Phasor-Driven Acceleration for FFT-based CNNs

Eduardo Reis* (Presenter)
Dept. of Electrical and Computer Eng.
Lakehead University

Thangarajah Akilan (Supervisor)
Dept. of Software Eng.
Lakehead University

Mohammed Khalid (Collaborator)
Dept. of Electrical and Computer Eng.
University of Windsor

*edreis@lakeheadu.ca

Check me on Github!

Summary

- Recent research in Deep Learning (DL) has investigated the use of the Fast Fourier Transform (FFT) to accelerate the computations involved in Convolutional Neural Networks (CNNs)
  - Traditionally, this approach the rectangular form to represent complex numbers.
- In this paper, we propose using the phasor form—a polar representation of complex numbers, as a more efficient alternative to the traditional approach.
- Given the modular aspect of our approach, the proposed method can be applied to any existing convolution-based DL model without design changes.

Objectives

- Speedup training and inference of FFT-Based CNNs methods.
- Develop a platform agnostic approach.
- Develop a modular and easy to use approach.

Challenges

- Reduce the number of floating point operations.
- Research vs Application

Datasets

- CIFAR-10
- CIFAR-100

Methodology

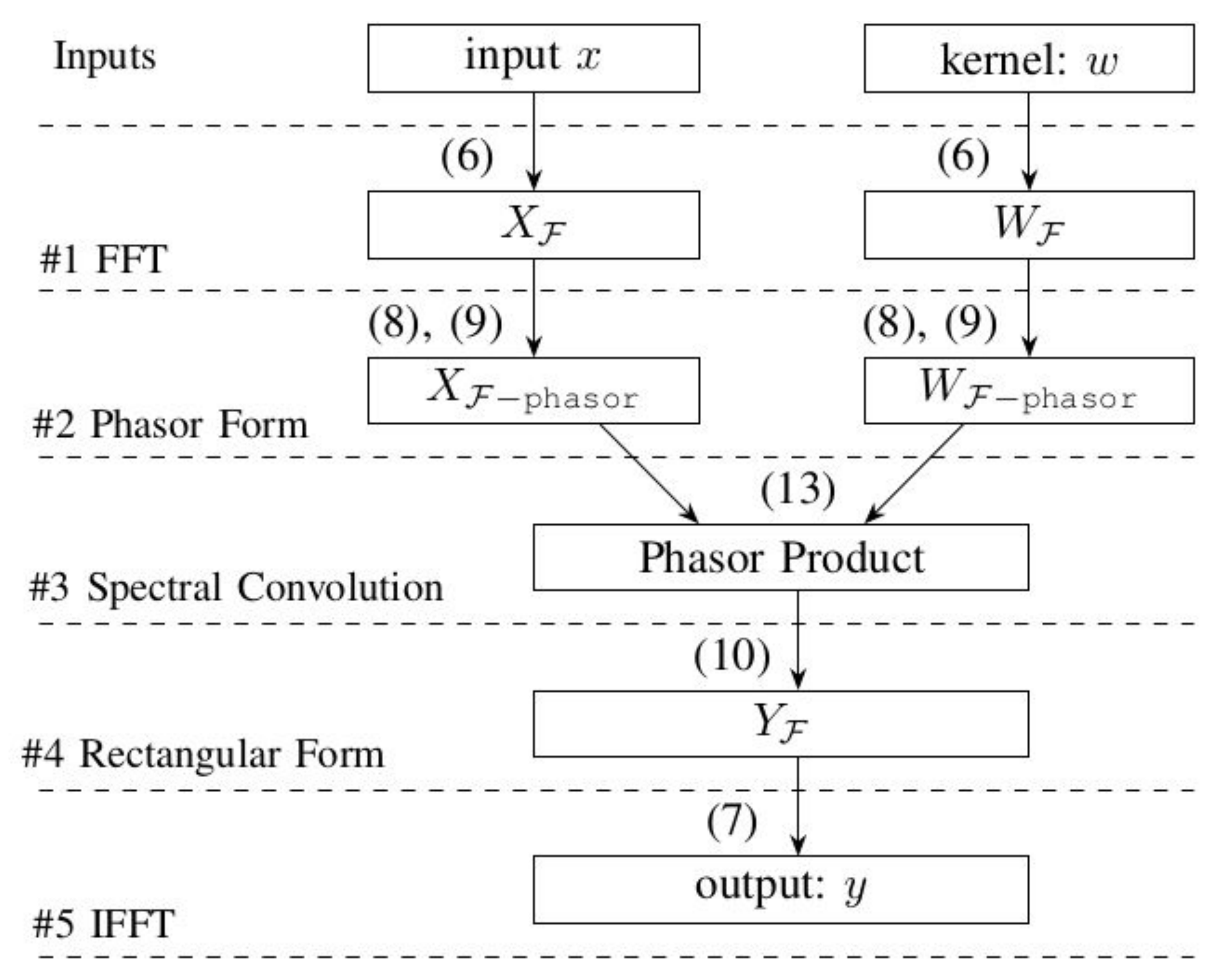


Figure 1. Overview of the proposed method using phasor product to reduce the number of operations between $X_{\mathcal{F}}$ and $W_{\mathcal{F}}$.

