International Journal of Computer Vision (2026) 134:461
https://doi.org/10.1007/s11263-026-03005-9

Fix Your Downsampling ASAP! Aliasing and Sinc Artifact Free Pooling in the Fourier Domain

Julia Grabinski[1,2] · **Steffen Jung**[2] · **Janis Keuper**[3] · **Margret Keuper**[2,4]

Received: 14 April 2025 / Accepted: 11 August 2026
© The Author(s) 2026

Abstract

Convolutional Neural Networks (CNNs) are successful in various computer vision tasks. From an image and signal processing point of view, this success is counter-intuitive, as the inherent spatial pyramid design of most CNNs is apparently violating basic signal processing laws, i.e. the *Sampling Theorem* in their downsampling operations. This issue has been broadly neglected until recent work in the context of adversarial attacks and distribution shifts showed that there is a strong correlation between the vulnerability of CNNs and aliasing artifacts induced by bandlimit-violating downsampling. As a remedy, we propose an alias-free downsampling operation in the frequency domain, denoted *Frequency Low Cut Pooling (FLC Pooling)* which we further extend to *Aliasing and Sinc Artifact-free Pooling (ASAP)*. ASAP is alias-free and removes further artifacts from sinc-interpolation. Our experimental evaluation on ImageNet-1k, ImageNet-C and CIFAR datasets on various CNN architectures demonstrates that networks using FLC Pooling and ASAP as downsampling methods learn more stable features as measured by their robustness against common corruptions and adversarial attacks, while maintaining a clean accuracy similar to the respective baseline models.

Keywords Image processing and computer vision · Sampling · Anti-aliasing · Fourier theory

1 Introduction

Most Convolutional Neural Network (CNN) architectures use a combination of small convolutional kernels and downsampling to increase the network's receptive field while keeping the computational costs low. However, standard downsampling methods such as MaxPooling, AveragePooling, or Convolution with a stride of two suffer from a significant drawback: their susceptibility to aliasing (Grabinski et al., 2022b; Zhang, 2019; Zou et al., 2020; Grabinski et al., 2022c), which has been shown to correlate with the network's vulnerability to distribution shifts (Zhang, 2019) and adversarial attacks (Grabinski et al., 2022b; Li et al., 2021). Based on these results, we investigate the manner in which 2D signals, in case of CNNs' input images and feature maps, are downsampled and how this results in undesired artifacts. So far, prior research mainly focused on aliasing artifacts (Grabinski et al., 2022b; Hossain et al., 2023; Zhang, 2019; Zou et al., 2020; Grabinski et al., 2022c), mostly proposing the use of blur kernels for mitigation (Zhang, 2019; Zou et al., 2020). However, these approaches are neither capable of removing aliasing completely, nor address other types of

Communicated by Yasuyuki Matsushita.

✉ Julia Grabinski
julia.grabinski@visinf.tudarmstadt.de

Steffen Jung
steffen.jung@uni-mannheim.de

Janis Keuper
janis.keuper@hs-offenburg.de

Margret Keuper
margret.keuper@uni-mannheim.de

[1] Visual Inference Lab, TU Darmstadt, University of Mannheim, Hochschulstr. 10, 64289 Darmstadt, Hessen, Germany

[2] Department of Machine Learning, University of Mannheim, B 6, 26, 68159 Mannheim, Baden-Württemberg, Germany

[3] Institute for Machine Learning and Analytics, Offenburg University, Badstraße 24, 77652 Offenburg, Baden-Württemberg, Germany

[4] Department of Computer Vision and Machine Learning, Max Planck Institute for Informatics, Saarland Informatics Campus, 66123 Saarbrücken, Saarland, Germany

Published online: 05 October 2026

spectral leakage artifacts related to downsampling in CNNs.

This paper is a consolidation and significant extension of our previously accepted ECCV 2022 publication (Grabinski et al., 2022a) which introduces **F**requency **L**ow **C**ut (FLC) Pooling (Grabinski et al., 2022a), an aliasing-free method for the downsampling of CNN feature maps in the frequency domain. The original conference paper focused on avoiding catastrophic overfitting during adversarial training by alias-free downsampling. In addition, we here conduct an in-depth study of the properties of CNN feature maps after frequency domain downsampling and propose a further improved method, termed **Aliasing and Sinc Artifact-free P**ooling (ASAP). ASAP provides enhanced feature extraction stability, leading to further improved inherent model robustness against common corruptions (Hendrycks & Dietterich, 2019), and adversarial attacks (Goodfellow et al., 2015; Kurakin et al., 2017). Further, all our proposed methods which guarantee aliasing-free downsampling can increase stability during adversarial training. However, our new evaluations show that even aliasing-free pooling methods like FLC Pooling can still be prone to other spectral corruptions, visible as ringing artifacts in the spatial domain. An example of such artifacts can be observed in Fig. 1, specifically in the third row and third column, where the structures near the zebra's head are repeated in a rippled manner. In contrast, the standard downsampling method used in many CNN architectures, like MaxPooling, completely distorts the zebra's structure (first row Fig. 1), and strided convolutions result in severe aliasing artifacts, visible as grid structures on the zebra's fur (second row Fig. 1). Consequently, it is crucial to reevaluate the choice of downsampling techniques employed within CNNs. In order to reduce artifacts, ASAP consolidates FLC Pooling by using a Hamming window in the frequency domain and considering appropriate padding.

Our contributions are as follows:

- We analyse and extend **F**requency **L**ow **C**ut Pooling (FLC Pooling) for aliasing-free downsampling without additional hyperparameters.
- We show that even aliasing-free downsampling can be prone to corruptions in the frequency domain, namely sinc interpolation artifacts.
- Consequently, we introduce **Aliasing and Sinc Artifact-free P**ooling, short ASAP, allowing for more stable feature extraction.
- To validate the robustness of FLC Pooling and ASAP, we empirically evaluate against adversarial attacks (Goodfellow et al., 2015; Kurakin et al., 2017; Croce & Hein, 2021) as well as common corruptions incorporated in ImageNet-C (Hendrycks & Dietterich, 2019).
- Moreover, we combine FLC Pooling and ASAP with simple FGSM (Goodfellow et al., 2015) and PGD (Kurakin et al., 2017) adversarial training and show that the models achieve favorable performance in terms of clean and robust accuracy by avoiding catastrophic overfitting.

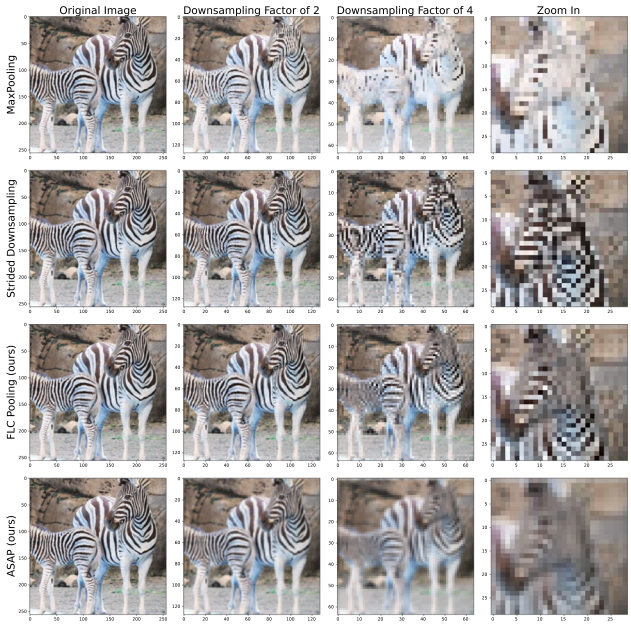

Fig. 1 The influence of different downsampling methods illustrated with natural images. The first and second rows show the commonly used MaxPooling and strided downsampling. In the third and fourth row, we apply our FLC Pooling and aliasing and sinc artifact-free pooling (ASAP), respectively. While MaxPooling does not preserve the image structure well, FLC Pooling and ASAP retain structural and spatial information much better. Strided downsampling also preserves, for example, the zebra's structure, yet suffers from severe aliasing artifacts, visible as grid artifacts on the zebra's fur. These artifacts are removed with our FLC Pooling and ASAP. While, FLC Pooling exposes sinc artifacts, visible for example around the zebras head after the first two downsampling stages, such artifacts are removed with ASAP (here, we show $\text{ASAP}_{\text{stbl}}$, details are given in Sect. 3)

2 Related Work

2.1 Aliasing in CNNs

The issue of aliasing effects in CNN-based neural networks has been extensively explored in the literature from various perspectives. (Zhang, 2019) enhance the shift-invariance of CNNs by incorporating anti-aliasing filters implemented as convolutions with fixed blur kernels. Building on this work, shift invariance is further improved in Zou et al. (2020) by utilizing learned blurring filters instead of predefined kernels. In Li et al. (2021), the pooling operations leverage the low-frequency components of wavelets to mitigate aliasing and enhance robustness against common image corruptions. Depth adaptive blurring filters before pooling are proposed in Hossain et al. (2023), along with

an anti-aliasing activation function. The importance of anti-aliasing is also recognized in the field of image generation. The use of blurring filters via Kaiser window to reduce aliases during image generation in generative adversarial networks (GANs) is suggested in Karras et al. (2021), while Durall et al. (2020) and Jung and Keuper (2021) incorporate additional loss terms in the frequency domain to address aliasing. In Grabinski et al. (2022b), we empirically demonstrate that adversarially robust models exhibit lower levels of aliasing in their downsampling layers compared to non-robust models, using a proposed aliasing measure. Motivated by these findings, we propose an aliasing-free downsampling method in the frequency domain for stable feature extraction and to prevent catastrophic overfitting. In extension, we present another approach achieving not only aliasing-free downsampling but also sinc interpolation artifact-free downsampling, which further increases the stability of our extracted features.

2.2 Spectral Leakage Artifacts

In contrast to the specific case of aliasing, spectral leakage artifacts have so far received less attention in the context of CNNs. A common case is the induction of sinc interpolation artifacts, which often arise when applying finite windows to periodic signals. These artifacts manifest as ringing artifacts in the spatial domain, as described in Gonzales and Wintz (1987), and are associated with the Gibbs phenomenon (Hamming & Stearns, 1979). To mitigate these spectral leakage artifacts, various window functions can be employed, as discussed in Prabhu (2014).

Window functions play a crucial role in the spectral analysis for biomedical image processing Jähne (2005); Semmlow and Griffel (2021). More recently, spectral leakage artifacts within CNNs have been studied in Tomen and Van Gemert (2021), showing that small spatial kernels can contribute to such artifacts. Consequently, they propose to learn larger spatial kernels while also applying a Hamming window to the convolution weights. In contrast, we apply window functions in the frequency domain.

2.3 Robustness

We assess the feature stability of models relying on two test scenarios, common corruptions and adversarial attacks.

