大規模言語モデルの再パラメタ化に基づく初期化による損失スパイクの抑制

西田光甫　西田京介　齋藤邦子
日本電信電話株式会社　NTT 人間情報研究所
{kosuke.nishida, kyosuke.nishida, kuniko.saito}@ntt.com

概要

大規模言語モデル (LLM) の事前学習中に損失関数値が突然発散してしまう損失スパイクが LLM 事前学習の課題である．本研究は，パラメータのノルムに対するパラメータ更新量のノルムの相対値であるパラメータの更新比率がモデル内で不均一であり，この不均一性が損失スパイクの一因であることを指摘した．本研究は，全てのパラメータにゲートパラメータを導入し，共通の標準偏差によって初期化することを提案した．提案手法が，13B モデルの LLM の事前学習においてモデル内の更新比率を均一化し，損失スパイクを抑制することを確認した．

1　はじめに

大規模言語モデル（LLM）の事前学習では，損失関数値が突然発散してしまう損失スパイク現象が知られている [1, 2]．損失スパイクは，小さな学習率を設定する必要が生じて LLM の性能が劣化する，時に完全に発散し学習を失敗させる，といった点から，LLM 事前学習の大きな課題である．

本研究では，損失スパイクを説明するため，パラメータの**更新比率** $\|\Delta \boldsymbol{W}\|/\|\boldsymbol{W}\|$ を定義する．ここで，$\boldsymbol{W}$ はパラメータ，$\Delta \boldsymbol{W}$ は 1 時刻における更新量である．更新比率はパラメータが相対的にどの程度激しく更新されるかを示す．ある $\boldsymbol{W}_i$ の更新比率が極端に大きい場合，$\boldsymbol{W}_i$ で更新が安定するように学習率を小さい値に設定するか，他のパラメータの更新を優先して $\boldsymbol{W}_i$ の急激な更新を許容するか，が必要になる．よって，更新比率はモデル全体である程度の範囲に収まることが好ましい．

しかし，LLM の事前学習において更新比率はパラメータによって大きく異なる．図 1 に学習初期の損失関数値，最終 down-projection 層の重み $\boldsymbol{W}_d$ と up-projection 層の重み $\boldsymbol{W}_u$ の更新比率を示す．学習

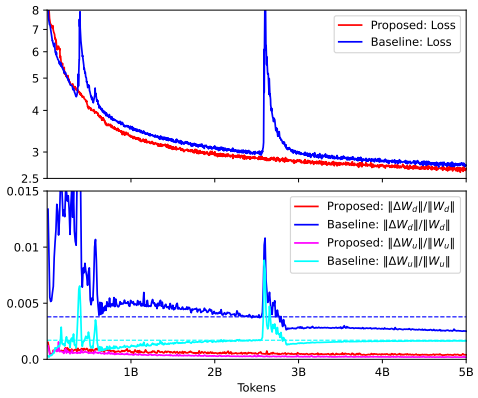

図 1　13B モデルの学習初期の損失関数値（上）と最終層の $\boldsymbol{W}_d, \boldsymbol{W}_u$ における更新比率（下）．

の最初期は $\boldsymbol{W}_d$ の更新比率が極端に大きいこと，小規模なスパイクを起こしながら $\boldsymbol{W}_d$ の更新比率が小さくなっていくこと，最後のスパイクの前後で $\boldsymbol{W}_d$ の更新比率の値が大きく減少し，$\boldsymbol{W}_d, \boldsymbol{W}_u$ の更新比率がある程度の範囲に収まることが確認できる．この観察から，更新比率の不安定性・不均一性と損失スパイクとの関連性が示唆される．

