FREQUENCY POOLING: SHIFT-EQUIVALENT AND ANTI-ALIASING DOWN SAMPLING

Anonymous authors
Paper under double-blind review

ABSTRACT

Convolutional layer utilizes the shift-equivalent prior of images which makes it a great success for image processing. However, commonly used down sampling methods in convolutional neural networks (CNNs), such as max-pooling, average-pooling, and strided-convolution, are not shift-equivalent. This destroys the shift-equivalent property of CNNs and degrades their performance. In this paper, we propose a novel pooling method which is *strict shift equivalent and anti-aliasing* in theory. This is achieved by (inverse) Discrete Fourier Transform and we call our method frequency pooling. Experiments on image classifications show that frequency pooling improves accuracy and robustness w.r.t shifts of CNNs.

1 INTRODUCTION

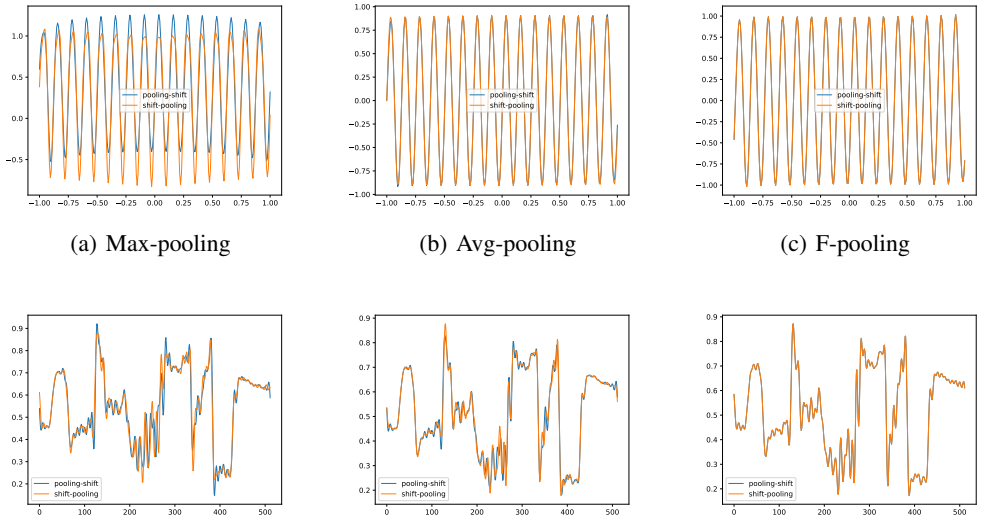
Convolutional neural networks (CNNs) have achieved great success on image processing Goodfellow et al. (2016), natural language processing Yin et al. (2017), game playing Mnih et al. (2013) and so on. One of the reasons is that convolutions utilize the shift-equivalent prior of signals. Modern CNNs include not only convolutional layers but also down sampling or pooling layers. As an important part of CNNs, pooling layers are used to reduce spatial resolution of feature maps, aggregate spatial information, and reduce computational and memory cost.

Based on classical Nyquist sampling theorem Nyquist (1928), the sampling rate must be at least as twice as the highest frequency of a signal. Otherwise, frequency aliasing will appear, i.e. high-frequencies of the signal alias into low-frequencies. To anti-alias, a traditional solution is that applying low-pass filter to the signal before down sampling it. Following the spirit of blurred down sampling, early CNNs use average pooling to achieve down sampling Lecun et al. (1989). Later, empirical evidence suggests max-pooling Scherer et al. (2010) and strided-convolutions Long et al. (2015) provide better performance. They are widely used in CNNs but they don't consider about anti-aliasing.

Shift-equivalent is another expected property of pooling. Because shift-nonequivalent poolings destroy the shift-equivalent of CNNs and thus the shift-equivalent prior of signals is not fully utilized. Unfortunately, they are not shift-equivalent. Worse, small shifts in the input can drastically change the output when stacking multiple max-pooling or strided convolutions Engstrom et al. (2017); Azulay & Weiss (2018); Zhang (2019).

Recently, Zhang (2019) propose anti-aliasing pooling (AA-pooling) by low-pass filtering before down sampling. They observe increased accuracy and better generalization on image classification when low-pass filtering is integrated correctly. Specifically, they decompose a pooling operator with down sampling factor s into two parts: a pooling operator with factor 1 and a blur filter with factor s. Although AA-pooling reduces the aliasing effects and makes the outputs more stable w.r.t input shifts, it is not strict shift-equivalent in theory.

In this paper, we propose a novel pooling which is *strict shift equivalent* in theory. We first transform a signal or image into frequency domain via Discrete Fourier Transform (DFT). Then we only retain its low frequencies, i.e. the frequencies which are smaller than half of resolution of the down sampled signal. Finally, we transform the low frequencies back into time domain via inverse DFT. We call our method frequency pooling (F-pooling). We summarize our contributions as followings:



(a) Max-pooling (b) Avg-pooling (c) F-pooling

(d) Max-pooling (e) Avg-pooling (f) F-pooling

Figure 1: Simple tests of shift-equivalent. Blue lines are obtained by pooling, up sampling, and shift in order. Orange lines are obtained by shift, pooling, and up sampling in order. All Pooling operations down sample original signals by a factor 4. The shift operation shifts original signals by 2 pixels. The up sampling operation is set to equation 8. Best viewed on screen.