Common corruptions One aspect to assess the robustness of CNNs involves evaluating their resilience against common corruptions caused by factors such as diverse weather conditions, varying lighting conditions, or subpar camera quality. To measure this kind of robustness, the widely recognized ImageNet-C dataset is utilized (Hendrycks & Dietterich, 2019). This dataset aims to simulate real-world scenarios through synthetic means. Approaches that improve this form of robustness often employ data augmentation techniques (Hendrycks et al., 2020; Cubuk et al., 2018), include shape biasing (Geirhos et al., 2018), or combine adversarial training with augmentations (Kireev et al., 2022). Vasconcelos et al. (2021) use non-trainable lowpass filters to reduce aliasing in the network, increasing the network's robustness against common corruptions. In contrast, our approach completely eliminates aliasing and mitigates sinc-interpolation artifacts, resulting in more stable features and improved robustness against common corruptions.

Adversarial attacks Adversarial examples are crafted to deceive a network into making incorrect decisions and expose network-specific vulnerabilities. In a white-box attack scenario (Goodfellow et al., 2015; Kurakin et al., 2017; Croce & Hein, 2021), the attacker has full access to the network's architecture and parameters, while in a black-box attack (Andriushchenko et al., 2020), the attacker only has access to the network's outputs.

One well-known white-box attack is the Fast Gradient Sign Method (Goodfellow et al., 2015), FGSM, which is an efficient single-step attack. The basic idea behind FGSM is to compute the gradient of the model loss with respect to the input image, subsequently shifting the pixel values by a small step of size ϵ in the direction of the sign of this gradient to maximize the model prediction error. A larger ϵ increases the attack's success rate but makes the image distortion more noticeable to humans, highlighting a direct trade-off between adversarial strength and imperceptibility. In FGSM the shift by ϵ is only applied once. Thus, FGSM is fast to compute, yet not as effective as other methods that use multiple optimization steps, e.g. as in the white-box Projected Gradient Descent (PGD) (Kurakin et al., 2017) which essentially applies FGSM iteratively over multiple smaller steps while projecting the resulting perturbation back into a valid ϵ-bound. Other alternatives include black-box attacks such as Squares (Andriushchenko et al., 2020). AutoAttack (Croce & Hein, 2020) is an ensemble of different attacks, including an adaptive version of PGD, APGD (Croce & Hein, 2021), and is widely used to benchmark adversarial robustness due to its strong performance (Croce et al., 2021). In this work, we consider adversarial attacks, including FGSM, PGD, APGD and AutoAttack, as a probe of the model's vulnerability. We use low-budget, small-epsilon attacks to showcase our increased feature stability without adversarial training. Additionally, to compare with state-of-the-art adversarial training, we assess performance under strong attacks like AutoAttack and high-epsilon attacks.

2.4 Adversarial Training

Various defense methods have been developed to establish robustness against adversarial attacks. A key method for improving robustness is adversarial training (AT) (Goodfellow et al., 2015; Rony et al., 2019; Wong et al., 2020),

where networks are exposed to adversarial examples during training. These adversarial samples are incorporated by introducing an additional loss term during network training (Engstrom et al., 2019; Zhang et al., 2019). Other techniques include utilizing additional training data (Carmon et al., 2019; Sehwag et al., 2021), particularly the *ddpm* dataset (Gowal et al., 2021b; Rade & Moosavi-Dezfooli, 2021; Rebuffi et al., 2021), which consists of one million extra samples for CIFAR-10 and is generated using the model proposed by Ho et al. (2020). Data augmentation has also proven to enhance adversarial robustness (Gowal et al., 2021a), and combining it with weight averaging further improves performance (Rebuffi et al., 2021). Some more advanced techniques involve adding specifically generated images to the training dataset (Gowal et al., 2021b), additional frequency regularizations (Lukasik et al., 2023) or adding spatial anti-aliasing filters encompassing downsampling layers and activation functions (Rodríguez-Muñoz & Torralba, 2022).

However, a major drawback of most AT methods is the significant increase in computational resources required for training. Generating adversaries during training alone can increase training time by a factor of seven to fifteen (Kurakin et al., 2017; Wang et al., 2020; Wu et al., 2020; Zhang et al., 2019). Adding additional data or generating new images (Gowal et al., 2021b) further amplifies the computational burden. FLC Pooling and ASAP allow to train models efficiently with FGSM adversarial training.

Specifically, FGSM Goodfellow et al. (2015) is a single-step attack and therefore more efficient than more complex, multi-step methods like PGD (Kurakin et al., 2017). In common settings, the iterative process of PGD takes nearly nine times (Table 9) longer than FGSM training. However, FGSM training is susceptible to catastrophic overfitting (Wong et al., 2020). Catastrophic overfitting describes the phenomenon during adversarial training (e.g. with single-step FGSM) where the model overfits to the attack it is trained on, leading to increased susceptibility to other attacks (e.g. multi-step PGD). While we demonstrate a correlation between aliasing after downsampling and catastrophic overfitting in Grabinski et al. (2022b), in this work we propose downsampling in the frequency domain, specifically FLC Pooling and ASAP, as an option to reduce the risk of catastrophic overfitting. In consequence, we observe an empirical benefit on efficient adversarial training.

3 Aliasing- and Sinc Artifact-Free Pooling

First, we present an aliasing-free downsampling method, FLC Pooling, which completely removes aliasing artifacts during downsampling. In contrast, previous approaches (Hossain et al., 2023; Zhang, 2019; Zou et al., 2020) only reduce aliasing artifacts without fully removing them. Additionally, we further extend FLC Pooling by introducing Aliasing and Sinc Artifact-free Pooling, short ASAP, which addresses not only the well-known issue of aliasing, but also tackles sinc interpolation artifacts.

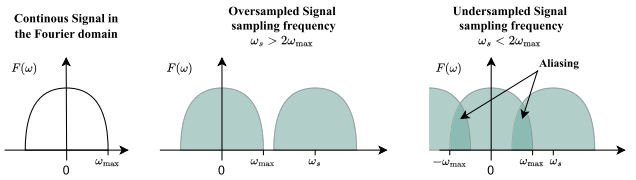

Fig. 2 Aliasing in the Fourier domain. Left: The frequency spectrum of a 1D signal with maximal frequency ω_{max}. After downsampling, replica of the signal appear at a distance proportional to the sampling rate. Center: The spectrum after sampling with a sufficiently large sampling rate. Right: The spectrum after under-sampling with aliases due to overlapping replica

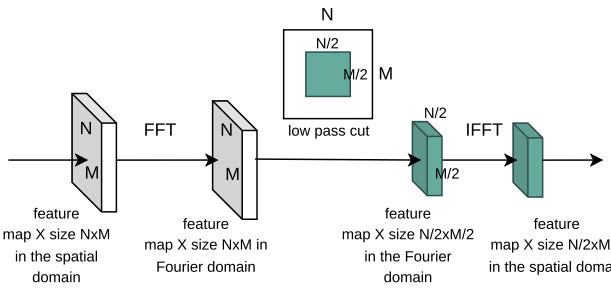

Fig. 3 Overview of our aliasing-free FLC Pooling. First, we transform the input with the FFT into the frequency domain and shift the low-frequency components in the center. Afterwards, we apply our Frequency Low Cut (FLC) to downsample aliasing-free. Lastly, we transform back into the spatial domain via IFFT

Aliasing Aliasing is a specific type of spectral leakage artifact that occurs when a signal is improperly sampled at a rate below twice the signal's bandwidth (Shannon, 1949). This leads to overlapping high-frequency components, making them indistinguishable from low-frequency components (as visualized in Fig. 2). These overlaps in the frequency domain become visible as grid-like artifacts in the spatial domain.

Aliasing-free downsampling Earlier methods reduce aliasing during downsampling via classical blurring operators in the spatial domain (Zhang, 2019; Zou et al., 2020). While those methods reduce aliasing, they can not entirely remove it due to theoretical sampling limitations and limited filter sizes in practice (see (Gonzalez & Woods, 2006) for details). In contrast, we perfectly remove aliases in CNNs' downsampling operations without adding additional hyperparameters. We directly address the downsampling operation in the frequency domain, where we can sample according to the Nyquist rate, i.e. remove all frequencies above $\frac{\text{samplingrate}}{2}$ and thus discard any potential aliases. Our proposed alias-free

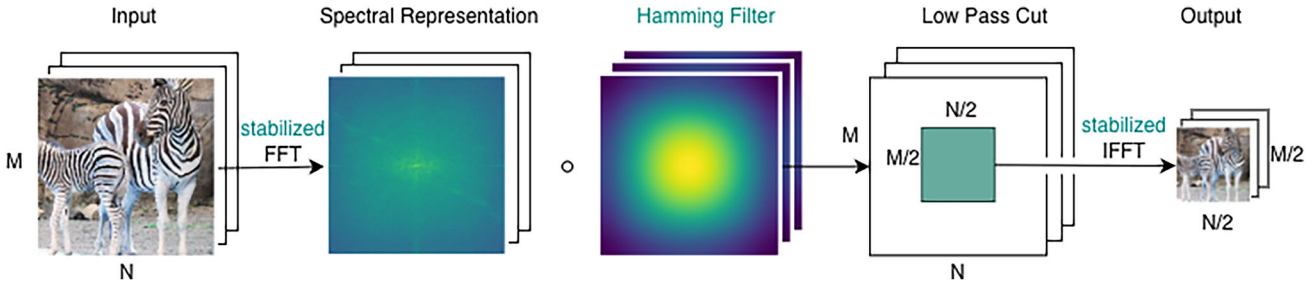

Fig. 4 Overview over our aliasing and sinc artifact-free downsampling, short ASAP. First, we transform the input with our stabilized FFT into the frequency domain and shift the low-frequency components in the center. Afterwards, we apply a Hamming filter on our frequency representation to prevent sinc interpolation artifacts. Further, we apply our Frequency Low Cut (FLC), which is similar to our FLC Pooling, to downsample aliasing-free. Lastly, we transform back into the spatial domain via our stabilized IFFT

downsampling operation, FLC Pooling (Grabinski et al., 2022a), is visualized in Fig. 3.