更新比率の不均一性に対処するため，LLM の初期化手法Weight Scaling as Reparameterization (WeSaR) を提案する．WeSaR は，各パラメータ行列 $\boldsymbol{W}_i$ をゲートパラメータ $\alpha_i \in \mathbb{R}$ によって $\bar{\boldsymbol{W}}_i = \alpha_i \boldsymbol{W}_i$ と再パラメタ化し，全てのパラメータ行列 $\boldsymbol{W}_i$ を均一な標準偏差で初期化する．ここで，パラメータの初期値の平均は 0 であるため，標準偏差は $\|\boldsymbol{W}\|$ の初期値の期待値に比例する．そのため，パラメータの初期値の標準偏差を一定値とすることは，学習初期の更新比率の不均一性を解決する．評価実験により，WeSaR が損失スパイクを抑制し，高性能な言語モデルを学習することを確認した．

2　準備

本節では Transformer [3] の初期化手法とその背景を概説する．

表 1　各初期化手法が設定する標準偏差（Weight 列）と，ゲートを通過した $\bar{\boldsymbol{W}}$ の標準偏差（Gate 列）．

	He		Small		Proposed	
	Gate	Weight	Gate	Weight	Gate	Weight
$\boldsymbol{W}_e$	1	$\sqrt{\frac{1}{d}}$	1	$\sqrt{\frac{2}{5d}}$	1	σ
$\boldsymbol{W}_k$	N/A	$\sqrt{\frac{1}{d}}$	N/A	$\sqrt{\frac{2}{5d}}$	$\sqrt{\frac{1}{d}}$	σ
$\boldsymbol{W}_q$	N/A	$\sqrt{\frac{1}{d}}$	N/A	$\sqrt{\frac{2}{5d}}$	$\sqrt{\frac{1}{d}}$	σ
$\boldsymbol{W}_v$	N/A	$\sqrt{\frac{1}{d}}$	N/A	$\sqrt{\frac{2}{5d}}$	$\sqrt{\frac{1}{d}}$	σ
$\boldsymbol{W}_o$	N/A	$\sqrt{\frac{1}{2Nd}}$	N/A	$\sqrt{\frac{2}{10Nd}}$	$\sqrt{\frac{1}{2Nd}}$	σ
$\boldsymbol{W}_u$	N/A	$\sqrt{\frac{1}{d}}$	N/A	$\sqrt{\frac{2}{5d}}$	$\sqrt{\frac{1}{d}}$	σ
$\boldsymbol{W}_d$	N/A	$\sqrt{\frac{2}{8Nd}}$	N/A	$\sqrt{\frac{2}{10Nd}}$	$\sqrt{\frac{2}{8Nd}}$	σ
$\boldsymbol{W}_p$	N/A	$\sqrt{\frac{1}{d}}$	N/A	$\sqrt{\frac{2}{5d}}$	$\sqrt{\frac{1}{d}}$	σ

2.1　初期化手法

Transformer のパラメータを，$\boldsymbol{W}_e$ が埋め込み層，$\boldsymbol{W}_k, \boldsymbol{W}_q, \boldsymbol{W}_v, \boldsymbol{W}_o$ がそれぞれ Self-Attention 層の key-, query-, value-, output-projection 層，$\boldsymbol{W}_u, \boldsymbol{W}_d$ が MLP 層の up-, down-projection 層，$\boldsymbol{W}_p$ が予測層とする．また，d を隠れ次元数，N を層数とする．

深層学習モデルの初期化でよく使われる手法は He Initialization [4] である．この手法は，層の前後で勾配のノルムが不変になるように設計されており，モデル全体での勾配爆発・消失を抑制する．しかし，大規模言語モデルの初期化では，より小さい標準偏差で初期化する Small Initialization [5] が経験的に安定する．これらの初期化手法を表 1 にまとめる．なお，文献 [6] は Layer Normalization 前後での勾配のノルムの変化を安定化するために埋め込み層 $\boldsymbol{W}_e$ の出力を標準偏差 1 に再パラメタ化することを提案しており，本研究では全ての実験で採用した．

2.2　理論的背景

本研究では，$\boldsymbol{W}_o, \boldsymbol{W}_d$ の二つの層で標準偏差に $\sqrt{\frac{1}{2N}}$ の小さな値がかけられている点に着目する．この工夫は GPT-2 [7] によって導入されたが，原論文中では根拠が示されていない．この工夫の背景を，ResNet における議論 [8] に基づいて説明する．Self-Attention 変換を f を書く．このとき，Self-Attention 層は $\boldsymbol{y} = f(\mathrm{LN}(\boldsymbol{x})) + \boldsymbol{x}$ の変換を行う．なお，LN は Layer Normalization である．このとき，Self-Attention 層における勾配は

