Optimizing Training Precision: Balancing Time and Accuracy

CPSC 440 Project Proposal — Romina Mahinpei

1 Motivating Problem

Most numerical algorithms from the scientific computing community are implemented in double precision, corresponding to a 64-bit floating point number format. Although double precision has its benefits, there has been recent interest in incorporating lower precision arithmetic, such as single, half, and even quarter precision, into numerical algorithms [1]. The majority of this interest originates from the speed, minimal energy usage, and reduced communication costs that lower precision offers relative to its higher precision counterpart [1]. To strike a balance between the speed of low precision arithmetic and the accuracy of high precision arithmetic, some studies have started to explore *multi-precision algorithms*, which allow an implementation to be executed in a user-specified precision out of multiple possible options, as well as *mixed precision algorithms*, which use two or more precisions within the same execution of an implementation [1].

As with the scientific computing community, there have been similar efforts in using low precision arithmetic within the machine learning community [2, 3, 4]. Most of these efforts have focused on integrating low precision into the training of deep neural networks. As an example, the work in [2] uses 8- and 4-bit networks for classification and shows that the inference is sufficiently close to the 32-bit baseline network. Similarly, the work in [4] explores 4-bit networks for deep learning models in computer vision, speech, and NLP and highlights a non-significant loss in accuracy. Although these efforts seem promising, there has been no evaluation (to the best of our knowledge) on whether low precision training can be used as a *general-purpose method* for speeding up the training of ML models without significantly reducing the model's accuracy. As such, this project aims to fill this gap by investigating the impact of different training precisions on the final attained training time and accuracy across a broad range of models and applications.

2 Proposed Solution

Since the goal of this project is to understand how the precision of training affects the total training time and the accuracy on a held-out validation set, the following steps will be taken:

1. Build multi-precision implementations of the chosen models;
2. Collect training time measurements for each model variant on a GPU;
3. Evaluate the accuracy of each model variant on a held-out validation set.

Each multi-precision implementation will support model training in either double, single or half precision, thus resulting in three different variants for each of our chosen models. As of right now, the plan is to develop multi-precision implementations for the following models:

1. Decision tree model (to be trained on scikit-learn's Iris dataset);
2. Linear regerssion model (to be trained on scikit-learn's diabetes dataset);
3. SVM classifier (to be trained on scikit-learn's Iris dataset);
4. Standard neural network (to be trained on TensorFlow's Fashion MNIST dataset);
5. Convolutional neural network (to be trained on the CIFAR10 dataset).

Once the multi-precision implementations are developed, timing and accuracy measurements will be made on an NVIDIA T4 GPU for all three variants of each of our five models (thus resulting in 15 sets of measurements).

3 Expected Contributions

The expected contributions of this project are as follows:

- **Multi-precision implementations of five different models**: Most implementations currently support training in *only one precision*, but our work will enable the user to specify the training precision as either half, single, or double.
- **An evaluation on the impact of training precision on training time and accuracy**: To the best of our knowledge, no investigation has yet been done on whether low precision training can be used as a general-purpose method for speeding up the training of ML models without significantly reducing the model's accuracy. This evaluation will enable us to address this gap for a range of models and applications.

References

[1] Nicholas J. Higham and Theo Mary. "Mixed precision algorithms in numerical linear algebra". In: *Acta Numerica* 31 (2022), pp. 347–414. DOI: 10.1017/S0962492922000022.

[2] Jeffrey L. McKinstry et al. "Discovering Low-Precision Networks Close to Full-Precision Networks for Efficient Inference". In: *2019 Fifth Workshop on Energy Efficient Machine Learning and Cognitive Computing - NeurIPS Edition (EMC2-NIPS)*. 2019, pp. 6–9. DOI: 10.1109/EMC2-NIPS53020.2019.00009.

[3] Xiao Sun et al. "Hybrid 8-bit Floating Point (HFP8) Training and Inference for Deep Neural Networks". In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach et al. Vol. 32. Curran Associates, Inc., 2019. URL: https://proceedings.neurips.cc/paper_files/paper/2019/file/65fc9fb4897a89789352e211ca2d398f-Paper.pdf.

[4] Xiao Sun et al. "Ultra-Low Precision 4-bit Training of Deep Neural Networks". In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle et al. Vol. 33. Curran Associates, Inc., 2020, pp. 1796–1807. URL: https://proceedings.neurips.cc/paper_files/paper/2020/file/13b919438259814cd5be8cb45877d577-Paper.pdf.