We first perform a Discrete Fourier Transform (DFT) of the feature maps f. Feature maps with height M and width N to be downsampled are then represented as

$$F(k,l) = \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m,n) e^{-2\pi j \left(\frac{k}{M}m + \frac{l}{N}n\right)}. \quad (1)$$

In the resulting frequency space representation $F(k,l)$, all coefficients of frequencies k,l, with $|k|$ or $|l| > \frac{\text{samplingrate}}{2}$ have to be set to 0 before downsampling. CNNs commonly downsample with a factor of two, such that the resulting sampling rate is $\frac{1}{2}$. Aliasing-free downsampling thus corresponds to removing coefficients where $|k|, |l| > \frac{1}{4}$. The remaining coefficients are then transformed back to the spatial domain via inverse DFT (Eq. 2):

$$f_d(\hat{m},\hat{n}) = \frac{1}{\hat{K}\hat{L}} \sum_{k=0}^{\hat{K}-1} \sum_{l=0}^{\hat{L}-1} F_d(k,l) e^{2\pi j \left(\frac{\hat{m}}{\hat{K}}k + \frac{\hat{n}}{\hat{L}}l\right)}. \quad (2)$$

Implementation The DFT returns an array F of complex numbers with size $K \times L = M \times N$, where the frequency $k,l = 0$ is stored in the upper left corner and the highest frequency is in the center. We thus shift the low frequency components into the center of the array via FFT-shift to get F_s and crop the frequencies below the Nyquist frequency as $\mathrm{F}_{sd} = \mathrm{F}_s[K' : 3K', L' : 3L']$ for $K' = \frac{K}{4}$ and $L' = \frac{L}{4}$, for all samples in a batch and all channels in the feature map. After the inverse FFT-shift, we obtain array F_d with size $[\hat{K}, \hat{L}] = [\frac{K}{2}, \frac{L}{2}]$, containing exactly all frequencies below the Nyquist frequency F_d, which we can transform back to the spatial domain via inverse DFT for the spatial indices $\hat{m} = 0 \ldots \frac{M}{2}$ and $\hat{n} = 0 \ldots \frac{N}{2}$.

We receive the aliasing-free downsampled feature map f_d with size $[\frac{M}{2}, \frac{N}{2}]$. Since we are cutting out the low frequencies, we denote this approach Frequency Low Cut Pooling, short FLC Pooling (Fig. 3).

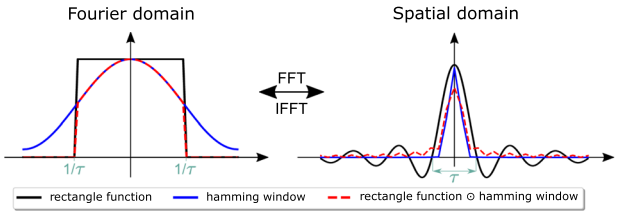

Fig. 5 Transformation of the rectangle function (black), the Hamming window (blue) and the point-wise multiplication of those two (red, dashed) from the Fourier domain (left) to the spatial domain (right). The rectangle function transforms to an infinite sinc function with infinite oscillations. In contrast, the Hamming window side lobes become near zero. Thus applying a Hamming window on the rectangle function in the Fourier domain leads to a suppressed version of the sinc in the spatial domain with lower oscillating side lobes

Sinc interpolation artifacts In Fig. 1 we visualize aliasing in the spatial domain. The leftmost image represents the original image before downsampling, the other images depict the downsampled versions (by a factor of two, four and eight in columns two, three and four, respectively). The second row (strided downsampling), downscaled without any anti-aliasing, exhibits prominent grid artifacts, particularly noticeable in the zebra's fur. In contrast, the image downsampled using FLC Pooling in the third row, shows an aliasing-free downscaled version.

However, after examination of the quality and stability of the FLC pooled image, we observe that it is still susceptible to sinc-interpolation artifacts, or *ringing artifacts*, predominantly as also visible around the zebra's head in column four. In the following, we briefly discuss these artifacts and present an approach to mitigate them.

FLC Pooling perfectly removes aliasing artifacts via the low pass cut in the frequency domain. It implicitly applies a point-wise multiplication with a rectangle function (defined

here between frequencies $-1/\tau$ and $1/\tau$):

$$rect(\omega, \tau) = \prod\left(\frac{\omega}{2/\tau}\right) = \begin{cases} 1 & \text{for } |\omega| \leq 1/\tau \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

with a length of $2/\tau$. Applying a rectangle function in the frequency domain can lead to leakage artifacts (also discussed in Tomen and Van Gemert (2021)). Specifically, the equivalent of a rectangle function in the Fourier domain is a sinc function in the spatial domain

$$f(x, \tau) = F^{-1}\left[\prod\left(\frac{\omega}{2/\tau}\right)\right] = \frac{1}{\tau\pi} sinc\left(\frac{x}{\tau}\right), \tag{4}$$

as visualized in Fig. 5.

As the rectangle function is applied as point-wise multiplication in the Fourier domain, the equivalent sinc-function in the spatial domain is applied as a circular convolution, i.e. a convolution with circular boundary conditions. This leads to (i) sinc-interpolation artifacts, also referred to as *ringing artifacts* in the spatial domain, and (ii) to wrap-around effects of the circular sinc-convolution at the boundaries of the feature map, where, for example, signal from the left boundary is convolved into the signal on the right.

Sinc interpolation artifact mitigation To achieve downsampling that is free of both aliasing and sinc interpolation artifacts, it is necessary to apply the rectangle function in the frequency domain. There it is common to apply a filter that smoothens the sharp edges of the rectangle function. Therefore, we propose the use of a Hamming window ($H(n)$), which is defined for 1D signals as:

$$H(n) = \alpha - (1 - \alpha) \cdot cos\left(\frac{2n\pi}{N}\right), \quad 0 \leq n \leq N \tag{5}$$

The 2D Hamming filter is defined as the outer product of two 1D Hamming filters, where $\alpha = 25/46$ and N represents the number of samples in the signal. However, unlike in Tomen and Van Gemert (2021), we do not utilize the Hamming filter as a window function in the spatial domain. Instead, we directly apply the Hamming filter in the frequency domain as a point-wise multiplication (Fig. 4).

The spatial representation of the Hamming window from the frequency domain is shown in Fig. 5 (right, blue line). The side slopes of the transformed Hamming window become near zero, effectively reducing interpolation artifacts. Further, we show in Fig. 5 the combination of the rectangle function and the Hamming window (red dashed line) in the Fourier domain (left) and the spatial domain (right). The Hamming window facilitates to suppress the oscillations of the sinc function, while remaining completely alias-free. Consequently, the possible artifacts from the circular convolution with the sinc function are reduced and artifacts that would come from the boundary of the feature map are suppressed as the side slope of the signal becomes near zero.

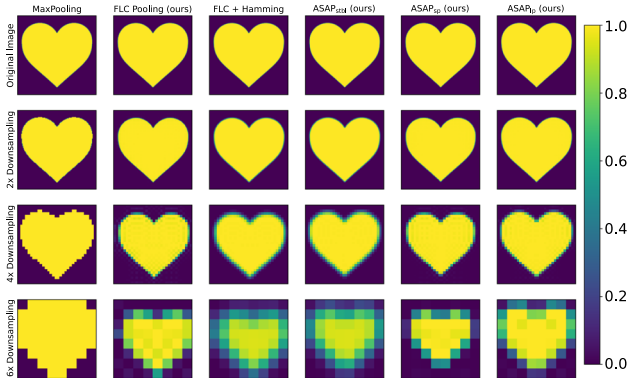

Fig. 6 We compare the influence of downsampling by different methods. The first column shows the commonly used MaxPooling. In the second column, we use FLC Pooling, and in the third column, we show FLC with additional Hamming window applied. In the last three columns, we show our ASAP variants, that include the Hamming window and different stabilizations.

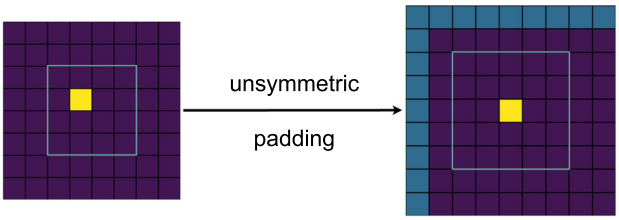

Fig. 7 We use unsymmetric padding to achieve an uneven input size. Thus, the frequency representation is symmetric and the DC component is centered. During the cut of the low frequency components the cut stays symmetric while this is not the case for the unpadded signal

Stabilized fast fourier transform (FFT) Further analysis shows that it is beneficial to stabilize the Fourier Transform. As shown in Fig. 6, columns two and three, FLC Pooling can introduce shifts in the image, that become clearly visible after multiple transform applications. We hypothesize that this originates from the fact that our signal is usually even-sized (even number of pixels in width and height), leading to non-centered frequency representations (see Fig. 7, left). The resulting asymmetric cut-out of the low-frequency components can be avoided by padding. Specifically, asymmetric padding can center the frequency representation, see Fig. 7, right.

In theory, padding can also serve as a mitigating factor against sinc interpolation artifacts. The rationale behind this is twofold. First, the padding compensates for possible ringing artifacts that fold in from the boundaries due to the assumption of periodicity and the application of the rectangular filter. As the padding is removed afterwards these ringing artifacts can thus be removed as shown in Fig. 8. Second, through the padding and transformation into the frequency

Table 1 Mean and standard deviation of the aliasing measure (Grabinski et al., 2022b) and power spectrum difference measured via KL divergence after downsampling with conventional and our downsampling methods over 1000 images of CIFAR-10

Name	Aliasing (↓)	Power spectrum difference (↓)
MaxPooling	0.26 ± 0.30	0.0113 ± 0.1019
Strided Conv	0.17 ± 0.17	0.0025 ± 0.0070
BlurPooling Zhang (2019)	0.17 ± 0.17	$\mathbf{9e{-}06 \pm 0.0008}$
ABlurPooling Zou et al. (2020)	0.22 ± 0.23	0.0007 ± 0.0115
Wavelet Pooling Li et al. (2020)	0.84 ± 0.30	0.0025 ± 0.0070
FLC Pooling (ours)	**0**	0.0036 ± 0.0107
$\text{ASAP}_{\text{stbl}}$ (ours)	**0**	0.0011 ± 0.0049
ASAP_{sp} (ours)	**0**	0.0006 ± 0.0075
ASAP_{lp} (ours)	**0**	0.0004 ± 0.0071

FLC Pooling as well as ASAP do not suffer from aliasing. Hence, aliasing is zero

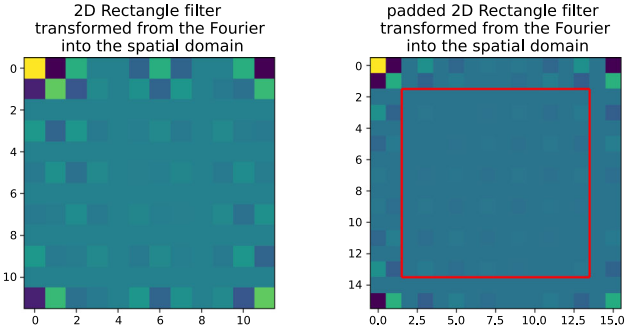

Fig. 8 Effect of padding (applied in the right) against sinc interpolation artifacts. As we only pad for the operation in the frequency domain, the padding is removed afterwards by a centered crop (red line). Thus, some of the dominant sinc interpolation artifacts are removed

domain, we artificially increased the resolution of the signal. Thus, when cutting at the Nyquist frequency of the higher resolution signal we artificially increase the value for $1/\tau$ resulting in a more narrow sinc in the spatial domain. In our analysis, we provide two options for padding. Small padding, denoted by ASAP_{sp}, only adds one line of zeros to the bottom and left. Large padding, denoted by ASAP_{lp}, pads by $\frac{\text{inputsize}}{2} - 1$ to the top and left. Both lead to centered representations (see Fig. 6). Larger padding could provide better results while smaller padding is more efficient. Yet, since the FFT algorithm is optimized for data in the array size of powers of two, even small padding can increase the compute costs when the size before padding is exactly a power of two.

