Hardware-Oriented Approximation of Convolutional Neural Networks (CNNs)

Philipp Gysel, Mohammad Motamedi and Soheil Ghiasi

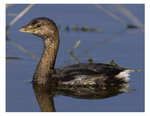
Department of Electrical and Computer Engineering, UC Davis

1. Convolutional Neural Networks

Deep neural networks have achieved state-of-art performance in many pattern recognition tasks. A particular type known as convolutional neural network (CNN) has proven to be especially suitable for computer vision. Samples of recently proposed applications for CNNs are shown below [1, 2].

Generating Visual Explanations:

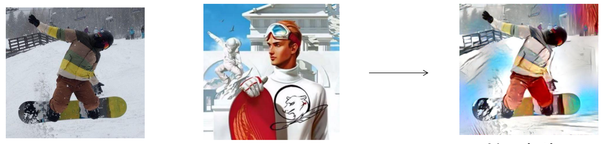
This is a **Kentucky warbler** because this is a **yellow bird** with a **black cheek patch** and a **black crown**.



This is a **pied billed grebe** because this is a **brown bird** with a **long neck** and a **large beak**.

AI Painter:

Paint image in the style of another one.



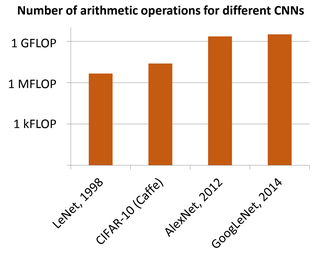
2. Resource Requirements

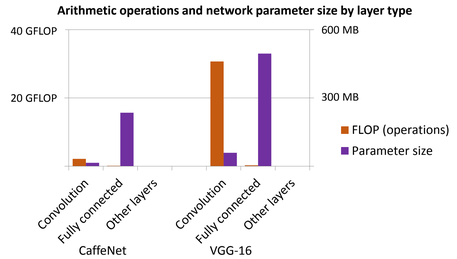
Computation:

The convolutional layers contain typically 90% of the required arithmetic operations. Inference of one image requires over 2 GFLOP and 30 GFLOP for CaffeNet and VGG-16, respectively.

Network size:

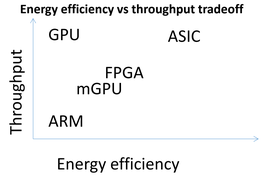
Fully connected layers contain over 90% of the network parameters. Using single precision format, the parameter size of CaffeNet and VGG-16 are 132MB and 500MB, respectively.

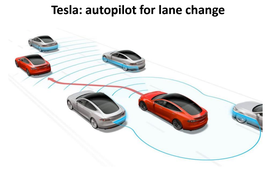




3. Accelerator Platforms

Artificial intelligence is becoming increasingly popular in mobile devices such as Google glasses, smartphones and cars. Fast and energy efficient image processing algorithms require dedicated hardware accelerators. Platforms such as mobile GPU, FPGA and ASIC allow for custom implementations, such as reduced-precision arithmetic. The first crucial step in hardware accelerator design is model condensation. Standard training of convolutional neural networks (CNN) uses single precision for forward propagation. However, recent research proves CNNs perform well using limited numerical precision.





4. Ristretto: Approximation Framework for CNNs

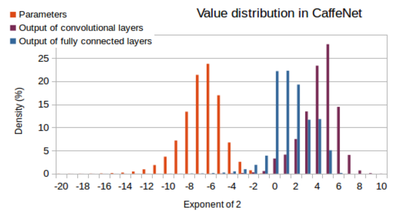
Ristretto

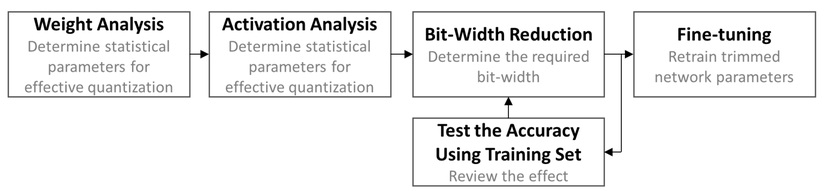
We present Ristretto, a Caffe [3] based approximation framework for CNNs with the following strengths:

- **Automation**: Ristretto automatically finds the optimal bit-width to represent layer outputs and parameters of convolutional and fully connected layers.
- **Flexibility**: Different quantization strategies are supported: fixed point, dynamic fixed point, mini floating-point, multiplier-free arithmetic.
- **Accuracy**: Ristretto fine-tunes trimmed networks to achieve high network accuracy.
- **Speed**: Network training and benchmarking runs on the GPU, using highly optimized CUDA routines.

Quantization flow

Ristretto analysis the dynamic range of values in the CNN. Based on statistical parameters, an appropriate reduced-precision format is chosen. In a next step Ristretto performs a binary search to find the optimal bit-width.