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{x}} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{y}} \frac{\partial \boldsymbol{y}}{\partial \boldsymbol{x}} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{y}} \left(\frac{\partial f(\mathrm{LN}(\boldsymbol{x}))}{\partial \boldsymbol{x}} + \boldsymbol{I} \right) \tag{1}$$

である．ここで，損失関数値を $\mathcal{L}$ とした．よって，$\left| \frac{\partial f(\mathrm{LN}(\boldsymbol{x}))}{\partial \boldsymbol{x}} \right|^2$ のオーダーを s としたとき，勾配のノルムの二乗は Self-Attention 層で $s+1$ 倍されて下の層に伝播する．式 1 内の $+\boldsymbol{I}$ は Residual 結合に起因する項であり，同じ事象は MLP 層でも起こる．よってモデル全体では，$(s+1)^{2N}$ 倍の勾配爆発が起こる．なお，簡便のため MLP 層の勾配のノルムの二乗も s で表した．勾配の N に関する指数的な増大は，N が大きくなる LLM では許容できない．そこで，$\boldsymbol{W}_o, \boldsymbol{W}_d$ 層の初期値の標準偏差を通常の $1/\sqrt{2N}$ 倍とすることで，$E[s] = \mathcal{O}\left(\frac{1}{2N}\right)$ とする．すると，モデル全体での勾配爆発 $(s+1)^{2N}$ が $N \to \infty$ で e に収束し，N に関する指数的な増大を回避できる．

3　提案手法

前節では，$\boldsymbol{W}_d, \boldsymbol{W}_o$ が小さな標準偏差で初期化されることを説明した．標準偏差の不均一性は更新比率の不均一性に繋がり，損失スパイクの一因となると考えられる．本研究では，標準偏差の不均一性を解消する手法を提案する．

3.1　WeSaR

パラメータ $\boldsymbol{W}_i$ の初期化を確率分布 $\mathcal{N}(0, \sigma_i)$ で行う場合を考える．このとき，提案手法は全パラメータで共通の標準偏差を σ として，

$$\boldsymbol{W}_i \sim \mathcal{N}(0, \sigma), \quad \bar{\boldsymbol{W}}_i = \frac{\sigma_i}{\sigma} \boldsymbol{W}_i$$

と再パラメタ化する．つまり，$\boldsymbol{W}_i$ を $\mathcal{N}(0, \sigma_i)$ からサンプリングする代わりに，共通の標準偏差 σ からサンプリングされた $\boldsymbol{W}_i$ をモデルに登録する．そして，パラメータをモデル内で使う際は常に $\frac{\sigma_i}{\sigma}$ をかけた $\bar{\boldsymbol{W}}_i$ として使用する．よって，$\bar{\boldsymbol{W}}_i$ の標準偏差は σ_i となる．ここで，$\frac{\sigma_i}{\sigma}$ は実際には学習可能なゲートパラメータ α_i とし，その初期値を $\frac{\sigma_i}{\sigma}$ とする．

再パラメタ化は，モデルの挙動を変えることなく均一な標準偏差での初期化を実現する．また，ゲートパラメータは事前学習後に元のパラメータにマージ可能である．マージ後のモデルは元にしたモデルと同じ構造を持つため，ライブラリ互換である．LLM の推論では専用のライブラリを用いることが多く，ライブラリ互換性は実用上重要である．

3.2　理論的正当性

Transformer の学習では，最適化手法 Adam [9] を用いることが安定性に寄与する [10, 11, 12]．本節で

Adam と併用した場合の WeSaR の挙動を示す．

Adam では時刻 t のパラメータの更新量 $\Delta \boldsymbol{W}_t$ を

$$\Delta \boldsymbol{W}_t = \mu_t \frac{\boldsymbol{M}_t}{\sqrt{\boldsymbol{V}_t}}, \tag{2}$$

