Neural network retraining for overlapping weights encoded in a transpose pattern

Contact Person: Mr. William Simon (william.simon@epfl.ch), Dr. Alexandre Levisse (alexandre.levisse@epfl.ch), Prof. David Atienza (david.atienza@epfl.ch)

Project Description

Over the last decade, neural networks have been applied to a wide range of applications with great success. However, the weights they store result in a large memory overhead at all levels of the memory hierarchy \[1\]. Much work has been done to mitigate the drawbacks of this characteristic of neural networks.

The goal of this project is to utilize a new transposable access RRAM memory to increase the storage capacity allocated to weights, reducing the necessity for memory movement.

Figure 1 presents a high-level view of the functionality of the transpose access memory array. At each memory access, each subarray can be accessed in either a regular or transposed manner, enabling new data access patterns that can accelerate application runtime.

In order to take advantage of such an architecture for the acceleration of neural networks, we propose to encode neural network weights such that they overlap in a regular/transpose manner, effectively doubling the capacity of the memory array at the cost of a loss of weight accuracy (not necessarily neural network accuracy however). Weights will be stored such that the MSBs of weights stored in regular fashion overlap with the LSBs of weights stored in transpose fashion. Such a configuration reduces the error induced by bit sharing. Individual bit values can then be modified to minimize the error of the final weights with respect to their original values. It has already been demonstrated empirically that it is possible to reduce weight error to an average of 0.6% and a max of 1.44% for an overlapping array of 32 16-bit random integer values arranged in such a configuration. This adjustment of weight values, combined with network retraining, for example through incremental retraining\[2\], may enable more compact weight storage at little to no loss of neural network accuracy. Figure 2 illustrates the storage strategy.
The concept presented here can be expanded upon in multiple directions. The primary objective will be implementing an algorithm for adjusting weight values to minimize individual weight error while retraining the network to maintain accuracy. Network retraining can be performed in a variety of ways, with the suggested starting point being the strategy outlined in [2]. Other objectives may include an upper bound on weight error, introducing the ability to shift weight values to provide another degree of freedom when encoding, and considering the expansion of the concept into the 3rd dimension (aka a 3d memory stack) to further increase capacity.

**Tasks of the Student:**
1. Exploration of current quantization methods.
2. Develop encoding/quantization strategy for storing weights at minimal error within Matlab.
3. Implement algorithm for modifying weights and retraining network within a neural network framework. (Tensorflow, Caffe, etc.)
4. If previous objectives are fulfilled, this work will be a part of a submission in a peer-reviewed conference or journal.
**Requirements:**
- Understanding of neural network theory, both training and inference.
- Previous work with Matlab, Python or C++.

**Appreciated Skills:**
- Previous work with Tensorflow or equivalent neural network framework.
- Motivation to learn new skills.
- Autonomous work ability.

**References:**
[2] Incremental Network Quantization: Towards Lossless CNNs with Low-Precision Weights