- We formally define shift-equivalent of functions which contain down sampling operations. A suitable up sampling operation $\mathcal{U}$ must be involved in the definition. Without it, the definition is ill-posed.
- We prove that F-pooling is the optimal anti-aliasing down sampling operation given $\mathcal{U}$. We also prove that F-pooling is shift equivalent. To best of our knowledge, F-pooling is the first pooling method which has those properties.
- Experiments on CIFAR-100 and a subset of ImageNet demonstrate that F-pooling remarkably increases accuracy and robustness w.r.t shifts of commonly-used network architectures.

F-pooling has other two properties: 1) it adapts to variable input size and variable down sampling factor; 2) it is a global operation while previous pooling methods are local operations. However, they are not the main point of this paper. Instead, we focus on shift-equivalent and anti-aliasing.

2 METHOD

In this section, we first define shift-equivalent for CNNs formally. Then we describe F-pooling in detail and prove its properties. Finally we discuss our implementation and analyze its computational complexity.

2.1 WHAT IS SHIFT-EQUIVALENT

Denote $\mathbf{X}$ as an $h_0 \times w_0 \times c$ tensor where h_0, w_0, and c are the height, width, and number of channels of $\mathbf{X}$ respectively. Denote $\mathbf{Y}$ as an $h_1 \times w_1 \times c$ tensor. We suppose $h_0 > h_1$ and $w_0 > w_1$. $\mathcal{M} : \mathbf{X} \to \mathbf{Y}$ is a pooling operation and $\mathcal{U} : \mathbf{Y} \to \mathbf{X}$ is a up sampling operation. We say $\mathcal{M}$ is shift-equivalent if and only if

$$\begin{aligned} Shift_{\triangle h, \triangle w}(\mathcal{U}\mathcal{M}\mathbf{X}) =& \mathcal{U}\mathcal{M}Shift_{\triangle h, \triangle w}(\mathbf{X}) \\ \forall(\triangle h, \triangle w), \quad & \exists \mathcal{U} \end{aligned} \tag{1}$$

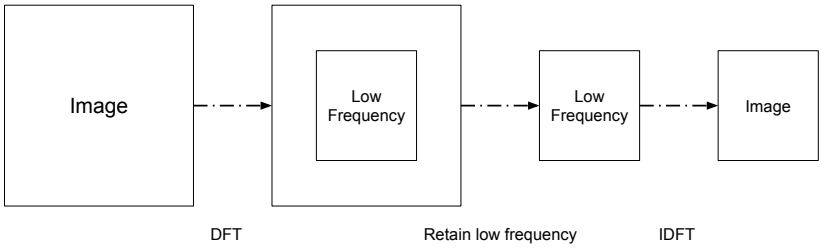


Figure 2: An illustration of the forward process of F-pooling. We assume the lowest frequencies are at center.

That is, if there is a suitable up sampling operation $\mathcal{U}$ which makes $\mathcal{U}\mathcal{M}$ and $Shift_{\triangle h,\triangle w}$ commutable, then $\mathcal{M}$ is shift-equivalent. The shift operation is required to be circular or periodic. When a shifted element hits the edge, it rolls to other side.

$$Shift_{\triangle h,\triangle w}(\mathbf{X}_{a,b}) = \mathbf{X}_{(a+\triangle h)\%h_0,(b+\triangle w)\%w_0} \tag{2}$$

where $\%$ is the modulus function. Our definition of shift-equivalent is similar with the one in Zhang (2019). The difference is that we introduce an up sampling $\mathcal{U}$ to make this definition more strict. *Without introducing $\mathcal{U}$, the definition of shift-equivalent for pooling is ill-posed.* Suppose $\mathcal{M}$ down samples signals by a factor s. Without introducing $\mathcal{U}$, one may define shift-equivalent as follows:

$$Shift_{\triangle h/s,\triangle w/s} \cdot \mathcal{M}(\mathbf{X}) = \mathcal{M} \cdot Shift_{\triangle h,\triangle w}(\mathbf{X}), \qquad \forall(\triangle h, \triangle w) \tag{3}$$

However, $Shift_{\triangle h/s,\triangle w/s}$ is not operable for discrete signals when $\triangle h\%s \neq 0$ or $\triangle h\%s \neq 0$. To make the shifted element which is not on lattice operable, one must interpolate or up sample the down sampled signals. This leads to the definition in equation 1.

As in equation 1, to make $\mathcal{M}$ shift-equivalent, one must find the corresponding up sampling operation $\mathcal{U}$. For a given $\mathcal{M}$, the first line in equation 1 may hold for some up sampling operations. But it may not hold for other up sampling operations.

2.2 F-POOLING

The basic idea of F-pooling is removing the high frequencies of signals and reconstructing the signals only using the low frequencies. In this paper, *high frequencies mean the frequencies which are beyond Nyquist frequency*, i.e. half of signal resolution. And *low frequencies mean the frequencies which are not higher than Nyquist frequency*. To remove high frequencies, we first transform signals into frequency domain via DFT. Then we remove high frequencies and retain the low frequencies. Finally, we transform the low frequencies back to time domain via inverse DFT (IDFT). Figure 2 gives an illustration of the forward process of F-pooling.

