Université de Rennes

Inria

# **Internship Offer** for 6 months:

## Limited-Precision Stochastic Rounding and its use in Gradient Descent-Based Optimization

**Filip Silviu-Ioan:** silviu.filip@inria.fr
**El Arar El-Mehdi:** el-mehdi.el-arar@inria.fr

## Main content

The *Deep Learning* (DL) boom of the last decade has enabled many state-of-the-art results in farreaching domains such as computer vision, natural language processing, and weather prediction, just to name a few. It has also galvanized many research communities to tackle the major hurdles that make the wide adoption of DL models a challenge. In the hardware space, this is exemplified by the massive memory and compute requirements that state-of-the-art neural networks demand, with many large language models, for instance, passing the billion and tens of billions parameter range, necessitating up to hundreds of GB in memory and storage. The main response has been a reduction in the size of the data formats used in such contexts, the improvement in terms of resource consumption usually being linearly proportional with the decrease in format size.

Whereas classic scientific computing applications have relied on IEEE-754 arithmetic with a combination of 32 and 64-bit formats (i.e., float and double in C/C++ parlance) available on mainstream CPUs, modern DL accelerators are increasingly moving towards 16-bit (binary16 and bfloat16) and 8-bit number formats (with the OCP and IEEE P3109 standards). While such smaller formats are better from a speed, efficiency, and memory standpoint, they are much more prone to computational rounding errors due to their much-reduced precisions when compared to their 32 and 64-bit big siblings. There have been various different strategies used to cope with this decrease in precision, but one in particular has shown significant promise for its effectiveness: *stochastic rounding* (SR). It consists of rounding a real value $x$ to one of the two closest representable values in the target format with probabilities that are proportional to the distances to the other candidate, meaning that $x$ will be rounded with a higher probability to the representable value that is closest to it. This is in stark contrast with other rounding modes such as round to nearest (RN), the default rounding mode for floating-point arithmetic. RN always picks, in a deterministic way, the representable value closest to $x$. Let us take the following toy example, with $x = 1.68$. Keeping just two significant digits in the representable result, RN will always result in the value $RN(x) = 1.7$. With SR, $SR(x) = 1.7$ with probability 0.8, and $SR(x) = 1.6$ with probability 0.2.

One can easily show that the expected value of the result of stochastically rounding a value $x$ is $x$ itself (indeed, in our previous example, the expected value is $0.8 \times 1.7 + 0.2 \times 1.6 = 1.68$), so the expected error is zero. Hence, SR maintains, in a statistical sense, some of the information that is discarded by a deterministic rounding scheme (such as those specified in the IEEE-754 standard for floating-point arithmetic). This proves quite beneficial in many situations, in particular the kind of computations that are common when training DL models.

While hardware support for SR is starting to appear (e.g., Graphcore IPUs and AMD MI300 GPUs), a major hurdle in its wide adoption is that it is not as straightforward to implement as other directed, rounding modes, incurring an overhead. Recent work has looked at optimizing its hardware implementation [3] and also at the theoretical consequences of using limited precision in such implementations when an ideal operator is not feasible.

Our goal with this internship is to expand on this analysis and study the impact of limited-precision SR during gradient descent-based optimization. Such an analysis of limited-precision SR in concrete algorithmic situations is important for accurately assessing and validating the impact SR stands to have in the future of numerical computing, both in the DL setting and at large.

# Objectifs

Concretely, this will entail using the error model introduced in [4] and applying it to the context of the gradient descent algorithm in a similar way to [6, 5]. Numerical simulations will be carried out using the mptorch [1] and srfloat [2] libraries developed within the TARAN team, using example problems from both classical nonlinear optimization and *deep learning* scenarios. In the first step, the student will be required to familiarize himself with the basics of floating-point arithmetic and the convergence of gradient descent-based algorithms. Depending on the student and her/his progress, she/he will proceed with the analysis proper, which can be carried out at various levels of granularity, focusing on just specific parts of the overall algorithm: (1) the parameter update part of the computation and/or (2) the computations involved in determining the gradient of the function to optimize.

# References


[1] https://github.com/mptorch/mptorch.

[2] https://github.com/sfilip/srfloat.

[3] S. B. Ali, S.-I. Filip, and O. Sentieys, *A stochastic rounding-enabled low-precision floating-point mac for dnn training*, in 2024 Design, Automation & Test in Europe Conference & Exhibition (DATE), 2024.

[4] E.-M. E. Arar, M. Fasi, S.-I. Filip, and M. Mikaitis, *Probabilistic error analysis of limited-precision stochastic rounding*, 2024.

[5] L. Xia, M. E. Hochstenbach, and S. Massei, *On the convergence of the gradient descent method with stochastic fixed-point rounding errors under the polyak-lojasiewicz inequality*, arXiv preprint arXiv:2301.09511, (2023).

[6] L. Xia, S. Massei, M. E. Hochstenbach, and B. Koren, *On the influence of stochastic roundoff errors and their bias on the convergence of the gradient descent method with low-precision floating-point computation*, 2023.