Review: Spectral Representations for Convolutional Neural Networks

James Smith
Department of Electrical and Computer Engineering
Auburn University
jss0036@auburn.edu

Mason Wishum
Department of Computer Science
Auburn University
mlw0032@auburn.edu

1 Paper Summary

By utilizing powerful properties of the Discrete Fourier Transform (DFT) and Fourier domain, Rippel et al. present spectral pooling and spectral parameterization of convolutional layers as a means to improve convolutional neural networks (CNNs) by improving training convergence, allowing flexible pooling dimensions, and retaining or improving competitive classification accuracies (Rippel et al. (2015)).

Spectral pooling truncates the spectral representation of a image-kernel product. Simply put, spectral pooling is simple low-pass filter, as described in Algorithm 1. The gradient backpropagation is simply the spectral pooling performed backwards, with zero-padding applied to invert the truncation step. This technique is desirable because it can be combined with the convolution theorem to achieve fast training results. The convolution theorem states that convolution can be sped considerably by being performed in the spectral domain as element–wise multiplication. This is given as:

$$\mathscr{F}\{f * g\} = \mathscr{F}\{f\} \cdot \mathscr{F}\{g\} \tag{1}$$

The transformation to the spectral domain can take advantage of the radix–2 based Cooley–Tukey Fast Fourier Transform (FFT) algorithm (Cooley and Tukey (1965)). By using the FFT, the time complexity for the respective convolution techniques assuming a l_1 x l_2 kernel and m_1 x m_2 image are:

$$T_{\text{spatial}} = O\left(l_1 l_2 m_1 m_2\right) \tag{2}$$
$$T_{\text{spectral}} = O\left(n_1 n_2 log\left(n_1 n_2\right)\right). \tag{3}$$

where:

$$n_i = l_i + m_1 + 1. \tag{4}$$

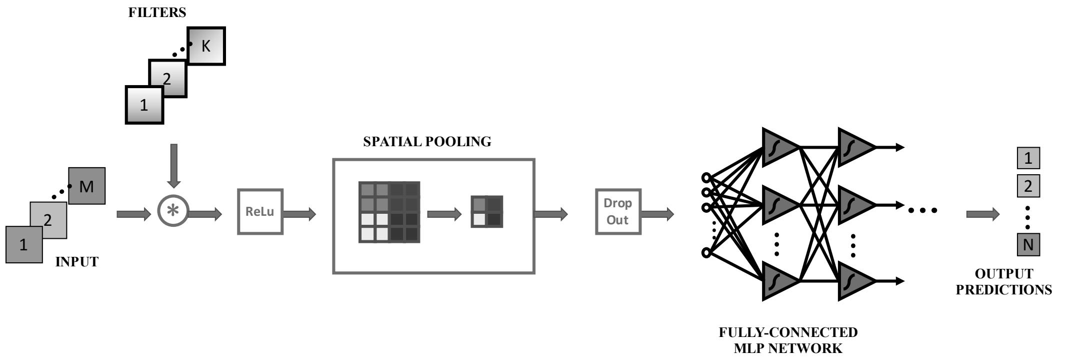
Reducing the time complexity gives a scalable advantage as data size increases. By combining the two steps of spectral pooling and convolution, the presented network and the commonly used max pooling network are visualized in Fig.1.

Spectral parameterization offers further advantages for spectral pooling in that kernels do not have to be transformed into the frequency domain. This is done by learning the filter weights in the spectral domain rather than the spatial domain. Spectral parameterization saves computation in that only two transformations must take place in each instance (the image to spectral domain and then the inverse transformation of the resulting product).