Since 2D (inverse) DFT can be decomposed into two 1D (inverse) DFT, we provide the formal representation of F-pooling in 1D case. Denote $\mathbf{x} \in \mathbb{R}^N$ as input signal and $\mathbf{y} \in \mathbb{C}^M$. Without loss of generality, we suppose M is even. $\mathbf{F}_N \in \mathbb{C}^{N\times N}$ is the so-called DFT-matrix:

$$\mathbf{F}_N = \begin{bmatrix} \omega_N^{0\cdot 0} & \omega_N^{0\cdot 1} & \dots & \omega_N^{0\cdot(N-1)} \\ \omega_N^{1\cdot 0} & \omega_N^{1\cdot 1} & \dots & \omega_N^{1\cdot(N-1)} \\ \vdots & \vdots & \ddots & \vdots \\ \omega_N^{(N-1)\cdot 0} & \omega_N^{(N-1)\cdot 1} & \dots & \omega_N^{(N-1)\cdot(N-1)} \end{bmatrix} \tag{4}$$

where $\omega_N = e^{-2\pi i/N}$. $\mathbf{F}_N^*$ is the inverse DFT-matrix. By definition, $\frac{1}{N}\mathbf{F}_N\mathbf{F}_N^* = \mathbf{I}$ where $\mathbf{I}$ is identity matrix. We denote $\mathcal{T}_\mu$ as a operation which selects the first μ rows and the last μ rows of a matrix.

$$\mathbf{T}_\mu(\mathbf{x}) = [\mathbf{x}_{\mathbf{1}:\mu}; \mathbf{x}_{N-\mu+1:N}] \tag{5}$$

That is we select the basis of frequencies ranged in $[-\mu, \mu)$, due to periodicity of DFT. When applying $\mathbf{T}_\mu$ to a signal, we obtain its lowest μ frequencies. Then the function of F-pooling for 1D

signals is represented as:

$$\mathbf{y} = \frac{1}{N}\mathbf{F}_{\mathbf{M}}^{*}\mathbf{T}_{\frac{M}{2}}\mathbf{F}_{N}\mathbf{x} \stackrel{def}{=} \mathbf{P}\mathbf{x} \tag{6}$$

where $\mathbf{P} \in \mathbb{C}^{M\times N}$ is the transform matrix of F-pooling.

Now we formalize F-pooling for CNNs. Recall that $\mathbf{X}$ is an $h_0 \times w_0 \times c$ tensor and $\mathbf{Y}$ is an $h_1 \times w_1 \times c$ tensor. We apply F-pooling to each channel of $\mathbf{X}$.

$$\mathbf{Y}_{:,:,i} = \mathbf{P}_h\mathbf{X}_{:,:,i}\mathbf{P}_w^{*}, \quad i \in [1, c] \tag{7}$$

where $\mathbf{P}_h \in \mathbb{C}^{h_1\times h_0}$ and $\mathbf{P}_w \in \mathbb{C}^{w_1\times w_0}$ are the transform matrices of F-pooling along vertical direction and horizontal direction respectively. Since F-pooling can be represented as two times matrix-matrix multiplications, its back propagation rule is easily derived.

As mentioned in section 2.1, the choice of up sampling operation $\mathcal{U}$ is important. In this paper, we set $\mathcal{U}$ to the re-sampling method which is widely used in signal processing community. Specifically, we transform a signal into frequency domain. Then we zero pad the transformed signal to match the resolution of output. Finally, we transform it back to time domain. This process is some-what symmetrical with F-pooling and can also be represented by matrix multiplications. For 1D signals, we have:

$$\mathbf{x} = \frac{1}{M}\mathbf{F}_{\mathbf{N}}^{*}\mathbf{Z}_{\frac{M}{2}}\mathbf{F}_{M}\mathbf{y} \tag{8}$$

where is $\mathbf{Z}$ is the zero padding operation.

$$\mathbf{Z}_{\mu} = [\mathbf{x}_{\mathbf{1}:\mu}; \mathbf{0}; \mathbf{x}_{M-\mu+1:M}] \tag{9}$$

$\mathcal{U}$ is formalized in a similar way for 2D signals.

2.3 Optimal anti-aliasing down sampling

In this section, we prove that F-pooling is the optimal anti-aliasing down sampling operation under a given up sampling operation $\mathcal{U}$. We focus on 1D case because it is easy to extend the conclusion into 2D case. Given $\mathcal{U}$, the optimal anti-aliasing down sampling is obtained by solving the following problem:

$$\min_{\mathcal{M}} ||\mathcal{U}\mathcal{M}\mathbf{x} - \mathbf{x}||_2^2, \quad s.t. \quad \mathcal{M} \in \mathcal{A} \tag{10}$$

where $\mathcal{A}$ is a set of all possible anti-aliasing down sampling operations. That is, we hope to find an anti-aliasing down sampling which minimizes the reconstruction error. Based on Nyquist sampling theory, $\mathcal{M}$ must remove high frequencies of signals and this holds for F-pooling. We focus on the optimality of F-pooling. We decompose $\mathbf{x}$ into low frequencies part $\mathbf{x}_l$ and high frequencies part $\mathbf{x}_h$, i.e. $\mathbf{x}_l = \frac{1}{N}\mathbf{F}_{\mathbf{N}}^{*}\mathbf{D}_{\frac{M}{2}}\mathbf{F}_{N}\mathbf{x}$ and $\mathbf{x}_h = \mathbf{x} - \mathbf{x}_l$. $\mathbf{D}_{\mu}$ is a diagonal matrix whose the first and the last μ diagonal elements equal to 1 while others equal to 0.

