モデル拡張によるパラメータ効率的な LLM の事前学習

矢野 一樹 [1]　伊藤 拓海 [1,2]　鈴木 潤 [1,3,4]
[1] 東北大学　[2]Langsmith 株式会社　[3] 理化学研究所　[4] 国立情報学研究所
yano.kazuki@dc.tohoku.ac.jp　{t-ito,jun.suzuki}@tohoku.ac.jp

概要

大規模言語モデル（LLM）の事前学習は，モデルパラメータの多さからメモリ要求量の問題に直面する．本稿では，モデル拡張を用いたメモリ効率に優れた事前学習法である STEP を提案する．実験結果から，STEP を用いることにより，標準的な事前学習と比較して，最大で 53.9%の最大メモリ要求量を削減しながら同等の性能を達成することを確認した．さらに，STEP により訓練されたモデルが，下流タスクに対しても標準的に訓練されたモデルと同等の性能を達成することを示す．

1　はじめに

大規模言語モデル（LLM）は，人工知能分野の基盤的な技術となり，実用システムにおいても用いられるようになった．性能の高い LLM を獲得するためには，スケーリング則 [1] に基づき，膨大なパラメータを持つ Transformer モデル [2] を大規模コーパスで事前学習する流れが主流である [3]．結果として，事前学習を現実的な時間で終えるには膨大な数の GPU を搭載した計算資源が要求される [4]．この巨大な計算資源の要求は，LLM の事前学習研究への重大な参入障壁となっている．

計算資源量の課題を受け，本研究では，LLM 事前学習の計算資源量削減について議論する．計算資源量削減には様々な観点があり得るが，本稿では，標準的な事前学習と同等の性能を維持しながら，事前学習に必要となる GPU の最大メモリ要求量を一定量に抑制可能な事前学習法を提案する．概要としては，層追加によるモデル拡張法と，微調整学習でよく用いられるパラメータ効率のよい学習法を組合せた新たな事前学習法を構築する．提案法の具体的な説明として，図 1 に提案法の手順に関する概要図を示す．提案法では，逐次モデル拡張手順での各ステージの最大メモリ要求量をモデル設定値を変数とした整数計画問題として定式化し，その解を得て実際の各ステージでのモデル設定に用いることで，事前学習の実行前に実行中の最大メモリ要求量が最小になるようにモデル拡張の設定を制御する．これにより，最大メモリ要求量を一定量に抑えた事前学習が実現できる．以降，本提案手法を STEP（STaged parameter Efficient Pre-training）と記述する．

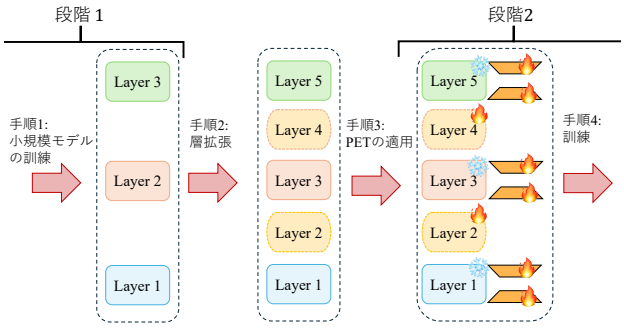

図 1　STEP の概要図．小規模なモデルに対して標準的な事前学習を実施する（手順 1）．その後，学習済みモデルに新たに層を追加しモデルを拡張する（手順 2）．学習済みの層はパラメータを固定し，代替の学習用に PET（Parameter-Efficient Training）を適用（手順 3），拡張後のモデルを再度学習する（手順 4）．手順 4 では，層拡張で追加したパラメータと PET によって導入された小規模なパラメータのみが学習対象となる．

本稿では，一定の FLOPs の条件下で，既存手法との性能比較を通じて STEP の有効性を検証する．その結果，従来の事前学習法と比べ，事前学習モデルの標準的な評価タスクにおいて同等の性能を維持しながら，最大メモリ要求量を 53.9%削減可能であることが示された．さらに，標準的な微調整学習を実施した場合でも，STEP で学習したモデルは，標準的な事前学習によるモデルと下流タスクにおいて同等の性能を達成することを示す．

2　関連研究

メモリ効率の良い事前学習に向けて，様々な手法が提案されている [5, 6]．主要なアプローチの一つとして，訓練対象のパラメータを削減する方法が挙げられる．その代表例として，Adapter [7] や

LoRA [8] などのパラメータ効率の良い調整（PET）方法が挙げられる．

一方，事前学習時の FLOPs 削減のため，小規模なモデルから学習を開始し，その後モデルパラメータを拡大させ学習を継続するモデル拡張と呼ばれる手法が提案されている [9, 10]．本提案手法は，PET とモデル拡張を適切に組み合わせることにより，メモリ効率の良い事前学習の達成を目指している．

2.1 パラメータ効率的調整

PET は主に LLM の微調整学習を目的として開発されてきた．例えば，LoRA は事前学習済みの LLM のパラメータを固定したまま，新たに適応器（低ランク行列）を追加し，その適応器のみを学習する手法である．適応器は通常少ないパラメータであるため，少ない計算資源で学習ができる．PET は事前学習にも適用されつつある．ここでは，その代表的な手法である ReLoRA [11] と GaLore [12] について説明する．ReLoRA は LoRA を用いて LLM の事前学習を行う手法である．ReLoRA の特徴として，学習の初期は標準的な事前学習を行い，途中から LoRA を適用する．つまり，最大のメモリ要求量の観点では，ReLoRA は標準的な事前学習とメモリ要求量が同じである．GaLore は勾配の低ランク構造を活用し，標準事前学習と同等の性能を維持しながら最適化器の状態を削減する手法である．GaLore は，ReLoRA とは異なり，学習プロセス全体を通して低メモリで動作する．また，これらの手法は標準的な事前学習と比較して，性能低下を伴うことが指摘されている．