Therefore, we evaluate an efficient heuristic to avoid the numerical artifacts from the row-first 2D FFTs: we transpose the feature map before every other FFT, such that row-first and column-first FFT are applied in an alternating manner. We denote this stabilization method as $\text{ASAP}_{\text{stbl}}$. We discuss the computation cost in Sect. 5.

4 Experiments

For evaluation, we first visualize aliasing artifacts following downsampling, as well as sinc-interpolation artifacts resulting from FLC Pooling in Subsection 4.1. Second, we assess the performance of state-of-the-art models trained using different downsampling techniques against both our FLC Pooling and ASAP in Subsection 4.2. This evaluation demonstrates that FLC Pooling and ASAP learn more stable features leading to improved robustness against common corruptions and low budget adversarial attacks. Third, we evaluate our FLC Pooling and ASAP networks in combination with adversarial training and demonstrate their ability to prevent catastrophic overfitting during FGSM adversarial training in Subsection 4.3. Lastly, we conduct ablation studies in Sect. 4.4.

4.1 Artifact Representation

Qualitative analysis in the spatial domain We first visually inspect downsampling artifacts for several downsampling stages (factor 2, 4, and 8) after downsampling by a factor of 8 in a toy example in Fig. 6. MaxPooling (first column) has the expected effect of disintegrating the spatial structure of the sample. In the bottom row of column two, the sinc-interpolation artifacts for FLC Pooling, discussed in Sect. 3, become visible: they appear as ringing artifacts. The same effect can be observed in Figs. 1 and 9.

To mitigate these artifacts, we apply a Hamming filter, as described in Sect. 3, suppressing these artifacts (four last columns of Fig. 6). A remaining, potentially undesired effect is the slight shift of the signal to the lower right, which is removed in the ASAP variants.

Analysis in the frequency domain Fig. 9 depicts the 1D power spectrum after downsampling for a more realistic example. The first column shows the original image with the full power spectrum. Each column presents a different

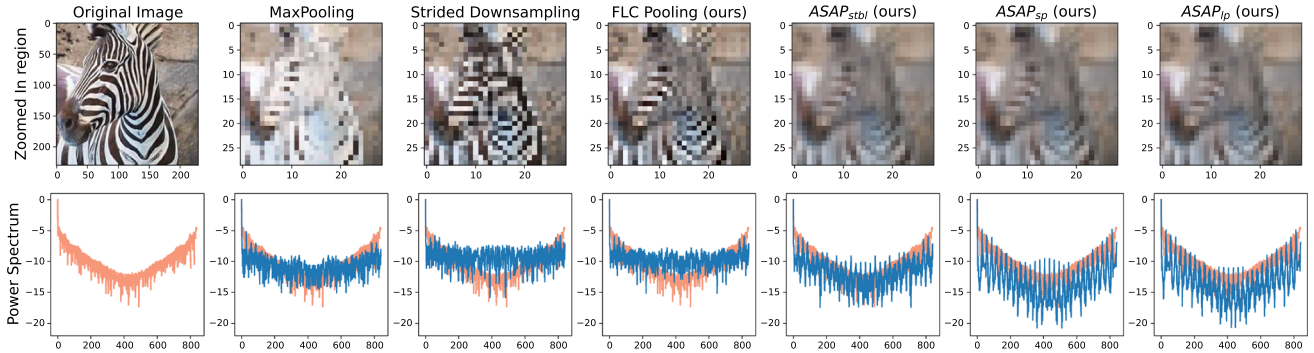

Fig. 9 Power spectra after different downsampling methods as well as the actual downsampled images. The first row shows the original images and the downsampled versions (downsampled by factor eight) with different pooling methods. After MaxPooling the zebra's fur structures are much less recognizable. When simple downsampling via stride is applied, grid structures appear, and we observe aliasing artifacts. Using FLC Pooling removes these aliasing artifacts. However, ringing artifacts surrounding the zebra's head become visible. Only ASAP is able to downsample the image without artifacts. The second row presents the power spectrum (in log scale for the y-axis, x-axis presents the frequency bands) of the images. The first column represents the original power spectrum of the image. Underlying each power spectrum of the downsampled versions we plotted the spectrum of the original image in red. ASAP is the only method to achieve a similar power spectrum to the original image

downsampling technique, with the qualitative result after downsampling in the first row and its power spectrum in the second row. All our ASAP variants obtain a power spectrum similar to the one of the original image.

Quantitative analysis To quantify, we evaluate the aliasing measure proposed in Grabinski et al. (2022b) as well as the difference in power spectrum for different downsampling methods in Table 1. We use the pretrained weights for Zou et al. (2020); Zhang (2019); Li et al. (2020) and trained a model with standard stride according to the training schedule described in Sect. 4.2. Methods based on blurring before downsampling, including all variants of our proposed ASAP, maintain the power spectrum of the original image best. However, our FLC Pooling and all variants of our ASAP are the only entirely alias-free approaches. ASAP$_{lp}$ offers the most favorable trade-off, being alias-free and preserving the power spectrum well.

FFT stabilization Following Sect. 3, we apply the FFT by using our stabilization heuristic or additional padding. Figure 6 (columns four to six) depict the effect of using the stabilization heuristic by transposing (ASAP$_{stbl}$), large ASAP$_{lp}$ and small ASAP$_{sp}$ asymmetric padding before transforming into the frequency domain so that all representations are correctly centered. As additional padding potentially increases the computational cost of the FFT, leading to an increase in computational costs for ASAP$_{sp}$ by 8% and by a factor of 5.6 for ASAP$_{lp}$ on an NVIDIA A100 as reported in Table 2.

Table 2 Time evaluation of our additional padding compared to no padding. We evaluate the time for one execution over the full original validation set of CIFAR-10 input images in 100 independent runs over the validation set on an NVIDIA A100.

Padding size	Sec per Single execution	Sec per sequence
No Padding	0.0522 ± 0.0181	0.0738 ± 0.0195
Small Padding	0.0623 ± 0.0211	0.0794 ± 0.0185
Large Padding	0.2903 ± 0.0254	0.3782 ± 0.0282

Small padding only adds one line of zeros on the bottom and left. Large padding pads by $\frac{\text{inputsize}}{2} - 1$ to the top and left. A sequence includes three operations stacked, such that the input is downsampled by a factor of eight, while a single execution only downsamples by a factor of two

4.2 Feature Stability Without Adversarial Training

We evaluate the stability of the features learned by our proposed FLC Pooling and ASAP by considering different adversarial samples as well as common corruptions Hendrycks and Dietterich (2019). When models are trained without adversarial training, using low ϵ budgets makes most sense Lukasik et al. (2023). These models can not be expected to be robust against strong attacks but we merely want to probe the improved stability of their learned representation. We conduct experiments using two different datasets, ImageNet-1k Deng et al. (2009) and CIFAR-10 Krizhevsky and Hinton (2009).

For high-resolution data we use ImageNet-1k Deng et al. (2009) and trained one network per ASAP method (stabilized FFT, large and small padding). The baseline networks utilized the pre-trained weights provided by PyTorch. The weights for BlurPooling are provided by Zhang (2019) and

Table 3 Clean, robust and common corruptions accuracy (in percent) for several common models trained without adversarial training on ImageNet-1k.

Arch	Method	Acc@1	Acc@5	APGD	FGSM	Corr@1	Corr@5
ResNet-18	Baseline	69.56	89.09	0.01	21.20	34.37	54.66
	BlurPooling Zhang (2019)	71.38	90.12	0.07	21.78	35.97	56.48
	Wavelet Pooling Li et al. (2020)	71.29	90.12	0.01	23.10	37.53	58.34
	FLC Pooling (ours)	69.16	88.91	**0.32**	35.21	40.19	61.82
	$ASAP_{stbl}$ (ours)	69.53	89.11	0.31	36.44	40.33	61.99
	$ASAP_{lp}$ (ours)	71.18	89.93	0.31	**40.33**	43.14	64.63
	$ASAP_{sp}$ (ours)	**71.64**$^{\pm 0.10}$	**90.23**$^{\pm 0.07}$	0.29$^{\pm 0.05}$	40.21$^{\pm 0.42}$	**43.83**$^{\pm 0.24}$	**65.32**$^{\pm 0.27}$
ResNet-50	Baseline	75.85	92.88	0.07	36.00	40.80	61.18
	BlurPooling Zhang (2019)	77.19	93.38	0.24	39.44	43.03	63.60
	Wavelet Pooling Li et al. (2020)	76.56	92.95	0.13	38.45	42.13	62.79
	low-pass Vasconcelos et al. (2021)	77.50	–	–	–	30.00	–
	low-pass Vasconcelos et al. (2021) + RA	78.40	–	–	–	34.50	–
	low-pass Vasconcelos et al. (2021) + Swish + RA	**78.80**	–	–	–	35.10	–
	FLC Pooling (ours)	77.13	93.44	0.53	**59.05**	50.51	71.59
	$ASAP_{stbl}$ (ours)	77.12	93.45	0.67	57.72	50.76	71.87
	$ASAP_{lp}$ (ours)	78.11	94.00	0.67	58.37	53.34	74.00
	$ASAP_{sp}$ (ours)	78.54	**94.10**	**0.94**	56.84	**54.20**	**74.83**
ResNet-101	Baseline	77.25	93.54	0.04	38.66	46.09	67.01
	BlurPooling (Zhang, 2019)	78.15	94.03	0.25	42.76	46.92	67.70
	Wavelet Pooling Li et al. (2020)	78.06	93.96	0.04	46.11	48.49	69.16
	FLC Pooling (ours)	78.16	93.95	1.26	58.49	52.91	73.72
	$ASAP_{stbl}$ (ours)	78.11	94.12	1.32	**59.56**	53.13	73.99
	$ASAP_{lp}$ (ours)	79.07	94.35	1.57	58.99	55.82	76.19
	$ASAP_{sp}$ (ours)	**79.34**	**94.63**	**1.78**	58.06	**56.14**	**76.51**
WRN-50-2	Baseline	78.29	94.03	0.28	37.74	45.23	65.75
	BlurPooling Zhang (2019)	78.60	94.18	0.60	39.39	46.22	66.58
	FLC Pooling (ours)	79.67	94.74	**0.64**	**54.00**	48.48	69.22
	$ASAP_{stbl}$ (ours)	79.68	94.71	0.61	53.43	48.99	69.63
	$ASAP_{lp}$ (ours)	80.01	**94.98**	0.45	49.88	50.17	70.40
	$ASAP_{sp}$ (ours)	**80.31**	94.96	0.51	48.52	**50.93**	**70.98**
MobileNet-v2	Baseline	71.36	90.12	0.00	14.29	34.13	54.43
	BlurPooling Zhang (2019)	**72.47**	**90.69**	0.00	13.37	34.33	54.50
	Wavelet Pooling Li et al. (2020)	71.94	90.46	0.00	13.04	34.28	54.61
	FLC Pooling (ours)	66.81	87.72	0.26	25.21	34.70	55.64
	$ASAP_{stbl}$ (ours)	66.83	87.71	0.29	25.58	35.10	56.18
	$ASAP_{lp}$ (ours)	68.70	88.80	**0.42**	**28.37**	37.15	58.48
	$ASAP_{sp}$ (ours)	69.14	88.87	0.41	28.06	**38.34**	**59.98**

Attacks are performed with $\epsilon = 1/255$ to probe the stability of learned representations; the $\epsilon = 4/255$ typical for adversarially trained models makes all these models fail completely. We report the mean and std over three seeds for ResNet-18 for our $ASAP_{sp}$ and show that the observed improvements are not artifacts of seed selection. FLC Pooling and ASAP improve prediction stability under adversarial attacks; the absolute values are not usable robustness, which requires AT (Table 5). In addition, $ASAP_{sp}$ outperforms all methods on common corruptions
Bold values signifies the best entry/entries per row

Table 4 Mean accuracy (in percent) and standard deviation on clean samples, perturbed samples with FGSM Goodfellow et al. (2015) and PGD Kurakin et al. (2017) as well as corrupted samples Hendrycks and Dietterich (2019) for four different architectures (five different random seeds) trained without adversarial training on CIFAR-10.