$$\begin{aligned} ||\mathcal{U}\mathcal{M}\mathbf{x} - \mathbf{x}||_2^2 &= ||\mathcal{U}\mathcal{M}\mathbf{x} - \mathbf{x}_l - \mathbf{x}_h||_2^2 && (11)\\ &= ||\mathcal{U}\mathcal{M}\mathbf{x} - \mathbf{x}_l||_2^2 + ||\mathbf{x}_h||_2^2 + \langle \mathcal{U}\mathcal{M}\mathbf{x}, \mathbf{x}_h\rangle - \langle \mathbf{x}_l, \mathbf{x}_h\rangle && (12)\\ &= ||\mathcal{U}\mathcal{M}\mathbf{x} - \mathbf{x}_l||_2^2 + ||\mathbf{x}_h||_2^2 && (13)\end{aligned}$$

equation 13 holds because the third term and the forth term of equation 12 equal to 0. $\mathcal{M}$ removes high frequencies and $\mathcal{U}$ doesn't introduce new frequencies. Thus $\mathcal{U}\mathcal{M}\mathbf{x}$ only contains low frequencies. Due to the orthogonality of frequency spectrum, $\langle \mathcal{U}\mathcal{M}\mathbf{x}, \mathbf{x}_h\rangle = 0$. Similarly, $\langle \mathbf{x}_l, \mathbf{x}_h\rangle = 0$. When $\mathcal{M}$ is F-pooling, we have

$$\begin{aligned} \mathcal{U}\mathcal{M}\mathbf{x} &= \frac{1}{MN}\mathbf{F}_{\mathbf{N}}^{*}\mathbf{Z}_{\frac{M}{2}}\mathbf{F}_{M}\mathbf{F}_{\mathbf{M}}^{*}\mathbf{T}_{\frac{M}{2}}\mathbf{F}_{N}\mathbf{x} && (14)\\ &= \frac{1}{N}\mathbf{F}_{\mathbf{N}}^{*}\left(\mathbf{Z}_{\frac{M}{2}}\mathbf{T}_{\frac{M}{2}}\right)\mathbf{F}_{N}\mathbf{x} && (15)\\ &= \frac{1}{N}\mathbf{F}_{\mathbf{N}}^{*}\mathbf{D}_{\frac{M}{2}}\mathbf{F}_{N}\mathbf{x} \stackrel{def}{=} \mathbf{x}_l && (16)\end{aligned}$$

equation 16 holds due to the definition of operations $\mathbf{T}$, $\mathbf{Z}$, and $\mathbf{D}$. See appendix A for proof. Since the fist term of equation 13 equals to 0, equation 10 is minimized. Thus, we have proved that F-pooling is the optimal anti-aliasing down sampling operation.

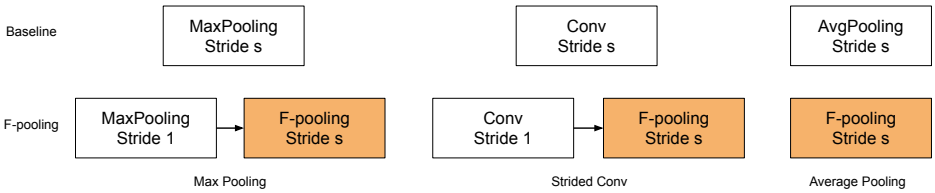


Figure 3: An illustration of how to replace max-pooling, average pooling, and strided convolution with F-pooling. We follow the settings in Zhang (2019).

2.4 SHIFT-EQUIVALENT

Based on shift theorem of Fourier transform, we have

$$\mathbf{F}_N Shift_{\triangle h}(\mathbf{x}) = \mathbf{F}_N \mathbf{x} \odot \mathbf{S}_{\triangle h} \tag{17}$$

$$Shift_{\triangle h}(\mathbf{F}_N^* \mathbf{x}) = \mathbf{F}_N^* \mathbf{x} \odot \mathbf{S}_{\triangle h} \tag{18}$$

where $\mathbf{S}_{\triangle h} \in \mathbb{C}^N$ whose kth element is $e^{-\frac{2\pi k}{N} \triangle h}$ and $\odot$ is element-wise multiplication. Combining with equation 16, we have

$$\mathcal{UM} Shift_{\triangle h}(\mathbf{x}) = \frac{1}{N} \mathbf{F}_\mathbf{N}^* \mathbf{D}_{\frac{M}{2}} \mathbf{F}_N Shift_{\triangle h}(\mathbf{x}) \tag{19}$$

$$= \frac{1}{N} \mathbf{F}_\mathbf{N}^* \left(\mathbf{D}_{\frac{M}{2}} \mathbf{F}_N \mathbf{x} \right) \odot \mathbf{S}_{\triangle h} \tag{20}$$

$$= Shift_{\triangle h} \left(\frac{1}{N} \mathbf{F}_\mathbf{N}^* \mathbf{D}_{\frac{M}{2}} \mathbf{F}_N \mathbf{x} \right) \tag{21}$$

$$= Shift_{\triangle h}(\mathcal{UM} \mathbf{x}) \tag{22}$$

We have proved that F-pooling is shift-equivalent. We highlight that equation 22 is not hold if other up sampling operations are used, such as linear interpolation.

