Stochastic Rounding for LLM Training: Theory and Practice

Kaan Ozkara[1,2], Tao Yu[1], Youngsuk Park[1]
[1]Amazon Web Services, [2]UCLA

amazon | science

TLDR

- We propose **BF16+SR strategy** for **pre-training** of LLMs that keeps every tensor including model weights, optimizer states, and gradients in BF16 precision.
- Empirical results from pre-training models with up to **6.7B parameters** shows that compared to state of the art (BF16-FP32) mixed precision (MP) strategies, our method achieves **better validation perplexity**, up to **1.54× higher throughput**, and **30% less memory usage**.
- The key techniques that we employed to enable full BF16 training is to **share randomness** for model updates, and using a **higher learning rate** while training with SR.
- Theoretically, we show how SR results in an **implicit regularization** and quantization aware training. We find that training with SR results in **better convergence** properties in terms of error induced by quantization compared to MP training.
- Our theoretical results emphasize the importance of employing a high learning rate for a successful SR training, and SR training is **robust to training with high learning rates**.

Rounding Modes

Nearest Rounding (NR)

$$Q(x) := \mathrm{sign}(x)\cdot\Delta_x\cdot\left\lfloor \frac{x}{\Delta_x}+\frac{1}{2}\right\rfloor,$$

- Biased estimate
- Optimal worst case error
- Default rounding mode in FP operations

Stochastic Rounding (SR)

$$Q_{SR}(x) := \begin{cases} \lceil x\rceil, & \text{w.p. } \frac{x-\lfloor x\rfloor}{\Delta_x} \\ \lfloor x\rfloor, & \text{w.p. } \frac{\lceil x\rceil - x}{\Delta_x} \end{cases}.$$

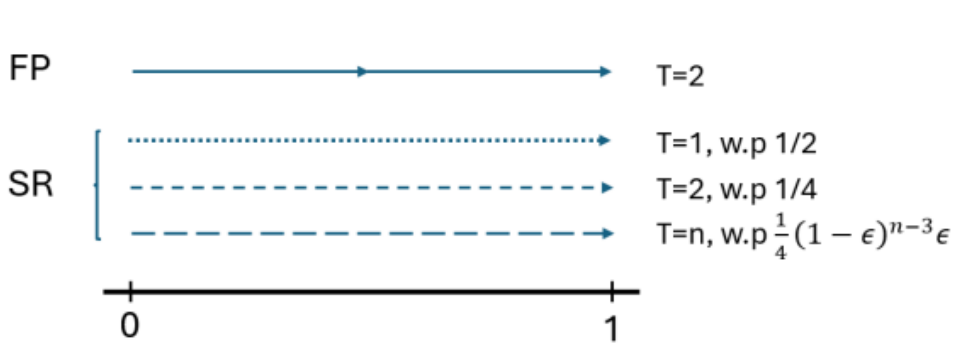
- Unbiased
- Lower error accumulation with consecutive operations
- Lower probabilistic error bounds

SR gives true update in expectation– prevents stagnation

Assume $x_{t+1} = x_t + u$, where $u \ll x_t$, $Q(x_t+u) = x_t$, $E[Q_{SR}(x_t+u)] = x_t+u$. Stagnation happens with deterministic rounding.

SR needs high learning rate

Previous works [1,2] also considered BF16 training with SR but reported MP training is superior in terms of ppl. For SR training, there is a risk of slow convergence when learning rate is not high enough. A toy example:



$$x_0 = 0, x_{target} = 1, u_1 = u_2 = \frac{1}{2}$$
$$u_{t>2} = \epsilon$$

Then

$T_{Fp} = 2, E[T_{SR}] > 2$ if $\epsilon < \frac{1}{2}$, $E[T_{SR}] \to \infty$ *when* $\epsilon \to 0$.

We will also see this effect in the upcoming theory results.

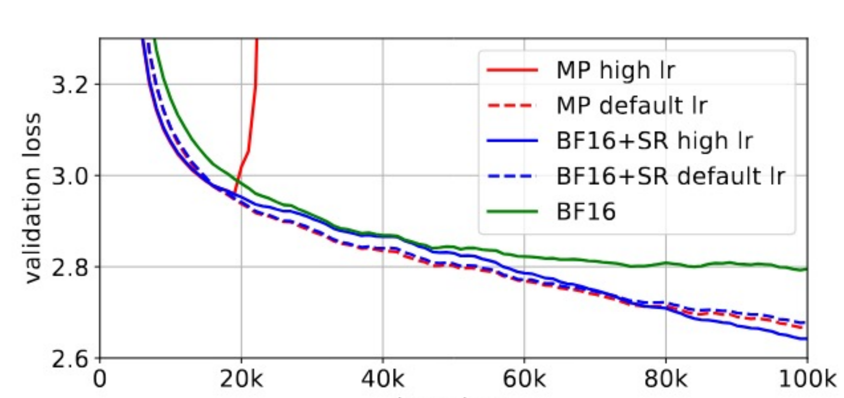
SR is robust to high learning rate

Correlation among consecutive gradients is considered as a culprit of divergence under high learning rates [3]. We can show that **stochasticity due to SR can reduce this correlation** resulting training more robust to high learning rates.