と定義する．ここで，$\boldsymbol{M}_t$ は勾配 $\frac{\partial \mathcal{L}}{\partial \boldsymbol{W}}$ の指数移動平均，$\boldsymbol{V}_t$ は勾配の二乗の指数移動平均，μ_t は学習率である．ゲートパラメータを用いたとき，$\boldsymbol{W}$ の勾配は

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{W}_{\cdot}} = \frac{\partial \mathcal{L}}{\partial \bar{\boldsymbol{W}}_{\cdot}} \frac{\partial \bar{\boldsymbol{W}}_{\cdot}}{\partial \boldsymbol{W}_{\cdot}} = \frac{\sigma_{\cdot}}{\sigma} \frac{\partial \mathcal{L}}{\partial \bar{\boldsymbol{W}}_{\cdot}}$$

より，$\bar{\boldsymbol{W}}$ の勾配から $\frac{\sigma_{\cdot}}{\sigma}$ 倍される．Adam のパラメータ更新 (式 2) は分母・分子ともに $\mathcal{O}\left(\frac{\partial \mathcal{L}}{\partial \boldsymbol{W}}\right)$ であるため，ゲートパラメータの影響は相殺される．

すなわち，WeSaR は再パラメタ化によって，パラメータ $\boldsymbol{W}$ から初期化手法が指定する標準偏差 $\sigma_{\cdot}$ の影響を排除する．全てのパラメータを再パラメタ化する利点は，どんなに極端な初期化手法であっても，その初期化手法を採用しながら，実際に更新されるパラメータを均一な標準偏差に設定できる点である．副次的に，初期値のスケールを d に依存させる既存研究と異なり，d への依存もゲートパラメータ α が担うことで，モデルサイズに非依存に初期値のスケールを定めることができる．

また，Adam を用いた場合はパラメータの更新量 $\|\Delta \boldsymbol{W}\|$ は勾配の指数移動平均と勾配の二乗の指数移動平均の平方根で正規化される．この点からも，更新比率 $\|\Delta \boldsymbol{W}\|/\|\boldsymbol{W}\|$ を一定にするためには，パラメータのノルム $\|\boldsymbol{W}\|$，ひいては標準偏差を一定にする必要があることが示唆される．

3.3 ハイパーパラメータ設定

ハイパーパラメータ設定の指針の一つに学習の安定性がある．提案手法が LLM の事前学習を安定化することで，慣習的な設定よりも急速な損失関数値の減少を実現するハイパーパラメータ値を設定できる．なお，提案手法の σ_i は He Initialization の規準によって決定した．

標準偏差 σ． 既存手法と異なり，WeSaR はパラメータの標準偏差 σ を任意の値に設定できる．本研究では，$\sigma^2 = 4\text{e-}5$ と設定した．この値は，Small Initialization の規準 $\sqrt{\frac{2}{5d}}$ では $d = 10,000$ のモデルに相当する．つまり，我々は σ を慣習的な値よりも小さい標準偏差を設定した[1)]．文献 [14] も小さな標準偏差が Transformer の学習に望ましいことを指摘しており，本研究の設定は妥当であると考える．

1) LLaMA3 70B モデルでは $d = 8192$ である [13].

学習率． 高速な学習を意図し，慣習的な値（1e-4 オーダ）よりも大きな 1e-3 を採用した．

バッチサイズ． LLM の事前学習を安定化させるため，バッチサイズは 4M トークンなどの大きな値を指定することが多い．小さいバッチサイズを指定することで，同じコーパスで学習したときのステップ数を増やすことができる．特に日本語データのように限られたデータで事前学習を行う際には，ステップ数を確保することが重要になる．一方で，LLM の事前学習は多数の GPU で並列に計算することが多いため，計算効率の観点からはリソースに合わせて設定することが望ましい．本研究では 1M トークンとして実験を行った．