Arch	Method	Acc@1	FGSM	PGD	Corruptions
ResNet-9	Baseline Lukasik et al. (2023)	94.29	59.58	53.04	–
	DCT Conv WD Lukasik et al. (2023)	93.18	59.25	56.08	–
	DCT Conv SD Lukasik et al. (2023)	93.09	59.87	56.89	–
	FLC Pooling (ours)	94.53 ± 0.11	**69.05 ± 0.22**	**66.60 ± 0.58**	**75.29 ± 0.75**
	$ASAP_{stbl}$ (ours)	**94.56 ± 0.16**	68.96 ± 0.61	65.81 ± 0.89	74.79 ± 0.64
	$ASAP_{lp}$ (ours)	94.52 ± 0.15	68.56 ± 0.52	66.06 ± 1.04	75.04 ± 0.73
	$ASAP_{sp}$ (ours)	94.55 ± 0.08	68.52 ± 0.53	65.28 ± 1.03	74.61 ± 0.73
ResNet-18	Baseline	93.03 ± 0.13	78.62 ± 0.28	72.49 ± 0.67	76.93 ± 0.45
	BlurPooling Zhang (2019)	**93.25 ± 0.17**	79.24 ± 0.32	75.23 ± 0.55	77.70 ± 0.54
	ABlurPooling Zou et al. (2020)	92.77 ± 0.15	**79.65 ± 0.52**	76.94 ± 0.79	76.59 ± 0.33
	WaveletPooling Li et al. (2020)	93.00 ± 0.06	79.15 ± 0.15	72.71 ± 0.55	78.40 ± 0.23
	DCT Conv WD Lukasik et al. (2023)	88.80 ± 0.16	60.53 ± 1.06	58.65 ± 1.21	73.88 ± 0.41
	DCT Conv SD Lukasik et al. (2023)	89.93 ± 0.10	61.37 ± 0.51	59.40 ± 0.74	75.84 ± 0.35
	Anti-Aliasing Rodríguez-Muñoz and Torralba (2022)	91.99 ± 0.07	78.33 ± 0.32	**77.32 ± 0.43**	78.52 ± 0.35
	FLC Pooling (ours)	93.12 ± 0.19	78.92 ± 0.26	74.17 ± 0.60	78.59 ± 0.29
	$ASAP_{stbl}$ (ours)	93.12 ± 0.25	79.08 ± 0.43	75.06 ± 0.76	**78.68 ± 0.19**
	$ASAP_{lp}$ (ours)	93.24 ± 0.15	79.17 ± 0.23	74.94 ± 0.56	78.65 ± 0.33
	$ASAP_{sp}$ (ours)	93.00 ± 0.12	79.12 ± 0.49	74.69 ± 1.36	78.42 ± 0.20
WRN-50-2	Baseline	94.33 ± 0.13	77.92 ± 0.64	69.36 ± 1.20	77.08 ± 0.38
	BlurPooling Zhang (2019)	94.42 ± 0.12	76.21 ± 0.30	68.66 ± 0.49	77.59 ± 0.47
	ABlurPooling Zou et al. (2020)	93.66 ± 0.18	78.26 ± 1.50	71.76 ± 1.91	76.74 ± 1.10
	WaveletPooling Li et al. (2020)	94.44 ± 0.13	78.12 ± 1.11	69.26 ± 1.02	**79.95 ± 0.42**
	FLC Pooling (ours)	94.33 ± 0.20	75.41 ± 0.41	66.30 ± 0.87	79.33 ± 0.43
	$ASAP_{stbl}$(ours)	**94.51 ± 0.17**	77.22 ± 0.89	71.24 ± 1.86	79.90 ± 0.37
	$ASAP_{lp}$ (ours)	94.20 ± 0.18	**78.63 ± 0.57**	**72.36 ± 0.67**	79.72 ± 0.41
	$ASAP_{sp}$(ours)	94.16 ± 0.21	77.30 ± 0.24	70.74 ± 1.14	79.61 ± 0.82
AlexNet	Baseline	**89.45 ± 0.23**	69.28 ± 0.22	69.97 ± 0.28	73.74 ± 0.13
	FLC Pooling (ours)	87.80 ± 0.10	70.40 ± 0.36	71.49 ± 0.36	**74.33 ± 0.50**
	$ASAP_{stbl}$ (ours)	87.59 ± 0.23	70.50 ± 0.26	71.52 ± 0.22	73.84 ± 0.27
	$ASAP_{lp}$ (ours)	87.90 ± 0.09	**70.87 ± 0.26**	**71.93 ± 0.20**	74.02 ± 0.42
	$ASAP_{sp}$ (ours)	87.68 ± 0.16	70.71 ± 0.30	71.82 ± 0.26	73.94 ± 0.21

Attacks are done with an epsilon of $\epsilon = 1/255$ and corruption performance is reported as mean over all severities. For CIFAR-10 our ASAP and FLC Pooling outperform the baseline and show overall a high robustness against adversarial attacks and common corruptions
Bold values signifies the best entry/entries per row

for Wavelet Pooling by Li et al. (2020). Zou et al. (2020) only provide weights for ResNet-101. For our FLC Pooling and ASAP, we follow the training procedures suggested by the original authors of each network.

Table 3 shows the performance of each network on clean, perturbed, and corrupted versions of the ImageNet-1k dataset. We observe that all models benefit from the use of our ASAP method for the robustness against common corruptions. Consistently, $ASAP_{sp}$ outperforms all other methods. Interestingly, all of our ASAP variants outperform the baseline and all other state-of-the-art methods, like BlurPooling Zhang (2019), ABlurPooling Zou et al. (2020) or WaveletPooling Li et al. (2020) on the corrupted data. For all ResNet-like networks, the clean performance of the network is improved with our $ASAP_{sp}$ and $ASAP_{lp}$. However, for MobileNet-v2 Sandler et al. (2018), our downsampling methods cannot beat the baseline in terms of clean accuracy. We hypothesize that this behavior is due to the highly optimized training schedule used to train a MobileNet-v2. Thus, including a new kind of downsampling might require additional

finetuning of their training hyperparameters. Analysing the adversarial robustness of our FLC Pooling and ASAP networks, we observe a trend towards higher robustness against FGSM and APGD for all downsampling methods, including the removal of high-frequency information in the frequency domain. Hence, networks with ASAP and FLC Pooling can maintain high accuracy under FGSM attack. The stronger APGD attack is able to fool the baseline in almost all cases completely. Other methods against aliasing are similarly weak in preventing the model from being fooled. In contrast, networks using our ASAP and FLC Pooling cannot be fooled on all samples by the attack. In summary, the improved robustness of models using FLC Pooling and ASAP indicates more stable feature learning.

For low-resolution data we train ResNet-18 He et al. (2016) and Wide-ResNet-50-2 Zagoruyko and Komodakis (2016) (WRN-50-2) models on CIFAR-10 Krizhevsky and Hinton (2009) with five different random seeds per network architecture. We compare the standard baseline network, BlurPooling Zhang (2019), adaptive BlurPooling Zou et al. (2020) (ABlurPooling) and WaveletPooling Li et al. (2020). All networks are trained with the same set of hyperparameters: 150 epochs, a batch size of 256, a cosine learning rate schedule with a maximum learning rate of 0.2 and a minimum of 0.0, a momentum of 0.9, and a weight decay of 0.002. We utilize label smoothing with a factor of 0.1, and Stochastic Gradient Descent (SGD) for optimization.

Table 4 shows the results of the low-resolution (32×32 pixel) dataset, CIFAR-10. FLC Pooling and ASAP consistently outperform the baseline in terms of robustness while maintaining similar clean performance indicating that stable representations have been learned. On ResNet-18, ABlurPooling Zou et al. (2020) shows the highest robustness against adversarial attacks, yet with a slight decrease in clean performance and robustness against common corruptions. Further, we compare to Rodríguez-Muñoz and Torralba (2022) which use anti-aliasing mechanisms for downsampling and the activation function. Figure 10 presents this comparison under different ϵ budgets to examine the stability of the learned features. Our improved downsampling techniques can consistently provide more stable features leading to higher robustness compared to Rodríguez-Muñoz and Torralba (2022) for all ϵ budgets without adversarial training.

4.3 Catastrophic Overfitting

Catastrophic overfitting refers to the issue that models adversarially trained with FGSM Goodfellow et al. (2015) tend to overfit to the FGSM attack Kim et al. (2021). This usually only happens after several training epochs and leads to very low robustness towards other attacks such as PGD Kurakin et al. (2017). In Grabinski et al. (2022b), we observed that catastrophic overfitting in FGSM adversarial training often coincides with a high amount of aliasing in the model's downsampling layer. As a consequence, we assume that FLC Pooling and ASAP should both reduce the risk of catastrophic overfitting in FGSM adversarial training, and thus facilitate to use this cheap training alternative for practically good results.

Qualitative analysis Fig. 11 confirms our hypothesis for the example of FGSM training on CIFAR-10 using PreAct-Resnet-18 (PRN-18), standard FGSM training, marked in red, exhibits a sudden increase in aliasing at the same time where the PGD test loss increase, i.e. when catastrophic overfitting takes place. In contrast, FLC Pooling, $\text{ASAP}_{\text{stbl}}$ and ASAP_{sp} do not suffer from catastrophic overfitting and offer the most favorable trade-off between performance and efficiency, exhibit a low PGD train and test loss while only using simple FGSM AT.