$$2CN^2 \log_2 N[Bf_1 + Bf_2 + f_2 f_1] + 4Bf_2 f_1 N^2, \tag{1}$$

$$y_{f_2} = \sum_{f_1} x_{f_1} \star w_{f_2 f_1}, \tag{2}$$

$$\frac{\partial L}{\partial x_{f_1}} = \frac{\partial L}{\partial y_{f_2}} * w^T_{f_2 f_1}, \tag{3}$$

$$\frac{\partial L}{\partial w_{f_2 f_1}} = \frac{\partial L}{\partial y_{f_2}} \star x_{f_1}, \tag{4}$$

$$e^{j\theta} = \cos(\theta) + j\sin(\theta) \tag{5}$$

$$X_{\mathcal{F}}[k] = \sum_{n=0}^{N-1} x[n] e^{-j2\pi kn/N} \tag{6}$$

$$x[n] = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} X_{\mathcal{F}}[k] e^{j2\pi kn/N} \tag{7}$$

$$|\mathbf{z}| = \sqrt{a^2 + b^2} \tag{8}$$

$$\phi = \tan^{-1}\left(\frac{b}{a}\right) \tag{9}$$

$$\mathbf{z} = |\mathbf{z}|\cos(\phi) + j|\mathbf{z}|\sin(\phi) \tag{10}$$

$$x[n] * w[n] = \mathcal{F}^{-1}\{X_{\mathcal{F}}[k] \cdot W_{\mathcal{F}}[k]\} \tag{11}$$

$$z_1 z_2 = (a_1 a_2 - b_1 b_2) + j(a_1 b_2 + a_2 b_1) \tag{12}$$

$$z_1 z_2 = |\mathbf{z}_1| \cdot |\mathbf{z}_2| \angle \phi_1 + \phi_2 \tag{13}$$

Experimental Results

Table 1: Batch Processing Time Analysis : Our method outperforms the baseline (based on [1]), for training on Cifar-10.

Architecture	Batch Size	Method	Total Time (sec)	Speedup (T_b/T_m)
VGG-16	4	Baseline	13.893	1.000
VGG-16	4	Our Method	11.019	1.261
DenseNet-121	8	Baseline	17.876	1.000
DenseNet-121	8	Our Method	13.476	1.326
EfficientNetB3	16	Baseline	20.337	1.000
EfficientNetB3	16	Our Method	14.967	1.359
Inception-V3	16	Baseline	40.967	1.000
Inception-V3	16	Our Method	29.222	1.402
AlexNet	64	Baseline	5.978	1.000
AlexNet	64	Our Method	4.433	1.349
ResNet-18	64	Baseline	19.676	1.000
ResNet-18	64	Our Method	14.310	1.375

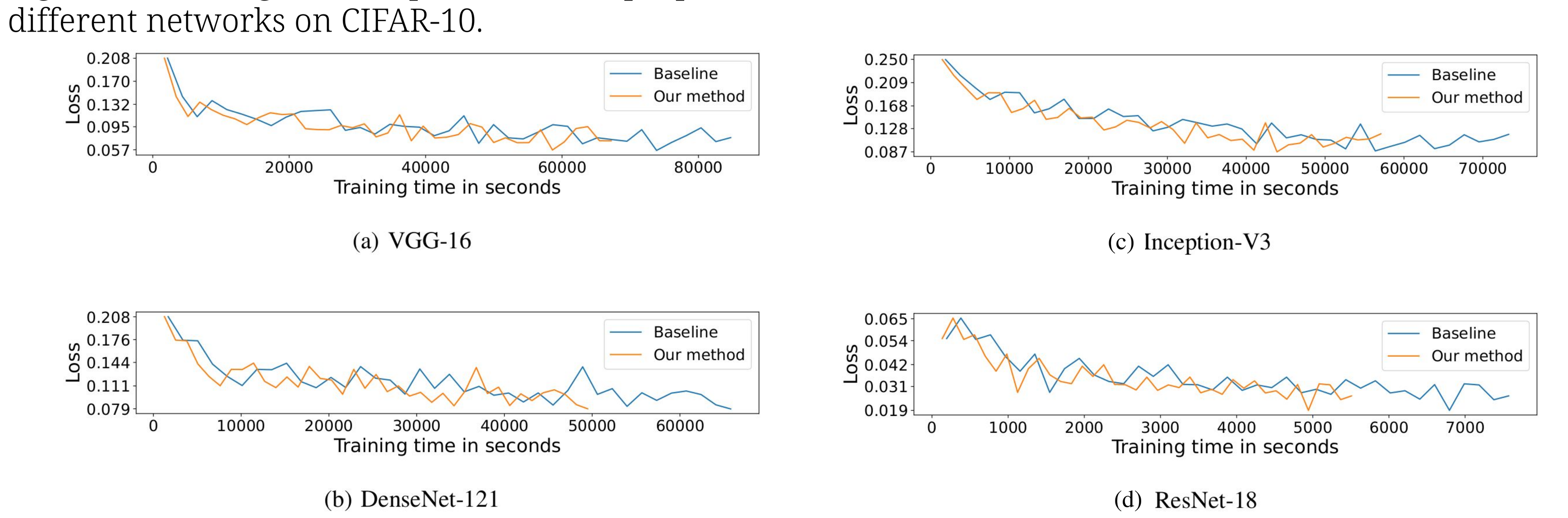
Table 2: Batch Processing Time Analysis : Our method outperforms the baseline (based on [1]), for training on Cifar-100.