4 評価

4.1 実験設定

パラメータ数 13 億 (1.3B) 個 と 130 億 (13B) 個の Transformer モデルの事前学習により評価した．RefinedWeb [15] データセットから 30B トークンをサンプリングして事前学習を行なった．評価尺度は LAMBADA [16] と WikiText [17] における perplexity とした．ハイパーパラメータとして，慣習的な値である Stable 設定（学習率 5e-4, バッチサイズ 4M）と，より高速な学習を意図とした Rapid 設定（学習率 1e-3, バッチサイズ 1M）を採用した．詳細を付録 A に示す．

4.2 比較手法

ベースラインモデルとして，LLM の事前学習で広く用いられる Small Initialization を採用した．また，再パラメタ化に基づく既存手法 Weight Normalization [18], σReparam [19], Residual Scaling [20] を比較手法に用いた．詳細を付録 B に示す．前者 2 手法と WeSaR との違いに，比較手法がパラメータの正規化を伴う点がある．パラメータの正規化は計算量を増加させる．Residual Scaling は $\boldsymbol{W}_d, \boldsymbol{W}_o$ のみで再パラメタ化を行う手法であり，WeSaR は全てのパラメータ行列を再パラメタ化する点が異なる．

4.3 結果

提案手法は損失スパイクを抑制するか？ 図 1 は学習初期における Rapid 設定の 2 モデルの比較である．提案手法は学習初期の損失関数値の減衰を安定化し，かつ更新比率を一定の小さな範囲に抑制し

表 2　事前学習済み 1.3B モデルの perplexity．本実験のみ 10B トークンでの事前学習とし，5 回の実験の平均と標準偏差を示す．最良の結果を太字，1 標準偏差以内の結果を下線で示す．

	WikiText	LAMBADA	Time	# Param.	Best σ^2
Small Init.	20.64 (0.52)	29.50 (0.53)	18.88	1,339.1M	N/A
Weight Normalization	18.87 (0.59)	<u>27.69</u> (0.86)	21.27 (+12.6%)	1,339.6M	16e-5
σReparam	23.64 (1.03)	30.56 (0.89)	20.09 (+6.39%)	1,339.1M	16e-5
Residual Scaling	23.56 (1.03)	30.78 (0.35)	19.18 (+1.58%)	1,339.1M	N/A
WeSaR	**17.74** (0.05)	**27.52** (0.28)	19.25 (+1.95%)	1,339.1M	4e-5

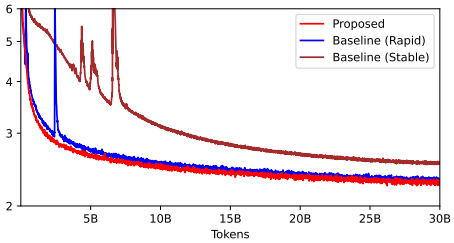

図 2　13B モデルの学習中の損失関数値．ベースラインモデルでは学習の初期に損失スパイクが発生．

表 3　事前学習済モデルの perplexity．

		WikiText	LAMBADA
1.3B	Small Init. (Rapid)	16.55	26.29
	Small Init. (Stable)	21.44	28.81
	WeSaR	**14.51**	**22.87**
13B	Small Init. (Rapid)	12.72	21.79
	Small Init. (Stable)	18.66	25.34
	WeSaR	**12.05**	**21.57**

ている．図 2 はベースラインの Stable 設定を加え，30B トークン全てでの学習の様子を示す．Stable 設定であっても損失スパイクが発生すること，提案手法は 30B トークンまで損失スパイクを起こさないことを確認できる．

提案手法は言語モデルの性能を向上するか？　表 3 に事前学習済みモデルの perplexity を示す．損失関数値に加えて言語モデルの perplexity としても，提案手法は Small Initialization で学習したモデルを上回った．

提案手法の設計は既存の再パラメタ化手法に対して優れるか？　表 2 に再パラメタ化手法との比較を示す．提案手法は perplexity に関して，全ての既存手法よりも小さい平均値と標準偏差を示す．よって，提案手法は高性能な言語モデルを安定して学習したと考える．加えて，パラメータの正規化を伴う Weight Normalization と σReparam は大きな計算量の増大が観察できる．Residual Scaling に対しては，$\boldsymbol{W}_d, \boldsymbol{W}_o$ だけでなく全てのパラメータ行列を再パラメタ化し均一な初期化を実現することが，言語モデルの性能向上に寄与することが確認できる．