Adversarial training on high-resolution data We train FLC Pooling and ASAP_{sp} with FGSM Wong et al. (2020) and PGD Kurakin et al. (2017) adversarial training on ImageNet-1k. We trained on a ResNet-50 and compare to models reported on RobustBench Croce et al. (2021) with the same architecture. Table 5 shows that our FGSM trained model outperforms the baseline model trained with FGSM Wong et al. (2020) in robust and clean accuracy, and even the model trained by Engstrom et al. (2019) which takes significantly longer than our method as shown in Table 9. The model by Salman et al. (2020) achieves higher robustness on AutoAttack, while being slightly worse on clean samples than our FGSM and PGD trained ASAP_{sp} model. While all models trained with PGD utilize a stronger adversary and consequently improve robustness against AutoAttack, the hyperparameter configuration appears to play a crucial role. For instance, Engstrom et al. (2019) falls behind both our ASAP_{sp} and Salman et al. (2020). However, identifying the decisive parameter is difficult, as the exact configurations used to train the models on the RobustBench leaderboard are not fully detailed in their respective papers.

Adversarial training on low-resolution data For CIFAR-10 we trained each model architecture with FGSM adversarial training using three different random seeds. All hyperparameters were kept consistent across architectures and downsampling methods. Each network underwent 300 training epochs with a batch size of 512 and a cycling learning rate schedule ranging from a maximum of 0.2 to a minimum of 0.0. The momentum was set to 0.9, and weight decay was set to 0.0005. We employed CrossEntropyLoss as the loss function and utilized Stochastic Gradient Descent (SGD) as the optimizer. The budget for the adversaries during training is $\epsilon = 8/255$.

The results in Table 6 indicate that ASAP, similar to FLC Pooling, learns more stable features during adversarial training leading to higher robustness against adversarial attacks

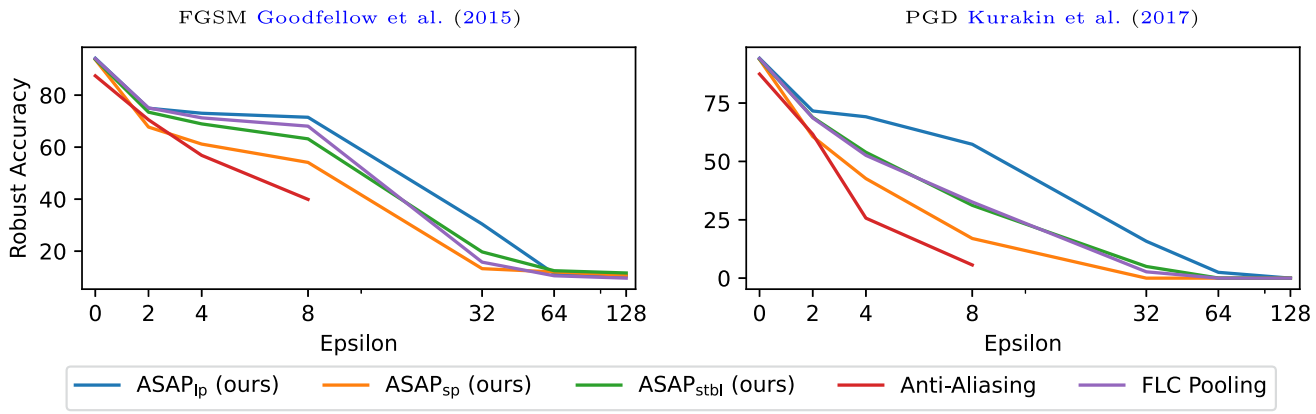

Fig. 10 Evaluating different attack ϵ budgets of ResNet-50 on CIFAR-10 without adversarial training. We compare models using FLC Pooling and ASAP to the approach by Rodríguez-Muñoz and Torralba (2022) (Anti-Aliasing). Our downsampling variants consistently exhibit higher robust accuracy on adversarial attacks than Rodríguez-Muñoz and Torralba (2022)

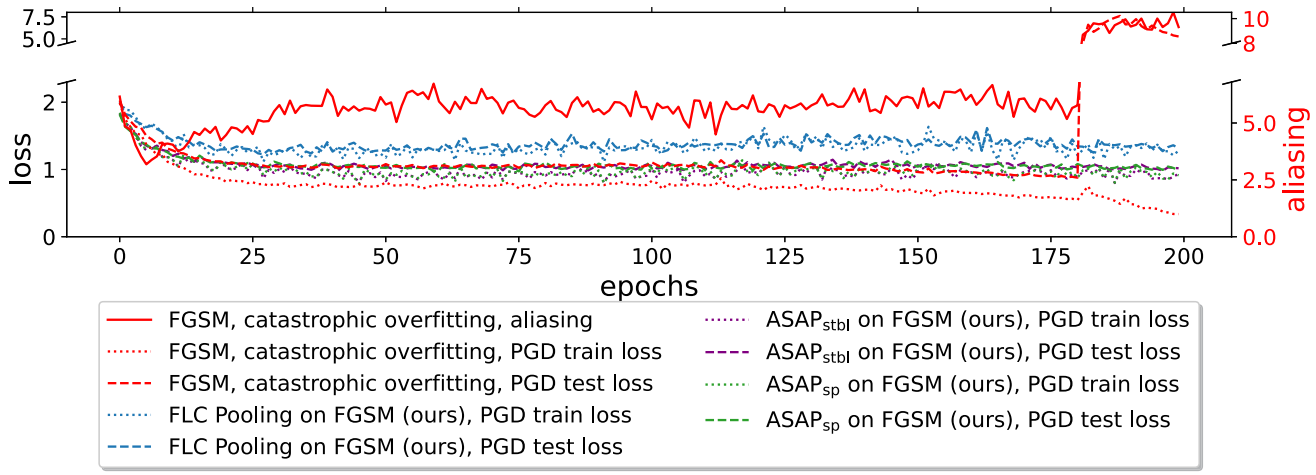

Fig. 11 Example of FGSM adversarial training facing catastrophic overfitting and its relationship to aliasing. FGSM training is prone to catastrophic overfitting (red lines) and experiences a huge increase in aliasing (red solid line) as soon as catastrophic overfitting happens, i.e. the test error on stronger adversaries like PGD increases (red dashed line) while the training error (red dotted line) stays low. Our methods, FLC Pooling and ASAP, are able to train with the fast FGSM adversarial training while preventing catastrophic overfitting (dashed and dotted lines)

Table 5 Comparison of ResNet-50 models that are trained non-adversarially or adversarially with FGSM or PGD on ImageNet-1k.

Method		Clean	AA L_{inf} $\epsilon = \frac{4}{255}$
Non-adversarial training Croce et al. (2021)		76.52	0.00
FGSM	FLC Pooling (ours)	63.52	27.29
	ASAP$_{sp}$ (ours)	64.51	30.93
	Wong et al., 2020 Wong et al. (2020)	55.62	26.24
PGD	Robustness lib, 2019 Engstrom et al. (2019)	62.56	29.22
	Salman et al., 2020 Salman et al. (2020)	64.02	34.96
	ASAP$_{sp}$ (ours)	64.54	31.02

Accuracies (in percent) shown on clean and perturbed (AutoAttack Croce and Hein (2020)) validation images. We compare against models reported on RobustBench Croce et al. (2021)

Table 6 Accuracy (in percent) for several common models trained with FGSM adversarial training Goodfellow et al. (2015) on CIFAR-10.

	Method	Acc@1	FGSM	PGD
ResNet-18	Baseline	78.85 ± 1.74	34.49 ± 2.68	21.14 ± 14.88
	FLC Pooling	79.77 ± 0.49	34.37 ± 1.07	32.23 ± 0.68
	$ASAP_{stbl}$	79.59 ± 0.64	35.13 ± 0.75	32.65 ± 0.59
	$ASAP_{lp}$	**80.63 ± 0.14**	**37.04 ± 0.65**	33.43 ± 0.13
	$ASAP_{sp}$	79.19 ± 0.32	35.44 ± 0.75	**33.68 ± 0.31**
WRN-50-2	Baseline	79.42 ± 0.34	39.18 ± 7.15	23.36 ± 16.35
	FLC Pooling	82.94 ± 0.89	39.23 ± 0.32	29.69 ± 11.81
	$ASAP_{stbl}$	83.63 ± 0.14	**39.67 ± 0.28**	37.62 ± 0.24
	$ASAP_{lp}$	**84.60 ± 0.13**	39.56 ± 0.88	36.99 ± 0.19
	$ASAP_{sp}$	83.26 ± 0.24	39.16 ± 0.37	**38.86 ± 0.17**
PRN-18	Baseline	77.92 ± 0.19	31.74 ± 0.56	32.52 ± 0.35
	FLC Pooling	79.99 ± 0.09	36.39 ± 0.74	33.15 ± 0.19
	$ASAP_{stbl}$	79.91 ± 0.17	36.25 ± 0.20	33.20 ± 0.14
	$ASAP_{lp}$	**81.29 ± 0.20**	**38.02 ± 0.85**	**33.48 ± 0.05**
	$ASAP_{sp}$	79.77 ± 0.20	35.88 ± 0.49	33.35 ± 0.19

We report adversarial robustness against FGSM Goodfellow et al. (2015) and PGD Kurakin et al. (2017) with 50 attack iterations and 10 random restarts. Both attack have an ϵ budget of 8/255. We clearly see, that our ASAP which neither suffers from aliasing nor from sinc artifacts, is also more robust in combination with adversarial training
Bold values signifies the best entry/entries per row

than the baseline. Particularly when confronted with more complex adversaries like PGD Kurakin et al. (2017) with 50 attack iterations and 10 random restarts, our $ASAP_{lp}$ consistently outperforms the baseline and FLC Pooling. The high variance in performance on PGD samples for the ResNet-18 and Wide-ResNet-50-2 baseline indicate that some of the trained models lose all their robustness against PGD during FGSM adversarial training due to catastrophic overfitting. In contrast, our FLC Pooling and ASAP do not experience this issue due to the stable feature learning and maintain high robustness against strong and simple adversaries in all models. For PreAct-ResNet-18, which is commonly used for adversarial training Gowal et al. (2021b); Rade and Moosavi-Dezfooli (2021); Rebuffi et al. (2021), none of the networks experiences catastrophic overfitting. This is one aspect, of why this network architecture might be used often for adversarial training. Still, our FLC Pooling and ASAP outperform the baseline on clean and perturbed samples. Furthermore, all ASAP variants exhibit improved robustness against FGSM attacks and higher clean accuracy compared to the baseline (up to 5% improvement against FGSM attacks and up to 4% improvement on clean data). $ASAP_{lp}$ improves the clean as well as the robust performance for the smaller models like ResNet-18 and PreAct-ResNet-18. The larger Wide-ResNet-50-2 only benefits from the large padding for clean accuracy.