Our Contribution:

	BF16+SR	(BF16,FP32) MP	BF16
Accuracy	↑*	↑	↓
Throughput	↑	↑	↑
Memory eff.	↑	↓	↑
Robustness	↑*	↓	↓

* denotes first shown in this work.

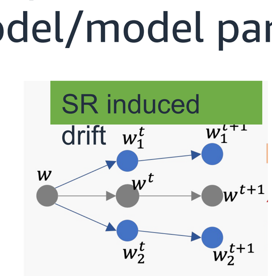


BF16 Adam with SR and shared randomness

BF16+SR vs MP

2: **for** $t = 1$ to T **do**
3: **for** $m = 1$ to M **do**
BF16 4: FP32 $g_t \leftarrow \nabla f(x_t)$ ## reduced gradient
BF16 5: FP32 $m_{t+1} \leftarrow \beta_1 \cdot m_t + (1-\beta_1)\cdot g_t$
BF16 6: FP32 $v_{t+1} \leftarrow \beta_2 \cdot v_t + (1-\beta_2)\cdot g_t^2$
7: $\hat{m}_{t+1} \leftarrow \frac{m_{t+1}}{1-(\beta_1)^t}$
8: $\hat{v}_{t+1} \leftarrow \sqrt{\frac{v_{t+1}}{1-(\beta_2)^t}}$
9: $r \leftarrow r_t^{(opt)}$ ## share the same random state across ranks for optimizer
BF16 10: FP32 $x_{t+1} \leftarrow Q_{SR}\left(x_t - \left(\alpha_t\cdot\frac{\hat{m}_{t+1}}{\sqrt{\hat{v}_{t+1}+\epsilon}} + \alpha_t\cdot\lambda\cdot x_t\right)\right)$
$r_t^{(opt)}$ is updated during SR
11: $r \leftarrow r_t^m$ ## get current random state

- In our BF16+SR strategy every variable is in BF16.
- For mixed precision every variable in FP32; only forward and backward operations are in BF16.
- Shared randomness for the optimizer that prevents drift of replicated model/model parts (line 10).



Implicit Regularization

$$F_{SR}(x) := \underbrace{F(x)}_{\text{loss function}} + \underbrace{\frac{\alpha}{4}\|\nabla F(x)\|^2}_{\text{Implicit reg due to GD}} + \underbrace{\boxed{\frac{\alpha}{4}\mathbb{E}[\|\xi_\alpha(x)\|^2]}}_{\text{Implicit reg due to SR}}$$

Quantization aware training

SR introduces quantization penalty and results in implicit quantization aware training

Convergence

Theorem 2 (No momentum, $\beta_1 = 0$). *Assuming access to full precision gradients, and learning rate $\alpha_t = \alpha\sqrt{\frac{1-\beta_2^t}{1-\beta_2}}$ with $\alpha > 0$, when SR is used for the update step to obtain quantized weights, we have*

$$G_T \le \frac{A}{T} + \frac{2Rd}{T}\left(\frac{2R}{\sqrt{1-\beta_2}} + \frac{\alpha L}{1-\beta_2}\right)\left(T\ln\left(\frac{1}{\beta_2}\right)\right)$$
$$+ \underbrace{\frac{Rd\Delta L}{T\sqrt{1-\beta_2}}\left(\sqrt{T\ln\left(1+\frac{R^2}{\epsilon(1-\beta_2)}\right)} + T\sqrt{\ln\left(\frac{1}{\beta_2}\right)}\right)}_{\text{quantization error}}$$

Here $\Delta = \max_i \Delta_{x_i}$, $G_T = \frac{1}{T}\sum_{t=1}^T \|\nabla F(x_t)\|^2$ and A is a constant that depends on d, R, β_2, α.