5　関連研究

PaLM [1] と OPT [2] は損失スパイクを発見し，スパイクの直前から再開してデータの一部をスキップすることで対処した．GLM [21] は埋め込み層における異常な勾配が損失スパイクを引き起こすことを発見した．文献 [22, 19] は長い系列長や Attention の異常な振る舞いが原因であることを指摘した．文献 [23] は Adam が仮定する時刻に関する独立性を一因として挙げた．文献 [6] は Layer Normalization の微分に着目し，埋め込み層出力の正規化 [3, 24] の有効性を確認した．損失スパイクを対象とする研究の多くは勾配や更新量に着目しているが，本研究はパラメータの更新比率の不均一性に初めて着目した．

6　おわりに

本研究は LLM の事前学習における損失スパイクの原因とその抑制手法を提案した．

本研究の独自性．　本研究はパラメータの更新比率に初めて着目した．学習初期に更新比率が大きいパラメータが存在すること，損失スパイクの前後で更新比率が大きく減少することを指摘した．更新比率の大きいパラメータの存在が GPT-2 の提案した小さな標準偏差に起因することを議論した．

本研究の重要性．　再パラメタ化により全てのパラメータを均一な標準偏差で初期化することを提案し，更新比率の不均一性を解消した．提案手法は 13B モデルの 30B トークンまでの事前学習において損失スパイクを抑制した．提案手法はシンプルかつ，学習後はゲートパラメータをマージすることでライブラリ互換性を損なわないため，導入が容易である．本研究は損失スパイクに対してパラメータの更新比率に基づく新たな角度からの分析を可能とし，発生原理の解明の加速が期待できる．

参考文献

[1] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. **Journal of Machine Learning Research**, Vol. 24, No. 240, pp. 1–113, 2023.

[2] Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. Opt: Open pre-trained transformer language models. **arXiv preprint arXiv:2205.01068**, 2022.

[3] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In **NIPS**, pp. 5998–6008, 2017.

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In **Proceedings of the IEEE International Conference on Computer Vision**, December 2015.

[5] Toan Q. Nguyen and Julian Salazar. Transformers without tears: Improving the normalization of self-attention. In **Proceedings of the 16th International Conference on Spoken Language Translation**, November 2-3 2019.

[6] Sho Takase, Shun Kiyono, Sosuke Kobayashi, and Jun Suzuki. Spike no more: Stabilizing the pre-training of large language models. **arXiv preprint arXiv:2312.16903**, 2023.

[7] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. **OpenAI blog**, 2019.

[8] Masato Taki. Deep residual networks and weight initialization. **arXiv preprint arXiv:1709.02956**, 2017.

[9] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In **ICLR (Poster)**, 2015.

[10] Jingzhao Zhang, Sai Praneeth Karimireddy, Andreas Veit, Seungyeon Kim, Sashank Reddi, Sanjiv Kumar, and Suvrit Sra. Why are adaptive methods good for attention models? **Advances in Neural Information Processing Systems**, Vol. 33, pp. 15383–15393, 2020.

[11] Yan Pan and Yuanzhi Li. Toward understanding why adam converges faster than SGD for transformers. In **OPT 2022: Optimization for Machine Learning (NeurIPS 2022 Workshop)**, 2022.

[12] Yushun Zhang, Congliang Chen, Tian Ding, Ziniu Li, Ruoyu Sun, and Zhi-Quan Luo. Why transformers need adam: A hessian perspective. **arXiv preprint arXiv:2402.16788**, 2024.

[13] AI@Meta. Llama 3 model card. 2024.

[14] Biao Zhang, Ivan Titov, and Rico Sennrich. Improving deep transformer with depth-scaled initialization and merged attention. In **Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing**, pp. 898–909, November 2019.