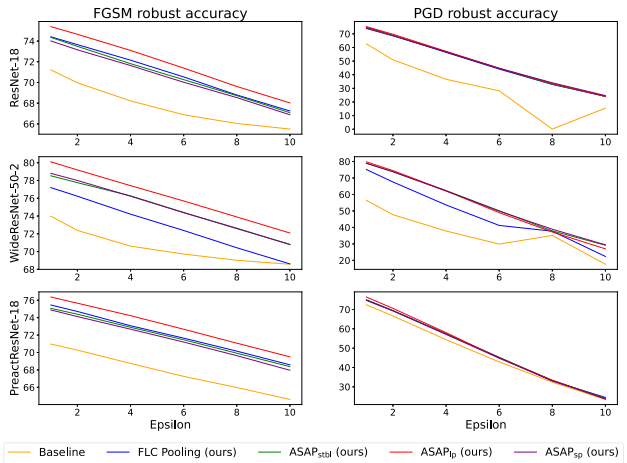

Fig. 12 Evaluation (accuracy in percent) of networks adversarially trained with FGSM Goodfellow et al. (2015) on CIFAR-10 evaluated on FGSM Goodfellow et al. (2015) (left) and PGD Kurakin et al. (2017) (right) adversaries with different ϵ budgets. Our three ASAP variants consistently exhibit higher robust accuracy on all architectures, adversarial attacks and across ϵ values than the baseline

4.4 Ablations

In the following, we conduct a series of ablations. First, we test the stability of our learned feature representations by examining the behavior of FLC Pooling and ASAP under attack with different ϵ values. Further, we analyze the specific window functions to be used and the combination of the

Table 7 Ablation on stabilization. We report accuracy (in percent) on clean samples, perturbed samples with FGSM Goodfellow et al. (2015) and PGD Kurakin et al. (2017) as well as corrupted samples Hendrycks and Dietterich (2019) for different settings of ASAP on ResNet-18 He et al. (2016) trained without adversarial training on CIFAR-10.

Architecture	Acc@1	FGSM	PGD	Corruptions
Baseline	93.03 ± 0.13	78.62 ± 0.28	72.49 ± 0.67	76.93 ± 0.45
$\text{ASAP}_{\text{stbl}}$	93.12 ± 0.25	79.08 ± 0.43	75.06 ± 0.76	$\mathbf{78.68 \pm 0.19}$
ASAP_{lp}	$\mathbf{93.24 \pm 0.15}$	$\mathbf{79.17 \pm 0.23}$	74.94 ± 0.56	78.65 ± 0.33
ASAP_{sp}	93.00 ± 0.12	79.12 ± 0.49	74.69 ± 1.36	78.42 ± 0.20
$\text{ASAP}_{\text{lp+stbl}}$	93.22 ± 0.07	78.37 ± 0.77	$\mathbf{75.65 \pm 1.43}$	78.38 ± 0.38
$\text{ASAP}_{\text{sp+stbl}}$	92.88 ± 0.11	77.29 ± 1.02	74.82 ± 1.90	78.77 ± 0.17

Attacks are done with $\epsilon = 1/255$ and corruption performance is reported as mean over all severities. Bold values signifies the best entry/entries per row

Table 8 Ablation on using different window functions in the frequency domain to reduce spectral artifacts.

Window	Hyperparameter	Acc@1	FGSM	PGD	Corruptions
None		93.12 ± 0.19	78.92 ± 0.26	74.17 ± 0.60	78.59 ± 0.29
Hamming		$\underline{93.12 \pm 0.25}$	$\mathbf{79.08 \pm 0.43}$	75.06 ± 0.76	$\underline{78.68 \pm 0.19}$
Gaussian	$\sigma = (k-1)/6$	92.34 ± 0.15	77.85 ± 0.30	69.47 ± 0.40	78.53 ± 0.14
Hanning		93.13 ± 0.19	$\underline{79.23 \pm 0.27}$	$\underline{75.32 \pm 0.92}$	78.56 ± 0.26
Kaiser	$\beta = 7$	$\mathbf{93.21 \pm 0.17}$	78.87 ± 0.33	74.57 ± 1.25	78.66 ± 0.21
Blackman		93.00 ± 0.17	78.82 ± 0.43	74.78 ± 1.16	$\mathbf{78.77 \pm 0.21}$
Fejér		92.93 ± 0.12	78.94 ± 0.36	$\mathbf{77.68 \pm 0.40}$	78.39 ± 0.28

Mean clean and robust accuracy (in percent) and their standard deviation on CIFAR-10 for five different window functions over five different random seeds. The best result is marked in **bold** and the second best via underlining. Using no window function or a simple Gaussian window which even needs an additional hyperparameter performs quite poorly. All window functions which are a specialization of a Kaiser window perform reasonably well. Our Hamming window performs consistently well considering the top two performing methods

different ASAP variants. Finally, we ablate on the different design choices for ASAP.

Ablating the attack strength We assessed the behavior of FLC Pooling and ASAP under different budget settings of ϵ. Figure 12 displays the mean robust accuracy trend across each architecture, varying the budget of ϵ. It is evident that ASAP consistently outperforms the baseline. Moreover, ASAP_{lp} demonstrates improved performance over all epsilon strengths under FGSM attack. In comparison, under PGD attack all our ASAP variants perform equally well, demonstrating improved stability in the learned feature representations.

Ablation on the window function For our work, we mainly focused on the Hamming window, but there are several widely known window functions that could be used to reduce sinc artifacts. Thus, we ablated five additional choices for the window function in our ASAP method. Here, we additionally evaluate a standard Gaussian kernel, a Blackman window, a Hanning window, and a Kaiser window with $\beta = 7$ and Fejér kernel. Similar to the Kaiser kernel, we needed to choose an additional hyperparameter σ for the Gaussian kernel. We set sigma σ in relation to the kernel size k such that $\sigma = (k-1)/6$ as the length of 99 percentile of the Gaussian pdf is 6σ. Table 8 presents the performance of five different random seeds trained on CIFAR-10 with the mentioned different window functions. One can note that the models using a Gaussian window do not support the robustness of the network well, while all models based on a Kaiser window[1] perform similarly well on clean, perturbed and corrupted data. When considering the top two performing methods, the Hamming window used for our ASAP performs consistently well and is thus a good choice.

Ablation on ASAP variants Our method to stabilize the FFT by transposing ($\text{ASAP}_{\text{stbl}}$) can also be combined with padding. Table 7 ablates on this combination (avg. over five random seeds). The combination of both approaches, stabilization through transposing the signal and padding, yields no further systematic benefit, indicating that both approaches address the same issue.

5 Discussion

Efficiency Our FLC Pooling and ASAP stabilize feature learning leading to higher native robustness against common corruptions and adversarial attacks and reduce the risk of catastrophic overfitting during FGSM adversarial training.

[1] Hamming, Hanning and Blackman window are all specializations of a Kaiser window with fixed $\beta = 6.0$, $\beta = 5.0$ and $\beta = 8.6$ respectively.

Table 9 Runtime of AT in seconds per epoch over 200 epochs and a batch size of 512 trained with a PreAct-ResNet-18 for training on the original CIFAR-10 dataset without additional data.

Method	Avg #seconds per epoch	Acc (%) clean	APGD
Baseline	6.6 ± 0.01	93.06	0.00
FGSM & early stopping Wong et al. (2020)	12.6 ± 0.01	82.88	11.82
FGSM & FLC Pooling (ours)	14.7 ± 0.01	80.94	31.16
FGSM & $ASAP_{stbl}$ (ours)	15.6 ± 0.08	80.47	31.75
FGSM & $ASAP_{sp}$(ours)	17.1 ± 0.16	80.47	31.40
FGSM & $ASAP_{lp}$ (ours)	36.4 ± 0.01	81.12	31.39
PGD (Kurakin et al., 2017)	115.4 ± 0.2	83.11	41.12
Robustness lib (Engstrom et al., 2019)	117 ± 19.0	76.37	33.09
AWP Wu et al. (2020)	179.4 ± 0.4	82.61	**53.53**
MART Wang et al. (2020b)	180.4 ± 0.8	55.49	10.03
TRADES Zhang et al. (2019)	219.4 ± 0.5	81.49	49.65

Experiments are executed on one Nvidia Tesla V100. Evaluation for clean and robust accuracy, higher is better, on APGD Croce and Hein (2021) with our trained models. The models reported by the original authors may have varying results due to different hyperparameter selection. The top row reports the baseline without AT

Yet, we need to perform additional operations to transform from the spatial into the frequency domain and vice versa, resulting in increased computational effort. Since FFT allows for heavy parallelization, the exact additional time requirements for its computation strictly depend on the available GPU type and memory as well as the specific implementation. Especially when we are using large additional padding, the cost for the transformation increases (e.g. by a factor of 5.6 in execution time on CIFAR-10 for a batch size of 512 on an Nvidia Tesla V100, smaller batch size might allow better FFT parallelization on the GPU, but require hyperparameter tuning). We could show that small padding can lead to an equal or better increase in robustness while increasing the computational effort only by a factor of 1.19 compared to using no padding (Table 2). In comparison, additional data augmentation to improve robustness against common corruptions increases the number of samples that need to be learned, and adversarial training requires several forward and backward passes for each batch. Our analysis shows that adding additional FFT operations increases the training time by a factor of 1.3 on a V100, while using more sophisticated adversarial training increases the training time by at least a factor of 9 or even 17 (Table 9). Our proposed ASAP increases the training time dependent on the stabilization process used. Stabilization by the transpose operation and small padding increases the adversarial training time per epoch by a factor of 1.2. While large padding increases the training time by a factor of 2.8 (Table 9), it is still faster than other sophisticated adversarial training methods. Further, we want to point out that incorporating additional data like *ddpm* Ho et al. (2020) which is a widely used source for adversarial training Gowal et al. (2021b); Rade and Moosavi-Dezfooli (2021); Rebuffi et al. (2021) increases the training time by a factor of 20 on a V100. In summary, adding FFT operations in the network achieves stable feature learning leading to high robustness while keeping the training time comparably low.

Variant selection All four variants (FLC Pooling, $ASAP_{stbl}$, $ASAP_{sp}$ and $ASAP_{lp}$) are entirely alias-free (Table 1), so the choice depends on how well residual sinc-interpolation artifacts are suppressed and on the affordable compute. Plain FLC Pooling is cheapest but retains visible ringing (Fig. 1). $ASAP_{lp}$ is theoretically cleanest and strongest under adversarial training, yet also the most expensive. We therefore recommend $ASAP_{sp}$ or $ASAP_{stbl}$, which remove the bias from off-center cutouts at relatively low cost. Combining stabilization and padding yields no further gain (Table 7), indicating both address the same issue.