Theorem 3. *Under the same setting in Theorem 2, Adam with NR gives the following convergence bound*

$$G_T \le \frac{A}{T} + \frac{2Rd}{T}\left(\frac{2R}{\sqrt{1-\beta_2}} + \frac{\alpha L}{1-\beta_2}\right)\left(T\ln\left(\frac{1}{\beta_2}\right)\right)$$
$$+ \frac{2Rd\Delta L}{T\sqrt{1-\beta_2}}\left(\sqrt{T\ln\left(1+\frac{R^2}{\epsilon(1-\beta_2)}\right)} + T\sqrt{\ln\left(\frac{1}{\beta_2}\right)}\right)$$
$$+ \frac{\sqrt{1-\beta_2}d(R\Delta + L\Delta^2)}{\alpha}.$$

Due to biasedness, Nearest rounding results in a **larger error** bound due to (i) multiplicative term of ×2 in the second line in Theorem 3 and (ii) **additional error term in third line.**

Experiments

Baselines:
- O1 level mixed precision (torch.amp),
- O2 level mixed precision (Micikevicius et al., 2018),
- BF16 training (with default nearest rounding mode).

Models:
- GPT-2(350M, 770M) on ~50B tokens
- GPT-Neo(1.3B,2.7B,6.7B) on 40B,20Btokens

Learning rate tuned individually for each method

Table 2 Default, tuned for mixed precision, tuned for SR learning rates (×1e−4); and whether mixed precision training **converges with SR's learning rate**. Note SR converges in all cases.

Model size	Default	MP	SR	MP converges
350M	3	3	7	×
770M	2	2	7	×
1.3B	2	4	4	✓
2.7B	1.6	4	5	✓
6.7B	1.2	3.6	5.5	×

Validation PPL

Table 1 Validation perplexity of competing methods for GPT models.

Method	GPT-2(350M)	GPT-2(770M)	GPT-Neo(1.3B)	GPT-Neo(2.7B)	GPT-Neo(6.7B)
BF16+SR (ours)	**14.07**	**12.63**	**10.46**	**10.22**	**10.05**
(BF16,FP32) MP	14.45	12.83	10.48	10.31	10.11
BF16	16.38	15.28	11.81	11.67	11.64

Throughput

Table 3 Throughput (sequences/second) of SR compared to O1 level mixed precision training (`torch.amp`).

Method	GPT-2(350M)	GPT-2(770M)	GPT-Neo(1.3B)	GPT-Neo(2.7B)	GPT-Neo(6.7B)
BF16+SR (ours)	501	254	125	84	43
(BF16,FP32) MP	469	224	105	57	28
Gain	1.07×	1.14×	1.19×	1.47×	1.54×

Table 5 Throughput of SR compared to O2 level mixed precision training evaluated in Megatron-LM.

Method \GPT-Neo size	1.3B	2.7B	6.7B
BF16+SR (ours)	135	75	32
(BF16,FP32) MP	129	71	31
Gain	1.05×	1.06×	1.03×

Memory

Table 4 Memory consumption (GB) per node; SR compared to O1 level mixed precision training (`torch.amp`).

Method	GPT-2(350M)	GPT-2(770M)	GPT-Neo(1.3B)	GPT-Neo(2.7B)	GPT-Neo(6.7B)
BF16+SR (ours)	186	208	184	171	184
(BF16,FP32) MP	234	298	206	197	232
Tensor parallelism	1	1	8	8	8
Memory savings	21%	30%	11%	13%	21%

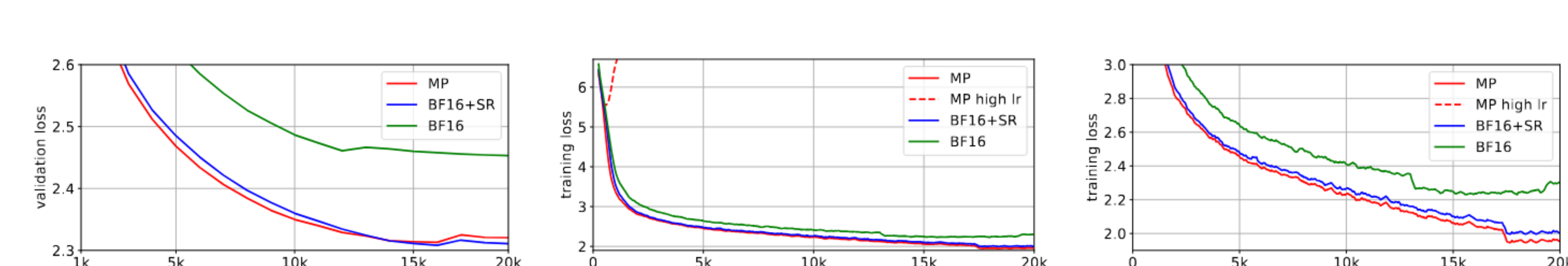


Figure 3: Validation (left) and training losses (middle, right) for training GPT-Neo (6.7B).

SR results in better validation perplexity despite lower wall clock time and memory usage.

*Corresponding author: pyoungsu@amazon.com