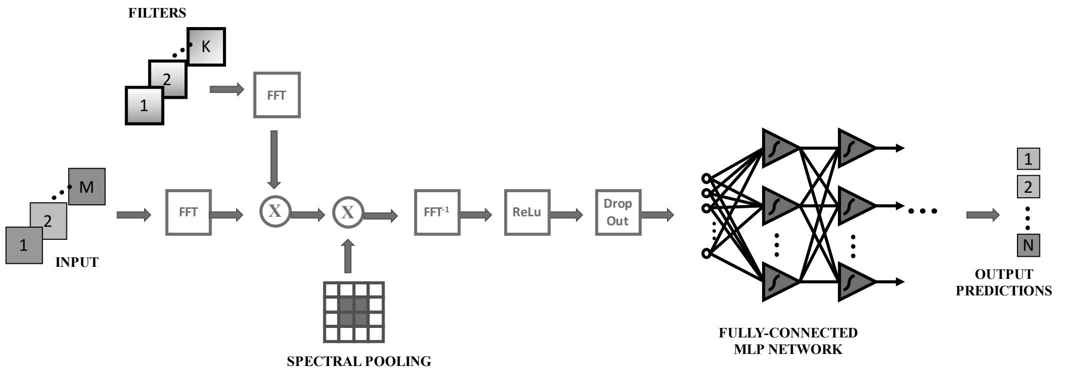
2 Strengths

This paper provided both a strong theoretical insight and empirical proof for the spectral pooling technique. The outcome gives a fast and effective pooling method that is easily implementable in any

```
Algorithm 1 Spectral Pooling
Input: $X \in c$ x $m_1$ x $m_2$ image
Output: $Y \in c$ x $p_1$ x $p_2$
  compute FFT of X
  truncate X to size $c$ x $p_1$ x $p_2$
  return IFFT of X
```



(a) Spatial Pooling



(b) DFT Spectral Pooling

Figure 1: Convolution Neural Network Architecture Implementation for Various Pooling Methods

library. Specifically, the advantages of the paper's methods are: fast convolution computations, high parameter information retention (compression), flexibility in pooling output dimensions, and further computation savings by implementing spectral parameterization. The experiments given in the paper were well executed, but had shortcomings that are discussed in the next section.

In order to validate the results presented in the paper, the pooling technique was implemented in a vanilla network using standard Matlab statements and used on the MNIST dataset (Lecun et al. (1998)). Stochastic gradient descent with moment was used to train a simple convolutional network with 64 3x3 filters and a single fully connected layer at the end. Grid-search was used to tune hyper-parameters and each network was trained for 50 epochs with results averaged over 10 epochs. Our experiments are consistent with the paper's findings, but we acknowledge that these experiments are weak (done in short time-period) and could therefore be misleading.

3 Weaknesses

The greatest weakness of this paper is the lack of proof substantiating translation invariance for this pooling technique. The paper sells its idea as a method to pact high amounts of information into few

Table 1: Training Results for Various Pooling Methods

Pooling	Stride	Size	Truncation (%)	Reduction (%)	Training Time	Classifaction Accuracy (%)
Max	2	2	~	75	4236.5	70.4
Mean	2	2	~	75	4265.3	72.6
FFT	~	~	75	75	2438.5	73.4

parameters, but the intuitive proof of max pooling is that it saves only the important information (i.e., we do not want to save all information about an image).

Additionally, the method suffers from inability to be implemented in a tensor-based library. The FFT algorithm is recursive and must include many overhead computations. It should be observed that the transformation can be realized with a linear operator and therefore will at worst perform as computationally efficient as spatial convolution. This means that spatial pooling can be implemented with tensor methods without significantly adding computations.

Another weakness is the inability to differentiate this work from standard compression methods. Some other work uses similar means to compress network parameters (Wang et al. (2016)). This paper should address how this pooling technique is different from simply compressing images. Is the novelty that the pooling occurs in intermediate layers, whereas compression would only occurs as a data pre-processing?

Finally, the paper does not present enough experimental results to overcome the intuitive weaknesses. As authors, we would have found a more difficult dataset to use and compared with more state of the art networks rather than standards that can already be beat.

4 Future work

In addition to future directions discussed in the previous section, an interesting possible extension of this paper is what would happen if you were to build an entire network in the frequency domain. An intermediate solution mentioned in the paper is wavelets (Bruna and Mallat (2013)).

To further extend the before-mentioned topic of compression versus pooling, another direction this work could go in would be to apply the spectral pooling along with max pooling layers. Spectral pooling could be used as a great dimensionality reduction method, especially in the initial convolutional layers. Images of arbitrary size could be inserted into the network and the first layer's output would contain compressed, informative parameters of a specified size. This would retain the advantages of spectral pooling while still retain max pooling as a method to offer translation invariance.

5 Questions and Answers

Question 1 Could you describe the process of SP to Computer Science (non EE) students in 1-2 sentences?

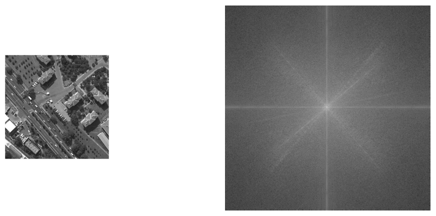
First we apply a domain transformation algorithm (the Fourier Transform and some extra math[1]) to represent the image in a form where information we are more likely to care about is concentrated in the center of the new image representation. We then crop the output and apply the reverse of that domain transformation to get back a normal image of smaller size.

Question 2 Do you know any existing (on line) interactive demo that you know could help them understand FT (e.g. how changing an image in spatial space affects the resulting magnitude/phase image?)