The proofs in section 2.3 and 2.4 can be easily extended for 2D signals.

2.5 DEALING WITH IMAGINARY PART

Generally, the output of F-pooling, i.e. $\mathcal{M}\mathbf{x}$ contains both real part and imaginary part. However, for commonly used CNNs, the feature maps must be real. Thus one must ignore the imaginary part of F-pooling. On the other hand, $\mathcal{M}\mathbf{x}$ is treated as complex in the proofs in section 2.3 and 2.4. If we ignore the imaginary part, the conclusions in section 2.3 and 2.4 are no longer hold. That is F-pooling is no longer the optimal anti-aliasing down sampling (but still anti-aliasing) and F-pooling is no longer strict shift-equivalent.

Fortunately, when the resolution of down sampled signals is odd, the imaginary part of $\mathcal{M}\mathbf{x}$ is zero. Suppose $M = 2\mu + 1$, the frequencies ranged in $[-u, u]$ are retained. Due to its symmetry, $\mathcal{M}\mathbf{x}$ only contains real part. If $M = 2\mu$, the frequencies ranged in $[-u, u)$ are retained. In this case, $\mathcal{M}\mathbf{x}$ contains imaginary part. But we can recover the symmetry and eliminate the imaginary part by setting $(-u)$th frequency to zero. We call this trick *odd padding*. A drawback of *odd padding* is that more information is lost during down sampling which may reduces accuracy.

Whether $\mathcal{M}\mathbf{x}$ contains imaginary part is very important in theory. It determines the theoretical nature of F-pooling. In practice, we find there are no significant differences if we don't use *odd padding*. In figure 1, *odd padding* is used. On real-world datasets, i.e. CIFAR and ImageNet, *odd padding* is not used in this paper. We show the effects of *odd padding* in appendix B.

2.6 IMPLEMENTATION

We implement F-pooling in PyTorch Paszke et al. (2017). Theoretically, it is best to implement F-pooling with fast Fourier transform (FFT) and inverse FFT. Suppose F-pooling down samples

Table 1: Accuracy and consistency on sub-ImageNet

accuracy/consistency	densenet-121	resnet-50	mobilenet-v2
Origin	77.47/59.48	74.44/59.46	73.01/59.80
AA-pooling	77.14/60.60	**76.12**/63.16	74.03/59.44
F-pooling	**77.56/62.39**	76.05/**63.70**	**74.72/63.37**

Table 2: Accuracy and consistency on CIFAR-100 with shift augment

accuracy/consistency	densenet-41	resnet-18	resnet-34
Origin	74.90/40.83	75.52/32.44	76.56/32.05
AA-pooling	**75.55**/37.54	**77.43**/33.49	76.95/**38.56**
F-pooling	75.45/**40.90**	77.36/**40.00**	**77.68**/35.91

an $h_0 \times w_0 \times c$ tensor to a $h_1 \times w_1 \times c$ tensor. Then its time complexity is $ch_0w_0 \log(w_0h_0))$. Unfortunately, we find such a implementation in PyTorch is comparatively slow.

Instead, we implement F-pooling via 1×1 convolutions. As in equation 7, 2D F-pooling can be represented as two times matrix-matrix multiplications along vertical direction and horizontal direction respectively. This is equivalent to 1×1 convolutions along vertical direction and direction horizontal. First, we use two $1 \times 1 \times w_0 \times w_1$ convolutions to compute the real part and imaginary part of $\mathbf{X}_{:,:,i}\mathbf{P}_w^*$ in equation 7. This requires $2ch_0w_0w_1$ float multiplications. Now $\mathbf{X}$ is mapped to an $h_0 \times w_1 \times c$ tensor. We use another two $1 \times 1 \times h_0 \times h_1$ convolutions to compute the real part of $\mathbf{Y}$. This requires $2ch_0h_1w_1$ float multiplications. Thus the total number of float multiplications is $2cw_1(h_0w_0 + h_0h_1)$. If we first use two $1 \times 1 \times h_0 \times h_1$ convolutions then use two $1 \times 1 \times w_0 \times w_1$ convolutions, the total number of float multiplications becomes $2ch_1(h_0w_0 + w_0w_1)$. Its time complexity is $\mathcal{O}(h_0w_0(h_1 + w_1))$ which is higher than the optimal complexity.

F-pooling requires much more computational costs than average pooling or max-pooling. F-pooling is faster than a convolutional layer when the resolution of feature maps is smaller than the number of channels, as the situations of image classifications. Moreover, the number of pooling layers is limited compared with the number of convolutional layers. Thus, introducing F-pooling will not introduce too many computational costs into CNNs.

Zhang (2019) claims that it is important to integrate anti-aliasing pooling into CNNs in a correct way. In this paper, we follow their settings. Specifically, a max-pooling with stride s is replaced with a max-pooling with stride 1 and an F-pooling with stride s. A convolution with stride s is replaced with a convolution with stride 1 and an F-pooling with stride s. An average pooling with stride s is replaced with an F-pooling with stride s. See the illustration in figure 3.