[15] Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. The refinedweb dataset for falcon llm: outperforming curated corpora with web data, and web data only. **arXiv preprint arXiv:2306.01116**, 2023.

[16] Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Ngoc Quan Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. The LAMBADA dataset: Word prediction requiring a broad discourse context. In Katrin Erk and Noah A. Smith, editors, **Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)**, pp. 1525–1534, August.

[17] Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In **International Conference on Learning Representations**, 2017.

[18] Tim Salimans and Durk P Kingma. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, **Advances in Neural Information Processing Systems**, Vol. 29, 2016.

[19] Shuangfei Zhai, Tatiana Likhomanenko, Etai Littwin, Dan Busbridge, Jason Ramapuram, Yizhe Zhang, Jiatao Gu, and Joshua M. Susskind. Stabilizing transformer training by preventing attention entropy collapse. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, **Proceedings of the 40th International Conference on Machine Learning**, Vol. 202 of **Proceedings of Machine Learning Research**, pp. 40770–40803, 23–29 Jul 2023.

[20] Lorenzo Noci, Sotiris Anagnostidis, Luca Biggio, Antonio Orvieto, Sidak Pal Singh, and Aurelien Lucchi. Signal propagation in transformers: Theoretical perspectives and the role of rank collapse. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, **Advances in Neural Information Processing Systems**, Vol. 35, pp. 27198–27211, 2022.

[21] Aohan Zeng, Xiao Liu, Zhengxiao Du, Zihan Wang, Hanyu Lai, Ming Ding, Zhuoyi Yang, Yifan Xu, Wendi Zheng, Xiao Xia, Weng Lam Tam, Zixuan Ma, Yufei Xue, Jidong Zhai, Wenguang Chen, Zhiyuan Liu, Peng Zhang, Yuxiao Dong, and Jie Tang. GLM-130b: An open bilingual pre-trained model. In **The Eleventh International Conference on Learning Representations**, 2023.

[22] Conglong Li, Minjia Zhang, and Yuxiong He. The stability-efficiency dilemma: Investigating sequence length warmup for training gpt models. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, **Advances in Neural Information Processing Systems**, Vol. 35, pp. 26736–26750, 2022.

[23] Igor Molybog, Peter Albert, Moya Chen, Zachary DeVito, David Esiobu, Naman Goyal, Punit Singh Koura, Sharan Narang, Andrew Poulton, Ruan Silva, et al. A theory on adam instability in large-scale machine learning. **arXiv preprint arXiv:2304.09871**, 2023.

[24] Teven Le Scao, Thomas Wang, Daniel Hesslow, Stas Bekman, M Saiful Bari, Stella Biderman, Hady Elsahar, Niklas Muennighoff, Jason Phang, Ofir Press, Colin Raffel, Victor Sanh, Sheng Shen, Lintang Sutawika, Jaesung Tae, Zheng Xin Yong, Julien Launay, and Iz Beltagy. What language model to train if you have one million GPU hours? In **Findings of the Association for Computational Linguistics: EMNLP 2022**, pp. 765–782, December 2022.

[25] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. **arXiv preprint arXiv:2302.13971**, 2023.

[26] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[27] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Huggingface's transformers: State-of-the-art natural language processing. **arXiv preprint arXiv:1910.03771**, 2019.

[28] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In **International Conference on Learning Representations**, 2018.

A 実験設定

表 4 と表 5 に設定を示す．モデル構造は LLaMA [25] に準拠したが，MLP 層の活性化関数には gelu 関数を用いた．全てのパラメータを再パラメタ化する手法では，標準偏差 σ^2 を $\{1, 4, 16, 64, 256\}$e-5 の中で最良のものに設定した．実験には 1.3B モデルで 8 個の NVIDIA H100 (80GB) GPUs，13B モデルで 64 個の GPUs を利用した．それぞれのモデルの事前学習に 60 時間と 40 時間を要した．実験には PyTorch (ver. 2.1.0)[2] [26], transformers (ver. 4.37.2)[3] [27], llm-foundry (ver. 0.5.0) [4] のライブラリを用いた．