Architecture	Batch Size	Method	Total Time (sec)	Speedup (T_b/T_m)
VGG-16	4	Baseline	13.904	1.000
VGG-16	4	Our Method	11.027	1.261
DenseNet-121	4	Baseline	9.809	1.000
DenseNet-121	4	Our Method	7.946	1.234
EfficientNetB3	8	Baseline	10.856	1.000
EfficientNetB3	8	Our Method	8.394	1.293
Inception-V3	8	Baseline	22.427	1.000
Inception-V3	8	Our Method	17.320	1.295
AlexNet	64	Baseline	5.922	1.000
AlexNet	64	Our Method	4.409	1.343
ResNet-18	64	Baseline	19.615	1.000
ResNet-18	64	Our Method	14.303	1.371

Table 3: Time Analysis: Our method outperforms the baseline (based on [1]) in training , with an average speedup of 1.316×, and inference , with an average speedup of 1.321, on Cifar-10.

Architecture	Batch Size	Method	Training Loss	Training Accuracy	Validation Loss	Validation Accuracy	Duration Training (sec)	Duration Validation (sec)	Training Speedup (T_b/T_m)	Validation Speedup (T_b/T_m)
VGG-16	4	Baseline	0.07705	97.125	0.23592	93.360	87371	6409	1.000	1.000
VGG-16	4	Our Method	0.07173	97.764	0.23491	93.640	69441	5087	1.258	1.260
DenseNet-121	8	Baseline	0.07874	97.691	0.10685	96.430	67381	4804	1.000	1.000
DenseNet-121	8	Our Method	0.07890	98.010	0.10725	96.390	50714	3589	1.329	1.338
EfficientNetB3	8	Baseline	0.12404	96.099	0.07638	97.530	76117	4828	1.000	1.000
EfficientNetB3	8	Our Method	0.12398	96.099	0.07632	97.540	59200	3777	1.286	1.278
Inception-V3	8	Baseline	0.11788	96.099	0.12478	95.960	74980	4469	1.000	1.000
Inception-V3	8	Our Method	0.11881	95.860	0.12713	95.880	58435	3441	1.283	1.299
AlexNet	64	Baseline	0.02280	99.297	0.32768	90.640	2270	157	1.000	1.000
AlexNet	64	Our Method	0.02271	99.297	0.32763	90.610	1666	115	1.363	1.362
ResNet-18	64	Baseline	0.02630	99.219	0.14690	95.050	7612	523	1.000	1.000
ResNet-18	64	Our Method	0.02627	99.297	0.14701	95.050	5534	376	1.376	1.390

Figure 2. Training loss comparison of the proposed model w.r.t. the baseline, based on [1], for four different networks on CIFAR-10.



(a) VGG-16
(c) Inception-V3
(b) DenseNet-121
(d) ResNet-18

References

[1] M. Mathieu, M. Henaff, and Y. LeCun, "Fast training of convolutional networks through ffts," in 2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings, 2014.
[2] N. Vasilache, J. Johnson, M. Mathieu, S. Chintala, S. Piantino, and Y. LeCun, "Fast convolutional nets with fbfft: A GPU performance evaluation," in 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings, 2015.
[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States, 2012, pp. 1106–1114.
[4] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings, Y. Bengio and Y. LeCun, Eds., 2015.
[5] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016, 2016, pp. 2818–2826.
[6] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016, 2016, pp. 770–778.
[7] Q. Zhang, M. Zhang, T. Chen, Z. Sun, Y. Ma, and B. Yu, "Recent advances in convolutional neural network acceleration," Neurocomputing, vol. 323, pp. 37–51, 2019.

21st Conference on Robotics and Vision (CRV 2024). Workshop Track Poster. University of Guelph. 50 Stone Road East, Guelph, ON. N1G 2W1

May 27th, 2024.