3 EXPERIMENTS

3.1 1D SIGNALS

We test the shift-equivalent of F-pooling on 1D signals. In the first row of figure 1, the original signal is a sine curve with period 16 and 256 discrete elements. In the second row of figure 1, the original signal is a randomly selected row of a 512×512 image. We apply max-pooling, average pooling, and F-pooling with stride 4 to those signals. As shown in figure 1, F-pooling is perfectly shift-equivalent. And average pooling is better than max-pooling from shift-equivalent perspective. *odd padding* is used here.

3.2 IMAGE CLASSIFICATION

CIFAR-100:[1] we test classification of low-resolution 32×32 color images. This dataset contains 50k images for training and 10k images for test. Images are classified into one of 100 categories.

[1]https://www.cs.toronto.edu/ kriz/cifar.html

Table 3: Accuracy and consistency on CIFAR-100 without shift augment

accuracy/consistency	densenet-41	resnet-18	resnet-34
Origin	71.81/33.67	67.60/24.58	68.11/23.54
AA-pooling	**73.81**/34.45	**74.49**/28.68	**74.00**/29.58
F-pooling	73.19/**35.20**	73.93/**32.21**	73.51/**31.48**

sub-ImageNet: we then test classification of high-resolution 224×224 color images. Original ImageNet dataset Russakovsky et al. (2015) contains 1.28M training and 50k validation images, classified into one of 1,000 categories. Each category has 1,280 images in average. To reduce the computational resources for training, we use a subset of ImageNet (sub-ImageNet) in this work. We randomly select 200 categories from 1,000 categories. For each category, we randomly select 500 images. Thus, we collect 100k images for training. And we select corresponding 10k validation images. All models are trained on a single GPU with batchsize 64 and 100 epochs. We decrease the initial learning rate by a factor 10 every 40 epochs. For other hyper-parameters, we follow PyTorch's official training script.[2]

We train CIFAR-100 using resnet-18 He et al. (2016) and densenet-41 Huang et al. (2017). We train sub-ImageNet using resnet-50 He et al. (2016), densenet-121 Huang et al. (2017), and mobilenet-v2 Sandler et al. (2018). Those models are widely used as benchmarks. Max-pooling, average pooling, and strided convolution are covered in those models. We also compare F-pooling with AA-pooling Zhang (2019). For a model trained on sub-ImageNet, its first down sampling operation is kept to reduce computational costs. This setting is also used in Zhang (2019). The results on CIFAR-100 are averaged by 3 runs. *odd padding* is not used for both datasets.

We study not only accuracy but also consistency. We follow the metric of consistency used in Zhang (2019): we check how often the network outputs the same classification, given the same image with two different shifts.

$$\mathbb{E}_{h_0,h_1,w_0,w_1} \mathbf{1} \left\{ \arg\max p(Shift_{h_0,w_0}) = \arg\max p(Shift_{h_1,w_1}) \right\} \tag{23}$$

We only evaluate diagonal shifts in this paper. For CIFAR-100, the number of shifted pixels ranges from -7 to 7. For sub-ImageNet, it ranges from -31 to 31.

Accuracy and consistency on sub-ImageNet is shown in table 1. See appendix C for loss curves. Results on CIFAR-100 with shift augment are shown in table 2. Results on CIFAR-100 without shift augment are shown in table 3. . Based on those results, we conclude that

- F-pooling is significantly and consistently better than original pooling methods in terms of accuracy and consistency.
- F-pooling is comparable with AA-pooling in term of accuracy. F-pooling is better than AA-pooling in term of consistency.
- Consistency on ImageNet for all methods is much better than CIFAR-100's. A probable cause is that the resolution of CIFAR-100 images is too low.

3.3 STRICT SHIFT-EQUIVALENT CNNS

CNNs with F-pooling in section 3.2 are not strict shift-equivalent. There are two reasons: 1) *odd padding* is not used. When we ignore the imaginary part of F-pooling output, strict shift-equivalent is destroyed; 2) Convolution layers are not strict shift-equivalent because of padding. The smaller the resolution, the larger effects of padding. Correspondingly, we can make the whole CNN strict shift-equivalent by: 1) use *odd padding* or set the resolution of F-pooling's output to odd; 2) set the convolutional kernel at all layers to 1×1. This has been verified by our experiments. But we have not yet trained such CNNs on CIFAR-100 or ImageNet.

[2]https://github.com/pytorch/examples/tree/master/imagenet

4 RELATED WORKS

Pooling which reduces the resolution of feature maps is an important part of CNNs. Early CNNs Lecun et al. (1989) use average pooling which is good for anti-aliasing. Later empirical evidence suggests max-pooling Scherer et al. (2010) and strided-convolutions Long et al. (2015) provide better performance. However, small shifts in the input can drastically change the output when stacking multiple max-pooling or strided-convolutions Engstrom et al. (2017); Azulay & Weiss (2018). Other variants such as Graham (2014); He et al. (2015); Lee et al. (2016) (we just list a few of them), focus on extending the functionality of pooling Lee et al. (2016) or making pooling adjusted to variable input size Graham (2014); He et al. (2015).