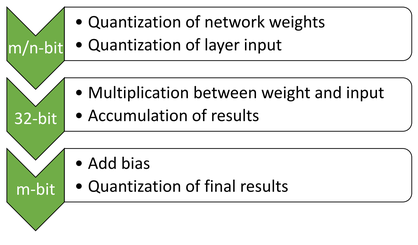


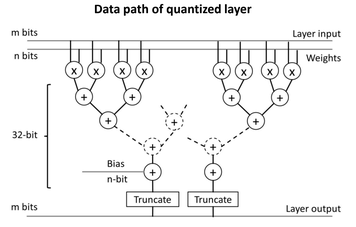


5. Approximation Strategies

Data path

Ristretto simulates the arithmetic of a hardware accelerator which uses reduced precision format for network parameters and layer outputs. The data path of convolutional and fully connected layers consists of a series of mutliplication-accumulation operations.





Approximation schemes

Dynamic fixed-point:

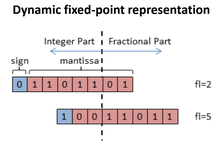
Fixed point numbers are represented with a fix number of integer bits and fractional bits. Since the numbers in a CNN cover a wide dynamic range, it is desirable to use a dynamic format which adapts the fractional length (fl) for layer outputs and parameters separately.

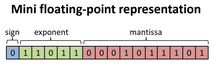
Mini floating-point:

Ristretto supports mini floating-point numbers which follow the IEEE-754 standard to a large degree. The number of exponent and mantissa bits can be arbitrary.

Multiplier-free arithmetic:

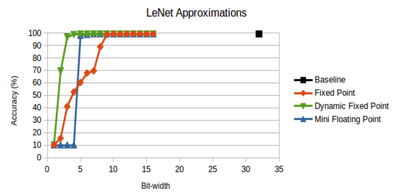
All parameters are integer power of two numbers, and thus multiplications turn into bit-shifts.



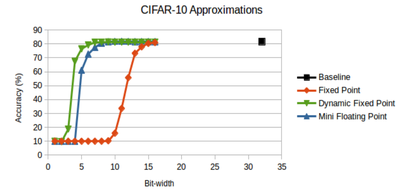


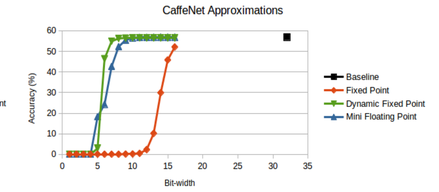
6. Results

We use Ristretto to approximate 32-bit networks. We consider five baseline networks: LeNet, CIFAR-10, CaffeNet, GoogLeNet and SqueezeNet. The quantization of parameters and layer outputs to limited precision format results in the following bit-width – accuracy tradeoff:



Network	Definition	Data set
LeNet	Network definition by Caffe, network by LeCun et al.	Handwritten digits
CIFAR-10	Caffe-network (Full CIFAR-10)	10 image classes such as airplane, bird, truck
CaffeNet	Caffe version of AlexNet	1000 image classes such as bird and plant species
GoogLeNet & SqueezeNet	Caffe Modelzoo	





Below are the attained classification accuracies after fine-tuning with Ristretto:

Network	Baseline accuracy (32-bit)	Fixed point	Dynamic fixed-point	Mini floating-point	Multiplier-free arithmetic
LeNet	99.15%	98.88% (8-bit)	98.81% (4/2-bit)	99.20% (8-bit)	99.16%
CIFAR-10	81.69%	81.38% (16-bit)	81.44% (8-bit)	80.85% (8-bit)	77.48%
CaffeNet top-1	56.90%	52.48% (16-bit)	56.00% (8-bit)	52.52% (8-bit)	53.25%
GoogLeNet top-1	68.92%	Not available	66.57% (8-bit)	Not available	Not available
SqueezeNet top-1	57.68%	yet	57.09% (8-bit)	yet	yet

Relation to previous work:

This work	Related work
Ristretto approximates existing neural network architectures. We don't increase the number of parameters.	**BinaryConnect**: New architectures with binary parameters.
Approximation without added complexity such as sparse computation or decompression.	**Deep compression pipeline**: Remove unnecessary parameters, and use shared parameter values.

7. Conclusion

We present Ristretto, an approximation framework for convolutional neural networks (CNNs). Results indicate large CNNs can be represented with 8-bit dynamic fixed-point and 8-bit mini floating-point. We compress the parameters and layer outputs of convolutional and fully connected layers without adding additional complexity such as decompression or sparse computation. AlexNet and SqueezeNet were successfully compressed to 8-bit dynamic fixed-point with an absolute accuracy drop below 1%.

Ristretto is available on Github: https://github.com/pmgysel/caffe
Ristretto documentation: http://ristretto.lepsucd.com/

8. References

[1] Hendricks, Lisa Anne, et al. "Generating Visual Explanations." *arXiv preprint arXiv:1603.08507* (2016).
[2] Gatys, Leon A., Alexander S. Ecker, and Matthias Bethge. "A neural algorithm of artistic style." *arXiv preprint arXiv:1508.06576* (2015) & https://www.instapainting.com/ai-painter
[3] Jia, Yangqing, et al. "Caffe: Convolutional architecture for fast feature embedding." *Proceedings of the ACM International Conference on Multimedia*. ACM, 2014.
Picture credit: teslamotors.com