Band-limited Training and Inference for Convolutional Neural Networks

Adam Dziedzic*, John Paparrizos*, Sanjay Krishnan, Aaron Elmore, Michael Franklin
Natural images
More information put in lower frequencies

Original image

Spatial domain

Frequency domain
Natural images
Transformations between the domains

Compression 50%

Spatial domain  Frequency domain
Method for ConvNets to constrain the frequency band in convolution operation for efficiency

Compression 50% in practice
FFT based convolution

Mathieu et al.: “Fast Training of Convolutional Networks through FFTs”
FFT based convolution

Mathieu et al.: “Fast Training of Convolutional Networks through FFTs”
FFT based convolution

Mathieu et al.: “Fast Training of Convolutional Networks through FFTs”

Data: x

FFT(x)

Filter: y

FFT(y)

xfft

yfft

xfft ⊙ yfft

offt

IFFT(offt)

Out: o
FFT based convolution

Mathieu et al.: “Fast Training of Convolutional Networks through FFTs”
FFT based convolution

Mathieu et al.: “Fast Training of Convolutional Networks through FFTs”
cuDNN: Substantial memory workspace needed for intermediate results.
Band-limited FFT based convolution

**Band-limiting** = masking out high frequencies

Data: $x$

1. $\text{FFT}(x)$
2. $\text{FFFT}(x)$
3. **Band-limited** ($\text{FFFT}(x)$)
4. $x \cdot \text{CFFT}$
5. $\text{IFFT}(\text{FFFT}(x))$

Filter: $y$

6. $\text{FFT}(y)$
7. $\text{FFFT}(y)$
8. **Band-limited** ($\text{FFFT}(y)$)
9. $y \cdot \text{CFFT}$
10. $\text{IFFT}(\text{FFFT}(y))$

Out: $o$

Band-limiting = masking out high frequencies
Band-limited FFT based convolution

Data: \( x \)  
FFT(\( x \)) \( \rightarrow \) \( xfft \)  
\( \text{Band-limited} \)  
(\( xfft \))  
\( \rightarrow \) \( xCfft \)  
\( \text{Less memory used} \)  
IFFT(\( offt \))  
\( \text{Out:} \ o \)

Filter: \( y \)  
FFT(\( y \)) \( \rightarrow \) \( yfft \)  
\( \text{Band-limited} \)  
(\( yfft \))  
\( \rightarrow \) \( yCfft \)  
\( xCfft \)  
\( \circ \)  
\( yCfft \)  
\( \rightarrow \) \( offt \)  
IFFT(\( offt \))  
\( \text{Out:} \ o \)
Band-limited FFT based convolution

Data: x
FFT(x) → xfft

Band-limited (xfft)

Filter: y
FFT(y) → yfft

Band-limited (yfft)

xcfft ⊙ ycfft

Less memory used

Out: o
IFFT(offt)

Faster computation
Preserve enough of the spectrum to retain high accuracy of models.
Band-limiting Technique

1. FFT of an input data
Band-limiting Technique

1. FFT of an input data
2. Conjugate symmetry
Band-limiting Technique

1. FFT of an input data
2. Conjugate symmetry

1+j
Band-limiting Technique

1. FFT of an input data
2. Conjugate symmetry
3. Real values
Band-limiting Technique

1. FFT of an input data
2. Conjugate symmetry
3. Real values
4. No constraints
Band-limiting Technique

1. FFT of an input data
2. Conjugate symmetry
3. Real values
4. No constraints
5. 1\textsuperscript{st} compression
Band-limiting Technique

1. FFT of an input data
2. Conjugate symmetry
3. Real values
4. No constraints
5. 1\textsuperscript{st} compression
Band-limiting Technique

1. FFT of an input data
2. Conjugate symmetry
3. Real values
4. No constraints
5. 1\textsuperscript{st} compression
6. 2\textsuperscript{nd} compression
Band-limiting Technique

1. FFT of an input data
2. Conjugate symmetry
3. Real values
4. No constraints
5. 1\textsuperscript{st} compression
6. 2\textsuperscript{nd} compression
Effects of band-limiting on accuracy

Test Accuracy (%) vs Compression rate (%)

ResNet-18 on CIFAR-10
Effects of band-limiting on accuracy

Test Accuracy (%) vs. Compression rate (%) for ResNet-18 on CIFAR-10

93.5% accuracy achieved.
Effects of band-limiting on accuracy

ResNet-18 on CIFAR-10

Test Accuracy (%) vs Compression rate (%) graph.
Effects of band-limiting on accuracy

- **ResNet-18 on CIFAR-10**
  - Test Accuracy: 93.5% to 92%

- **DenseNet-121 on CIFAR-100**
  - Test Accuracy: 75.3% to 71.2%
Main **take-aways** from Band-limited CNNs

- Method to constrain the frequency band in convolution.
- Models trained with band-limiting **gracefully degrade** the accuracy as the function of the compression rate.
- Effectively **control resource usage** (GPU/CPU and memory).
- The **low frequency** coefficients learned first during training.
- The **same compression rate** applied to training and inference.
- The more band-limited model, the more **robust to attacks**.
- Applicable to **other domains**: time-series & speech data.
Thank you

Poster: 6:30-9:00 PM @ Pacific Ballroom #132
github.com/adam-dziedzic/bandlimited-cnns
ady@uchicago.edu
Why is FFT based convolution important?