Recently, Zhang (2019) shows that CNNs will have better shift-equivalent and anti-aliasing properties when low-pass filtering is integrated correctly. However, their method is not strict shift-equivalent and anti-aliasing in theory. Williams & Li (2018) propose Wavelet-pooling. They decompose a signal via wavelet transform and retain the lowest sub-band. This process is repeated until the designed down sampling factor is met. The spirit of Wavelet-pooling is similar to F-pooling. However, they claim that Wavelet-pooling is better than others because it is a global transform instead of a local transform. They neither focus on shift-equivalent nor prove Wavelet-pooling is shift-equivalent or not.

F-pooling is a complex transformation because DFT and IDFT are involved in it. Many works integrate complex transformations or complex values into neural networks. Amin & Murase (2009) study single-layered complex-valued neural networks for real-valued classification problems. Complex numbers represented in polar coordinates are more suitable to deal with rotations. Based on this, Cohen et al. (2018) propose spherical CNNs which are rotation-equivalent to deal with signals projected from spherical surface. Trabelsi et al. (2018) propose general deep complex CNNs. They adjust batch normalization and non-linear activations for complex CNNs. F-pooling utilizes shift theorem of DFT and achieves shift-equivalent. This is a new success for the combination of complex transformations and neural networks.

5 CONCLUSIONS

In this paper, we have proposed frequency pooling (F-pooling) for CNNs. As the name suggested, F-pooling reduces the dimension of signals in frequency domain. We have defined shift-equivalent of functions which contain down sampling operations by introducing an up sampling operation. Under this definition, we have proved that F-pooling is the optimal anti-aliasing down sampling operation and F-pooling is shift-equivalent. We have integrated F-pooling into modern CNNs. We have verified that F-pooling remarkably increases accuracy and robustness w.r.t shifts of modern CNNs.

Several issues, such as the padding of convolutions and the loss of imaginary part of F-pooling, make the whole model shift-nonequivalent. We will train CNNs which are shift-equivalent as a whole in the future.

REFERENCES

Md Faijul Amin and Kazuyuki Murase. Single-layered complex-valued neural network for real-valued classification problems. *Neurocomputing*, 72(4-6):945–955, 2009.

Aharon Azulay and Yair Weiss. Why do deep convolutional networks generalize so poorly to small image transformations? *arXiv preprint arXiv:1805.12177*, 2018.

Taco S. Cohen, Mario Geiger, Jonas Khler, and Max Welling. Spherical CNNs. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=Hkbd5xZRb`.

Logan Engstrom, Brandon Tran, Dimitris Tsipras, Ludwig Schmidt, and Aleksander Madry. A rotation and a translation suffice: Fooling cnns with simple transformations. *arXiv preprint arXiv:1712.02779*, 2017.

Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT Press, 2016.

Benjamin Graham. Fractional max-pooling. *arXiv preprint arXiv:1412.6071*, 2014.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9):1904–1916, 2015.

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

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*, pp. 4700–4708, 2017.

Yann Lecun, Bernhard E Boser, John S Denker, D Henderson, R E Howard, W Hubbard, and L D Jackel. Handwritten digit recognition with a back-propagation network. pp. 396–404, 1989.

Chenyu Lee, Patrick W Gallagher, and Zhuowen Tu. Generalizing pooling functions in convolutional neural networks: Mixed, gated, and tree. pp. 464–472, 2016.

Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. pp. 3431–3440, 2015.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *Computer Science*, 2013.

Harry Nyquist. Certain topics in telegraph transmission theory. *Transactions of The American Institute of Electrical Engineers*, 47(2):617–644, 1928.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. doi: 10.1007/s11263-015-0816-y.

Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4510–4520, 2018.

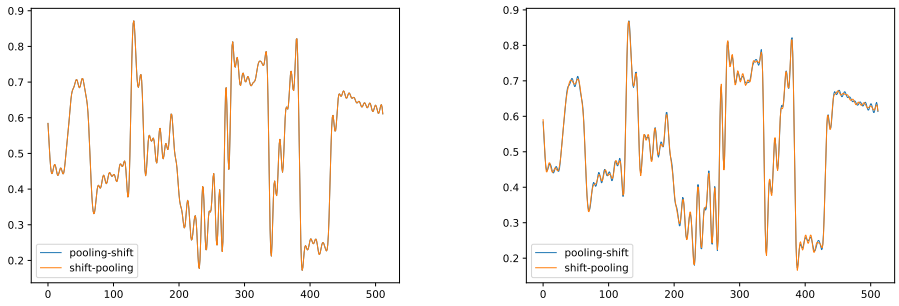
Dominik Scherer, Andreas Muller, and Sven Behnke. Evaluation of pooling operations in convolutional architectures for object recognition. pp. 92–101, 2010.

Chiheb Trabelsi, Olexa Bilaniuk, Ying Zhang, Dmitriy Serdyuk, Sandeep Subramanian, Joao Felipe Santos, Soroush Mehri, Negar Rostamzadeh, Yoshua Bengio, and Christopher J Pal. Deep complex networks. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=H1T2hmZAb`.

Travis Williams and Robert Li. Wavelet pooling for convolutional neural networks. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=rkhlb8lCZ`.

Wenpeng Yin, Katharina Kann, Mo Yu, and Hinrich Schutze. Comparative study of cnn and rnn for natural language processing. *arXiv: Computation and Language*, 2017.

Richard Zhang. Making convolutional networks shift-invariant again. *arXiv preprint arXiv:1904.11486*, 2019.