表 4 モデル設定．

	1.3B	13B
隠れ次元数 d	2048	5120
層数 N	24	40
Attention Head 数	16	40
系列長	2048	
語彙サイズ	32000	
RMSNorm ϵ	1e-5	
位置埋込	RoPE	
線形層のバイアス項	none	

表 5 訓練設定．

	Rapid Setting	Stable Setting
バッチサイズ [tokens]	1M	4M
学習率 μ	1e-3	5e-4
Warmup Steps	100	2000
精度	bfloat16	
コーパスサイズ [tokens]	30B	
Adam β_1	0.9	
Adam β_2	0.95	
Gradient Clipping Threshold	1	
Weight decay	0.01	
Z-loss	1e-4	

B 比較手法

本節では再パラメタ化に基づく比較手法について，各手法の目的・設計と WeSaR との違いを説明する．表 6 に概要を列挙する．

表 6 再パラメタ化手法の比較．“対象” は再パラメタ化の対象のパラメータを示す．“訓練” はゲートパラメータの訓練の有無である．“正規化” はパラメータの正規化の有無である．“単位” はスケーリングが行列単位か行単位かを示す．

手法	対象	訓練	正規化	単位
Weight Normalization	all	✓	✓	by-row
σReparam	all	✓	✓	by-matrix
Residual Scaling	$\boldsymbol{W}_o, \boldsymbol{W}_d$			by-matrix
WeSaR	all	✓		by-matrix

B.1 Weight Normalization

Weight Normalization [18] はパラメータのノルムの更新と方向の更新を分離するために提案された．Weight Normalization はパラメータ行列の各行 $\boldsymbol{w} \in \mathbb{R}^{d_{\text{in}}}$ に対して，L2 正規化と再パラメタ化を行う

$$\bar{\boldsymbol{w}} = \frac{\alpha}{\|\boldsymbol{w}\|}\boldsymbol{w}.$$

WeSaR は Weight Normalization と比較して，行列単位で再パラメタ化し，正規化を行わない点について計算効率が高い．Weight Normalizatio はパラメータ行列 $\boldsymbol{W}$ に対して d_{out} 個のゲートパラメータが必要であるが，提案手法は 1 個である．

B.2 σReparam

σReparam [19] は Self-Attention 層における Attention のエントロピーの安定性が学習の安定性に寄与する発見に基づき，パラメータの特異値を制御することで Attention の安定化を図る手法である．σReparam はパラメータ行列 $\boldsymbol{W} \in \mathrm{R}^{d_{\text{out}} \times d_{\text{in}}}$ に対して特異値正規化と再パラメタ化を行う

$$\bar{\boldsymbol{W}} = \frac{\alpha}{\|\boldsymbol{W}\|_2}\boldsymbol{W}.$$

ここで，$\|\boldsymbol{W}\|_2$ はスペクトルノルム（最大特異値）である．最大特異値は毎ステップ一回の冪乗法によって計算する [28]．特異値計算に関しては勾配計算を行わない．ゲートパラメータ α は 1 に初期化する．σReparam は初期値の最大特異値を 1 にすることを目的とし，WeSaR は任意の初期化手法（e.g., He Initialization）と組み合わせながら全てのパラメータを共通の標準偏差で初期化することを目的とする．

B.3 Residual Scaling

文献 [20] は $\boldsymbol{W}_d, \boldsymbol{W}_o$ の初期値の標準偏差を $1/\sqrt{2N}$ 倍する代わりに，Residual 結合を

$$\boldsymbol{y} = \frac{1}{\sqrt{2N}} f(\mathrm{LN}(\boldsymbol{x})) + \boldsymbol{x}.$$

と計算する手法を提案した．f 内部の最後の変換の線形性により，この手法は $\boldsymbol{W}_d, \boldsymbol{W}_o$ の再パラメタ化として解釈できる．WeSaR は全てのパラメータを再パラメタ化し，共通の標準偏差を採用することを提案した．

2) https://pytorch.org/
3) https://github.com/huggingface/transformers
4) https://github.com/mosaicml/llm-foundry