Limitations While FLC Pooling and ASAP consistently improve robustness, they introduce an efficiency trade-off. Specifically, $ASAP_{lp}$ provides the greatest robustness when combined with adversarial training but suffers from the longest execution time due to its large padding (Table 9). Additionally, our focus was restricted to classification. While aggressive high-frequency filtering is effective here, it may fail in pixel-level applications, such as image reconstruction, where Agnihotri et al. (2024); Chen et al. (2024) show that preserving high frequencies for reconstructing fine details, is critical for performance.

6 Conclusion

We introduce the novel application of two downsampling approaches in the Fourier domain to CNNs. Both completely eliminate aliasing through aliasing-free pooling (FLC Pooling). Aliasing and Sinc Artifact-free Pooling (ASAP)

additionally addresses sinc-interpolation artifacts. Through extensive qualitative analysis, we motivate the benefit of artifact-free pooling. Our quantitative analysis shows that FLC Pooling and ASAP learn beneficial, more stable features, leading, even without dedicated training, to improved robustness against common corruptions and adversarial attacks while maintaining high accuracy on clean data. Further, FLC Pooling and ASAP stabilize FGSM adversarial training and thus sustain high levels of robustness and accuracy by reducing the risk of catastrophic overfitting.

Acknowledgements Steffen Jung and Margret Keuper acknowledge funding by the DFG Research Unit 5336 - Learning to Sense and by the BMFTR project TrackOpt (01|S24074A-D).

Funding Open Access funding enabled and organized by Projekt DEAL. Open Access funding enabled and organized by Projekt DEAL.

Data Availability The datasets worked with during the current study are publicly available at ImageNet-1k and CIFAR-10, respectively.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

Agnihotri, S., Grabinski, J., Keuper, J., Keuper, M. (2024). Beware of aliases–signal preservation is crucial for robust image restoration. arXiv:2406.07435.

Andriushchenko, M., Croce, F., Flammarion, N., & Hein, M. (2020). Square attack: A query-efficient black-box adversarial attack via random search. In *European conference on computer vision* (pp. 484-501). Cham: Springer International Publishing.

Carmon, Y., Raghunathan, A., Schmidt, L., Duchi, J. C., & Liang, P. S. (2019). Unlabeled data improves adversarial robustness. *Advances in neural information processing systems*, 32.

Chen, L., Gu, L., Fu, Y. (2024). When semantic segmentation meets frequency aliasing. In *International Conference on Learning Representations* (Vol. 2024, pp. 43194–43216).

Croce, F., Andriushchenko, M., Sehwag, V., Debenedetti, E., Flammarion, N., Chiang, M., & Hein, M. (2021). Robustbench: a standardized adversarial robustness benchmark. *NeurIPS*.

Croce, F., & Hein, M. (2020). Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks.In *International conference on machine learning* (pp. 2206–2216). PMLR.

Croce, F., & Hein, M. (2021). Mind the box: l_1-apgd for sparse adversarial attacks on image classifiers. In *International Conference on Machine Learning* (pp. 2201–2211). PMLR.

Cubuk, E. D., Zoph, B., Mane, D., Vasudevan, V., & Le, Q. V. (2018). Autoaugment: Learning augmentation policies from data. arXiv:1805.09501

Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition* (pp. 248-255). Ieee.

Durall, R., Keuper, M., Keuper, J. (2020). Watch your up-convolution: Cnn based generative deep neural networks are failing to reproduce spectral distributions. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 7887–7896). IEEE.

Engstrom, L., Ilyas, A., Salman, H., Santurkar, S., Tsipras, D. (2019). *Robustness (python library)*.https://github.com/MadryLab/robustness

Geirhos, R., Rubisch, P., Michaelis, C., Bethge, M., Wichmann, F. A., & Brendel, W. (2018). Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. arXiv:1811.12231

Gonzales, R. C., & Wintz, P. (1987). *Digital image processing*. Inc: Addison-Wesley Longman Publishing Co.

Gonzalez, R. C., & Woods, R. E. (2006). *Digital image processing* (3rd ed.). USA: Prentice-Hall Inc.

Goodfellow, I. J., Shlens, J., & Szegedy, C. (2015). Explaining and harnessing adversarial examples. arXiv:1412.6572

Gowal, S., Qin, C., Uesato, J., Mann, T., & Kohli, P. (2021). Uncovering the limits of adversarial training against norm-bounded adversarial examples. arXiv:2010.03593

Gowal, S., Rebuffi, S. A., Wiles, O., Stimberg, F., Calian, D. A., & Mann, T. A. (2021). Improving robustness using generated data. *Advances in Neural Information Processing systems, 34*, 4218–4233.

Grabinski, J., Jung, S., Keuper, J., Keuper, M. (2022). Frequencylowcut pooling–plug & play against catastrophic overfitting. In *European Conference on Computer Vision* (pp. 36–57). Cham: Springer Nature Switzerland.

Grabinski, J., Keuper, J., Keuper, M. (2022a). Aliasing and adversarial robust generalization of cnns. *Machine Learning*, 1–27.

Grabinski, J., Keuper, J., Keuper, M. (2022b). Aliasing coincides with cnns vulnerability towards adversarial attacks. In *The AAAI-22 Workshop on Adversarial Machine Learning and Beyond* (pp. 1–5).

Hamming, R. W., & Stearns, S. D. (1979). Digital filters. *IEEE Transactions on Systems, Man and Cybernetics, 9*(1), 67–67.

He, K., Zhang, X., Ren, S., Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770-778).

Hendrycks, D., & Dietterich, T. (2019). Benchmarking neural network robustness to common corruptions and perturbations. *ICLR*.

Hendrycks, D., Mu, N., Cubuk, E. D., Zoph, B., Gilmer, J., & Lakshminarayanan, B. (2020). AugMix: A simple data processing method to improve robustness and uncertainty. *ICLR*.

Ho, J., Jain, A., & Abbeel, P. (2020). Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems, 33*, 6840–6851.

Hossain, M. T., Teng, S. W., Lu, G., Rahman, M. A., & Sohel, F. (2023). Anti-aliasing deep image classifiers using novel depth adaptive blurring and activation function. *Neurocomputing, 536*, 164–174.

Jähne, B. (2005). *Digital image processing*. Springer Science & Business Media.

Jung, S., & Keuper, M. (2021). Spectral distribution aware image generation. In *Proceedings of the AAAI conference on artificial intelligence* (Vol. 35, No. 2, pp. 1734-1742).

Karras, T., Aittala, M., Laine, S., Härkönen, E., Hellsten, J., Lehtinen, J., & Aila, T. (2021). Alias-free generative adversarial networks. *Advances in Neural Information Processing systems, 34*, 852–863.

Kim, H., Lee, W., Lee, J. (2021). Understanding catastrophic overfitting in single-step adversarial training. In *Proceedings of the AAAI conference on artificial intelligence* (Vol. 35, No. 9, pp. 8119–8127).

Kireev, K., Andriushchenko, M., Flammarion, N. (2022). On the effectiveness of adversarial training against common corruptions. In *Uncertainty in Artificial Intelligence* (pp. 1012–1021). PMLR.

Krizhevsky, A., & Hinton, G. (2009). *Learning multiple layers of features from tiny images*. Toronto: ON, Canada.

Kurakin, A., Goodfellow, I. J., & Bengio, S. (2017). Adversarial machine learning at scale. *ICLR*.

Li, Q., Shen, L., Guo, S., & Lai, Z. (2020). Wavelet integrated cnns for noise-robust image classification. *CVPR*.

Li, Q., Shen, L., Guo, S., & Lai, Z. (2021). Wavecnet: Wavelet integrated cnns to suppress aliasing effect for noise-robust image classification. *TIP, 30*, 7074–7089.

Lukasik, J., Gavrikov, P., Keuper, J., & Keuper, M. (2023). Improving native cnn robustness with filter frequency regularization. *TMLR, 2023*, 1–36.

Prabhu, K. M. (2014). *Window functions and their applications in signal processing*. Taylor & Francis.

Rade, R., & Moosavi-Dezfooli, S.M. (2021). Helper-based adversarial training: Reducing excessive margin to achieve a better accuracy vs. robustness trade-off. *International Conference on Learning Representations*.

Rebuffi, S. A., Gowal, S., Calian, D. A., Stimberg, F., Wiles, O., & Mann, T. (2021). Fixing data augmentation to improve adversarial robustness. arXiv:2103.01946

Rodríguez-Muñoz, A., & Torralba, A. (2022). Aliasing is a driver of adversarial attacks. arXiv:2106.05963

Rony, J., Hafemann, L. G., Oliveira, L. S., Ayed, I. B., Sabourin, R., & Granger, E. (2019). Decoupling direction and norm for efficient gradient-based l2 adversarial attacks and defenses. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 4317–4325). IEEE.

Salman, H., Ilyas, A., Engstrom, L., Kapoor, A., & Madry, A. (2020). Do adversarially robust imagenet models transfer better? *Advances in Neural Information Processing systems, 33*, 3533–3545.

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L. C. (2018). Mobilenetv 2: Inverted residuals and linear bottlenecks. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 4510-4520). Ieee.

Sehwag, V., Mahloujifar, S., Handina, T., Dai, S., Xiang, C., Chiang, M., & Mittal, P. (2021). Improving adversarial robustness using proxy distributions. arXiv:2104.09425

Semmlow, J. L., & Griffel, B. (2021). *Biosignal and medical image processing*. CRC Press.

Shannon, C. (1949). Communication in the presence of noise. *IRE, 37*(1), 10–21.

Tomen, N., & van Gemert, J.C. (2021). Spectral leakage and rethinking the kernel size in cnns. In*2021 IEEE/CVF International Conference on Computer Vision (ICCV)* (pp. 5118-5127). IEEE.

Vasconcelos, C., Larochelle, H., Dumoulin, V., Romijnders, R., Le Roux, N., & Goroshin, R. (2021). Impact of aliasing on generalization in deep convolutional networks. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)* (pp. 10529–10538). IEEE.

Wang, Y., Zou, D., Yi, J., Bailey, J., Ma, X., Gu, Q. (2020). Improving adversarial robustness requires revisiting misclassified examples. In *International conference on learning representations*.

Wong, E., Rice, L., & Kolter, J. Z. (2020). Fast is better than free: Revisiting adversarial training. In *International conference on learning representations*.

Wu, D., Xia, S. T., & Wang, Y. (2020). Adversarial weight perturbation helps robust generalization. *Advances in Neural Information Processing Systems, 33*, 2958–2969.

Zagoruyko, S., & Komodakis, N. (2016). Wide residual networks. arXiv:1605.07146

Zhang, H., Yu, Y., Jiao, J., Xing, E. P., Ghaoui, L. E., & Jordan, M. I. (2019). Theoretically principled trade-off between robustness and accuracy. In *International Conference on Machine Learning* (pp. 7472–7482). PMLR.

Zhang, R. (2019). Making convolutional networks shift-invariant again. In *International Conference on Machine Learning* (pp. 7324–7334). PMLR.

Zou, X., Xiao, F., Yu, Z., & Lee, Y. J. (2020). Delving deeper into anti-aliasing in convnets. *BMVC*.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.