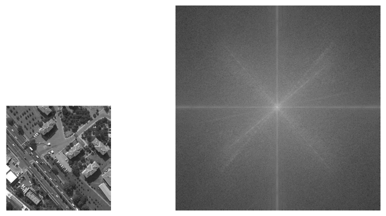
This is a great intuitive explanation behind 2D Fourier theory:

http://cns-alumni.bu.edu/~slehar/fourier/fourier.html

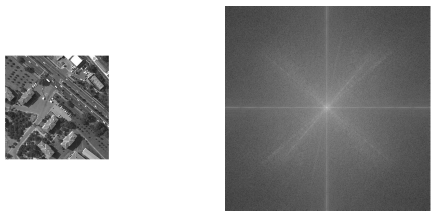
[1] We shift the resulting transformed image to put zero-frequency values in the center to satisfy complex-conjugate symmetrical constraints



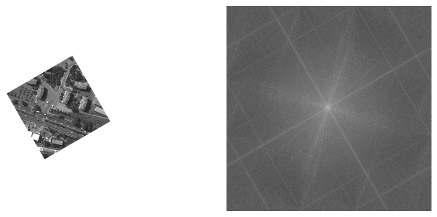
(a) Centered image



(b) Shifted image



(c) Flipped image



(d) Rotated image

Figure 2: Spatial (left) and frequency (right) representations of various image transformations

Here is a video explanation of the Fourier Transform:
https://www.youtube.com/watch?v=spUNpyF58BY

Perhaps the best tool to try is hands-on Matlab demos:
https://www.mathworks.com/help/images/fourier-transform.html

Question 3 Does SP offer translation invariance (like max pooling offers)?

First, spectral pooling can perform no worse in this aspect than compared to a network without pooling. Assuming that a network with no pooling offers some translation invariance (Goodfellow et al. (2016)), then we must assume that spectral pooling offers the same advantage. The intuition behind this is that the same convolution operation is applied; the only further operation performed is practically a network compression. That is, the parameters left are to represent the network as if no pooling had taken place at all (filtering out noise).

Max pooling provides the majority of the translation invariance in CNNs by pooling together activations across a given activation region. Intuitively, it can be argued that spectral pooling offers even greater translation invariance by not limiting feature detection to a small activation region. Empirically, we demonstrate this by analyzing the frequency domain with several image shifts. Fig.1 gives the spectral representation of an image as it is scaled, shifted, and rotated. The only case of concern is the image rotation; although the local shape is the same, the spectra are rotated. However, this is also a drawback of max pooling networks as well. Given the ability of spectral pooling to handle shifting features, we are confident from an intuitive perspective that spectral pooling offers translation invariance. It will take experimental results to empirically demonstrate this, and no experiments have been conducted to our best knowledge.

Question 4 How many more parameters does SP require (max pooling has 0 params), and what are they?

There are no learned parameters in spectral pooling. Similar to max pooling, it only needs a couple of parameters to specify the output size. Specifically, spectral pooling takes a map of size $M \times N$ (the input data) and a output size of $H \times W$ as its only inputs.

Question 5 Do we use the phase image at all during SP? (or only magnitude image) and why?

Spectral pooling uses both the magnitude and phase; when implemented correctly, only real-valued spatial coefficients should be returned. When taking the Fourier transformation of a 2-D image, the resulting coefficients are given as complex conjugate pairs. This is implemented by shifting the coefficients to place the zero-frequency components at the center. When spectral pooling is applied, the saved complex coefficients will be complex conjugate pairs around a center point and will result in only real value spatial representations This can also be a test to be sure spectral pooling is being correctly implemented: imaginary spatial values implies that a mistake has been made.

References

O. Rippel, J. Snoek, and R. P. Adams. Spectral Representations for Convolutional Neural Networks. *arXiv:1506.03767 [cs, stat]*, June 2015. URL http://arxiv.org/abs/1506.03767. arXiv: 1506.03767. 1

J. W. Cooley and J. W. Tukey. An Algorithm for the Machine Calculation of Complex Fourier Series. *Mathematics of Computation*, 19(90):297–301, 1965. ISSN 0025-5718. doi: 10.2307/2003354. URL http://www.jstor.org/stable/2003354. 1

Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, November 1998. ISSN 0018-9219. doi: 10.1109/5.726791. 2

Y. Wang, C. Xu, S. You, D. Tao, and C. Xu. CNNpack: Packing Convolutional Neural Networks in the Frequency Domain. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 253–261. Curran Associates, Inc., 2016. URL http://papers.nips.cc/paper/6390-cnnpack-packing-convolutional-neural-networks-in-the-frequency-domain.pdf. 3

J. Bruna and S. Mallat. Invariant scattering convolution networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1872–1886, Aug 2013. ISSN 0162-8828. doi: 10.1109/TPAMI.2012.230. 3

I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. 4