	-0.01
-0.01	-0.02	0.01	-0.11	-0.00	-0.01	-0.01
-0.03	-0.05	-0.09	-0.23	-0.06	-0.05	-0.03
-0.03	-0.06	0.02	0.94	0.04	-0.06	-0.03
-0.02	-0.02	0.00	0.12	0.01	-0.02	-0.02
-0.02	-0.02	0.01	0.09	0.00	-0.02	-0.02

(b) Filter 2

-0.00	-0.01	-0.02	-0.05	-0.04	-0.02	-0.00
-0.02	-0.02	-0.02	-0.04	-0.03	-0.02	-0.03
-0.02	-0.00	-0.01	-0.06	0.06	-0.01	-0.01
0.00	0.04	-0.06	-0.46	0.85	0.13	0.07
0.00	0.01	0.01	-0.12	0.07	0.02	0.01
-0.01	-0.01	-0.01	-0.05	-0.03	-0.01	-0.01
0.00	-0.01	-0.01	-0.04	0.00	-0.01	0.00

(c) Filter 3

-0.03	-0.02	-0.02	0.07	-0.02	-0.03	-0.03
-0.03	-0.02	0.01	0.14	0.01	-0.02	-0.03
-0.03	-0.04	0.10	0.88	0.11	-0.05	-0.04
-0.02	-0.02	-0.08	-0.36	-0.09	-0.03	-0.03
-0.02	-0.00	-0.05	-0.14	-0.05	-0.01	-0.02
-0.01	-0.01	0.01	0.01	0.00	-0.01	-0.01
-0.01	0.00	0.00	0.01	0.01	0.00	-0.00

(d) Filter 4

-0.04	-0.03	-0.02	-0.01	0.00	-0.00	-0.01
-0.04	-0.01	-0.04	-0.01	0.03	0.01	-0.01
-0.01	0.00	0.03	-0.05	0.00	0.02	0.01
0.04	0.08	0.87	-0.35	-0.30	-0.00	-0.00
-0.02	0.00	0.05	-0.01	-0.05	-0.00	-0.00
-0.03	-0.01	-0.01	0.00	0.00	0.00	-0.02
-0.04	-0.02	-0.01	-0.01	-0.00	-0.00	-0.00

(e) Filter 5

0.05	0.02	0.04	0.01	-0.04	-0.02	-0.07
0.04	0.03	0.05	0.02	-0.02	-0.01	-0.07
0.10	0.09	0.19	0.02	-0.17	-0.06	-0.09
0.20	0.20	0.54	-0.03	-0.53	-0.20	-0.22
0.09	0.08	0.19	0.01	-0.22	-0.09	-0.11
0.04	0.03	0.07	0.01	-0.04	-0.02	-0.07
0.05	0.02	0.05	-0.00	-0.04	-0.03	-0.07

(f) Filter 6

-0.07	-0.05	-0.08	-0.16	-0.07	-0.04	-0.06
-0.03	-0.01	-0.06	-0.14	-0.04	0.00	-0.03
-0.03	-0.04	-0.22	-0.47	-0.22	-0.03	-0.04
-0.01	-0.01	0.01	0.02	0.01	-0.00	0.00
0.02	0.03	0.20	0.68	0.20	0.02	0.03
-0.00	0.02	0.06	0.16	0.05	0.01	0.01
0.02	0.03	0.05	0.14	0.06	0.03	0.04

(g) Filter 7

-0.01	-0.01	-0.01	-0.02	-0.02	-0.00	-0.01
-0.01	-0.00	-0.02	-0.05	-0.01	-0.00	-0.01
-0.01	-0.01	-0.04	-0.06	-0.05	-0.01	-0.01
-0.01	-0.03	-0.01	0.98	-0.02	-0.04	-0.02
-0.01	-0.01	-0.05	-0.07	-0.06	-0.02	-0.02
-0.01	-0.01	-0.01	-0.05	-0.01	-0.00	-0.01
-0.01	-0.01	-0.02	-0.03	-0.02	-0.01	-0.01

(h) Filter 8

-0.04	-0.04	-0.04	-0.02	-0.04	-0.03	-0.04
-0.04	-0.03	-0.04	-0.04	-0.04	-0.03	-0.04
-0.04	-0.04	-0.02	0.16	-0.01	-0.04	-0.04
-0.02	-0.04	0.16	0.92	0.15	-0.04	-0.02
-0.04	-0.05	-0.03	0.15	-0.03	-0.05	-0.04
-0.04	-0.03	-0.04	-0.04	-0.04	-0.03	-0.04
-0.04	-0.04	-0.04	-0.02	-0.04	-0.03	-0.04

Figure 6: The 8 filters.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The claims in abstract and introduction are empirically supported in Sections 3 and 4 through experiments and filter visualizations.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper discusses limitations such as the need for further optimization of the 8 filters (Section 5).

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: The paper is primarily empirical and does not include formal theorems or proofs, though it provides equations for filter approximation (e.g., Equation 1 and 2 in Section 3.1).

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: The paper details the models used, datasets, training durations (300 epochs), and evaluation metrics. It also outlines the autoencoder and greedy search method in detail (Sections 3.1 and 4.1).

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: The paper uses public models and datasets. The scripts used in the paper will be available on a public Github repository after publication.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: The paper specifies training datasets, model sizes, training duration, and evaluation methods. In all experiments with each model, we used the exact same parameters sourced from original paper. See appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: Since most experiments are performed on ImageNet, it would be computationally unaffordable to do multiple rounds of training.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: See appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: This work follows the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: There is no societal impact of the work performed.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: This work does not involve models or data with high risk of misuse, such as generative models or private data.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All datasets and models used are cited.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We present 8 master key filters and provide them in the appendix.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve human subjects or crowdsourcing.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: No research involving human participants was conducted.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: This research does not involve LLMs as part of the core methodology.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.