(a) with *odd padding* (b) without *odd padding*

Figure 4: With *odd padding*, F-pooling is strict shift-equivalent. Without it, F-pooling is not strict shift-equivalent. But the error is acceptable. Best viewed on screen.

A PROOF OF OPERATION

We prove $\mathbf{D}_{\frac{M}{2}} = \mathbf{Z}_{\frac{M}{2}} \mathbf{T}_{\frac{M}{2}}$. By definition, $\mathbf{T}$ can be represented as an $M \times N$ matrix and $\mathbf{Z}$ can be represented as a $N \times M$ matrix.

$$\mathbf{T}_{\frac{M}{2}} = \begin{bmatrix} \mathbf{I}_{\frac{M}{2}} & \mathbf{0}_{\frac{M}{2}} & \mathbf{0}_{\frac{N-M}{2}} & \mathbf{0}_{\frac{N-M}{2}} & \mathbf{0}_{\frac{M}{2}} & \mathbf{0}_{\frac{M}{2}} \\ \mathbf{0}_{\frac{M}{2}} & \mathbf{0}_{\frac{M}{2}} & \mathbf{0}_{\frac{N-M}{2}} & \mathbf{0}_{\frac{N-M}{2}} & \mathbf{0}_{\frac{M}{2}} & \mathbf{I}_{\frac{M}{2}} \end{bmatrix} \tag{24}$$

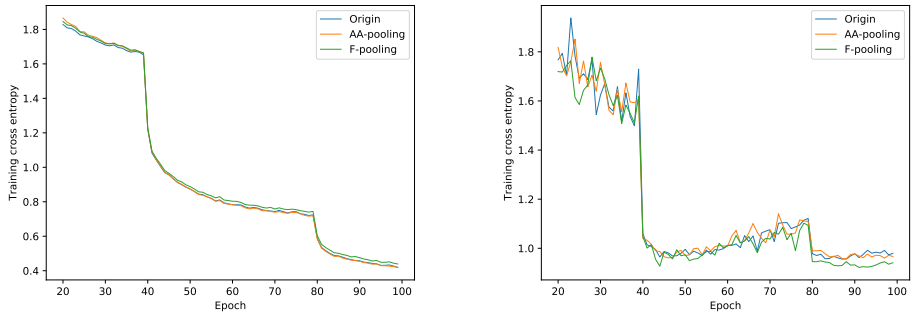
where $\mathbf{I}_M$ is an $M \times M$ identity matrix and $\mathbf{0}_M$ is an $M \times M$ zero matrix. $\mathbf{Z}_{\frac{M}{2}}$ is equal to the transpose of $\mathbf{T}_{\frac{M}{2}}$ by definition. Then

$$\mathbf{Z}_{\frac{M}{2}} \mathbf{T}_{\frac{M}{2}} = \begin{bmatrix} \mathbf{I}_{\frac{M}{2}} & \cdot & \cdot \\ \cdot & \mathbf{0}_{N-M} & \cdot \\ \cdot & \cdot & \mathbf{I}_{\frac{M}{2}} \end{bmatrix} \tag{25}$$

which is the same as $\mathbf{D}_{\frac{M}{2}}$ by definition.

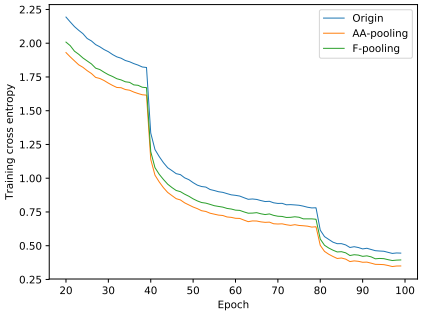
B EFFECT OF *odd padding*

C LOSS CURVES ON SUB-IMAGENET

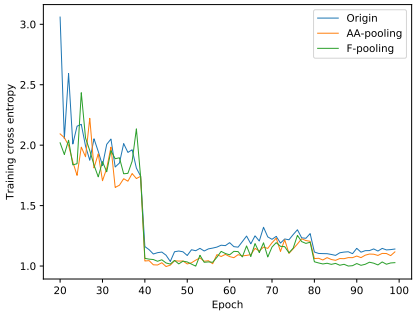


(a) densenet-train

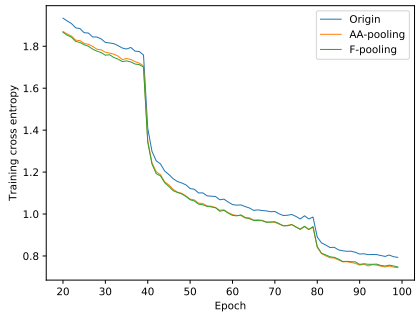
(b) densenet-test



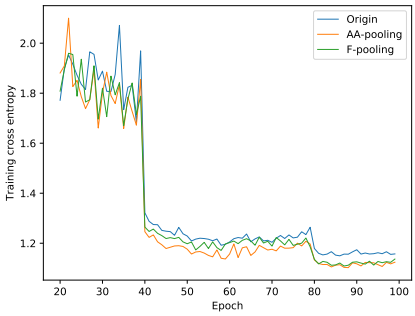
(c) resnet-train



(d) resnet-test



(e) mobile-train



(f) mobile-test

Figure 5: Loss curves on sub-ImageNet. Best viewed on screen.