- The theoretical properties of the Fourier domain are well understood. No such properties in other domains (Winograd).
- ResNet and DenseNet architectures use 7x7 filters in first layers.
- FFT based convolution can be combined with spectral pooling.
- Band-limiting minimize aliasing & serves as a simple defense.
- A standard algorithm included in popular frameworks (cuDNN).
- Gradient acts as a large filter in the backward pass.
- Zlateski et al. suggest using FFT based convolution on CPUs.
- The 1D FFT convolution for DSP where large filters are used.
Cross-correlate input data and filter: $x \ast_c y$

$$F_x[\omega] = F(x[n]) \quad F_y[\omega] = F(y[n])$$

$$x \ast_c y = F^{-1}(F_x[\omega] \odot F_y[\omega])$$

Spectrum of convolution: $S[\omega] = F_x[\omega] \odot F_y[\omega]$

$$M_c[\omega] = \begin{cases} 1, \omega \leq c \\ 0, \omega > c \end{cases}$$

$$x \ast_c y = F^{-1}[(F_x[\omega] \odot M_c[\omega]) \odot (F_y[\omega] \odot M_c[\omega])]$$

$$x \ast_c y = F^{-1}(S[\omega] \odot M_c[\omega])$$

**Energy** (Parseval’s theorem): $\sum_{n=0}^{N-1} |x[n]|^2 = \sum_{\omega=0}^{2\pi} |F_x(\omega)|^2$
Robustness to noise

Test accuracy (%) vs. Level of Gaussian noise (sigma)

- Blue: FP32-C=0% full spectra
- Gray: FP16-C=0% full spectra (reduced precision: 16 bits)
- Orange: FP32-C=0% early stopping
- Green: FP32-C=50% band-limited
- Red: FP32-C=85% band-limited
Compression Rate for Training vs Inference

DenseNet-121 on CIFAR-100

Test accuracy (%)

Inference Compression Rate (%)

Train compression: C=50, C=75
Compression Rate for Training vs Inference

DenseNet-121 on CIFAR-100

Test accuracy (%) vs Inference Compression Rate (%)

Train compression: C=0, C=50, C=75, C=85
Effectively control resource usage

ResNet-18 on CIFAR-10

Normalized performance (%)

GPU memory allocated

Normalized performance (%)

Epoch time

Compression rate (%)
Compression Rate for Training vs Inference

Inference Compression Rate (%)

Test accuracy (%)

Train Compression Rate (%):

\[ C=0 \]

ResNet-18 on CIFAR-10
Compression Rate for Training vs Inference

Test accuracy (%) vs Inference Compression Rate (%)

ResNet-18 on CIFAR-10

Train Compression Rate (%):
- C=0
- C=85
Test accuracy (%) vs Inference Compression Rate (%) for ResNet-18 on CIFAR-10.

Smooth degradation of accuracy during inference.
Compression Rate for Training vs Inference

ResNet-18 on CIFAR-10

Apply the same compression rate to training and inference
Tuning: Accuracy vs Higher Performance

- **ResNet-18 on CIFAR-10**: Test Accuracy (%) vs Compression rate (%)
  - Accuracy decreases as compression rate increases.
  - Approximately 95% accuracy at 0% compression.

- **DenseNet-121 on CIFAR-100**: Test Accuracy (%) vs Compression rate (%)
  - Accuracy decreases as compression rate increases.
  - Approximately 70% accuracy at 0% compression.

- **GPU Memory Allocated**: GPU memory usage decreases as compression rate increases.
  - Approximately 100% GPU memory at 0% compression.

- **Epoch Time**: Epoch time increases as compression rate increases.
  - Approximately 0 epoch time at 0% compression.

Note: These graphs illustrate the trade-off between model accuracy and computational efficiency.
ResNet-18 on CIFAR-10

Test accuracy (%) vs Compression ratio (%)

- fixed compression
- energy based compression

DenseNet-121 on CIFAR-100

Test accuracy (%) vs Compression ratio (%)

- fixed compression
- energy based compression
“Speaking of longer term, it would be nice if the community migrated to a fully open sourced implementation for all of this [convolution operations, etc.]. This stuff is just too important to the progress of the field for it to be locked away in proprietary implementations. The more people working together on this the better for everyone. There's plenty of room to compete on the hardware implementation side.”

Scott Gray

https://github.com/soumith/convnet-benchmarks/issues/93