High-speed Training Using Binary Neural Networks

Project Mentor: Dr. Richard Martin
Our Team

Sachin Matthew '22  Daniel Maevsky '24  Daniel Chen '24  Tommy Forzani '24  Serena Zhang HS
Goals

● Training machine learning systems is currently very slow
  ○ Floating point chips take ~1 million transistors

● Recent work has shown promise by using simpler representations of numbers than the commonly used floating point ones.
  ○ Integer chips take ~300 thousand transistors
  ○ **Uses less power and have simpler arithmetic than floating point**

● Our goal: create and measure neural networks which only use binary or fixed-point numbers for both training and inference
Floating Point vs Fixed Point

Floating Point Numbers
- Current standard for ML and other computer applications
- Think scientific notation e.g. $4.5 \times 10^6$
- Extremely precise with ability to store large range of numbers
- Contains sign, exponent and mantissa which needs to be normalized
- Uses ~1,000,000 transistors

Fixed Point Numbers
- Regular decimal number, contains an integer part (left of decimal point) and a fraction part (right of decimal point)
- Only uses ~300,000 transistors - much more efficient
- Limited range of numbers
  - A consideration we must make and test
Activation Functions - Sigmoid vs ReLU

Activation function: Helps network learn patterns; decides what to fire to next neuron

**Sigmoid**
- \( f(x) = \frac{1}{1 + e^{-x}} \)
- More complex but less efficient
- Outputs are constrained from 0 to 1

**ReLU**
- \( y = \max(0, x) \)
- Less complex, more efficient
- Outputs approach infinity, leading to poor accuracy
- Requires another layer, like Softmax, to function accurately
Datasets - MNIST Digits and Fashion

MNIST Digits
- Handwritten digits 0-9
- 28x28 grayscale image
- Easy to incorporate & train
- Highly Implemented with near perfect accuracy

MNIST Fashion
- Articles of clothing Ex. Sneakers, shirts, dresses, etc.
- 28x28 grayscale image
- Easy to incorporate & difficult to train
- More applicable for CV tasks
Method - Weeks 1-5

- Adapt the GoNN github repository for project (linear algebra library)
- Plot maxima and minima of the floating point weights to get dynamic range
- Check for accuracy plateau -> Lower bound for working range
- Implement fixed point matrix library (64 bit representation, sign bit, 15 bits preceding point, 48 bits following)
Method - Weeks 6-9

- Adapt activation function for fixed point representation (sigmoid)
- Analyze effects of number truncation (reduction of precision) on accuracy
- Begin Implementation of ReLU activation
- Apply fixed point schema to MNIST Digits and Fashion databases
- Collect and analyze accuracy and range data for fixed point models vs floating point
Issues

- Range does not converge, must find workable range for fixed point representation
- ReLU requires addition of SoftMax layer to function
- Sigmoid function within range overflows 64 bit fixed point representation, must be altered to function
Results

- Accuracy drop of a maximum 3% compared to the floating-point network
- Dynamic range of 10 is very small, and much precision is not really necessary for accuracy. This is promising
Further Research

- Apply fixed point schema to further databases
- Adapt code to wider variety of neural networks (modular layer sizes and numbers, new layer types like convolutional layers)
- Further testing of our ReLU implementation for efficacy vs Sigmoid
Thank You!
Any Questions?