2.2 モデル拡張

モデル拡張を適用することで，大規模なモデルをスクラッチから学習する場合と比較して，同等の性能を，より少ない FLOPs で達成できることが報告されている [13, 9, 10]．モデルを小規模なものから大規模なものに拡張させる操作は**拡張演算子**と呼ばれ，Transformer モデルの層の次元や新規層を増加させる操作をさす．通常，モデル拡張を用いる場合，全パラメータを訓練させるため，最大メモリ要求量はモデルサイズと共に増加する．

3 提案法：STEP

提案法の詳細を説明する．図 1 に概要図を示す．STEP では以下の手順で LLM を事前学習する．

手順 1．小規模モデルの訓練　最終的に訓練したいモデルサイズよりも小規模なモデルを初期モデルとして訓練する．

手順 2．拡張演算子の適用　モデルサイズを増大させるため，初期モデルに対して層追加を行う．

手順 3．PET の適用　手順 1 で訓練された層に対して PET を適用する．

手順 4．訓練の継続　手順 1 で学習した層のパラメータを固定したまま，手順 2 で新たに追加された層のパラメータと手順 3 で追加された適応器のパラメータの学習を継続する．

手順 4 の完了により事前学習済みモデルが構築される．または，手順 2 から 4 を反復することで，さらなる層の拡張も可能である．ここで，**i 番目の段階 (ステージ)** は，$i = 1$ のとき手順 1 による最初のモデルの訓練を，$i \geq 2$ のときは $i - 1$ 回目の手順 2 から 4 による訓練を指す．

層拡張演算子と PET　層拡張演算子は，モデルの層自体の構造を変化させる．本研究では既存の層の間に，新規層を追加する内挿法 [14, 15, 16] という層拡張演算子を使用する．また，新規層の初期化法として，上層と下層のパラメータの平均をとったものを初期値とする方法 [17] を採用する．手順 3 における PET として，ReLoRA [11] を選択した．

3.1 STEP の最大メモリ要求量

STEP では，その最大メモリ要求量に応じて，小規模モデルのサイズや追加する層数を設定する必要がある．そのため，STEP での最大メモリ要求量について議論する．本研究では，事前学習中におけるメモリの要求量は「モデル状態」と呼ばれる実体で推定されることを仮定する．モデル状態とは，モデル自体の重み，更新の際に必要とされる勾配，および最適化器の状態で構成される．さらに，モデルが一般的な Transformer モデル，最適化器が Adam，そして混合精度学習が行われると仮定する．この場合，重みと勾配は 16-bit の浮動小数点で，最適化器の状態は 32-bit の浮動小数点で表現される．一つの Transformer 層のパラメータ数 P_{layer} とし，モデルの層数を n とすると，メモリ要求量は

$$\begin{aligned} P_{\text{trn}} &= n(\underbrace{2P_{\text{layer}}}_{\text{model}} + \underbrace{2P_{\text{layer}}}_{\text{gradient}} + \underbrace{12P_{\text{layer}}}_{\text{optimizer}}) \\ &= 16nP_{\text{layer}}, \end{aligned} \tag{1}$$

となる．このとき，Adam 最適化器の状態は，重み，勾配の移動平均，および勾配の 2 乗の移動平均からなる．STEP の最大メモリ要求量に関して，$i-1$ 番目の段階から i 段階目の段階で追加される層数を n_i，N_i を i 番目の段階における全パラメータ数とすると，$N_0 = 0$ として，$N_i = \sum_{k=1}^{i} n_k$ となる．さらに，$E(P_{\text{layer}})$ をパラメータ数 P_{layer} の Transformer 層に対して PET で導入されるパラメータ数とする．すると，i 番目の段階における，STEP の最大メモリ要求量 P_i^{STEP} は次のようになる．

$$\begin{aligned} P_i^{\text{STEP}} =& 16 n_i P_{\text{layer}} + 2 N_{i-1} P_{\text{layer}} \\ &+ 16 N_{i-1} E(P_{\text{layer}}) \end{aligned} \tag{2}$$

ここで $2N_{i-1}P_{\text{layer}}$ は 1 段階目から $i-1$ 段階目でに既に訓練され，i 番目の段階では既に固定されている層部分のパラメータを指す．また，$16n_iP_{\text{layer}}$ は手順 2 において，追加された新規層のモデル状態を表す．最後に，$16N_{i-1}E(P_{\text{layer}})$ は手順 3 において導入される適応器自体のモデル状態を表している．

最終的に得られるモデルの層数を L とすると，以下の最小化問題の解が事前学習中の最大メモリ要求量を最小化することが可能である：

$$\underset{\{n_1,\ldots,n_K\}}{\text{minimize}} \left\{ \max_{i=1,\ldots,K} P_i^{\text{STEP}} \right\} \quad \text{s.t. } L = N_K. \tag{3}$$

この最小化問題はすべての i に対して n_i が非負整数であることから，本質的には整数線形計画問題である．そのため，標準的なソルバーを用いることで，あるいは K が 小さい場合（e.g. $K = 2$）は手計算によって，解集合 $\{n_i\}_{i=1}^{K}$ を得ることができる．

4 実験

本稿では，(i) STEP が効率的な事前学習を実現できるかを検証し，(ii) STEP で事前学習されたモデルに後段タスクに対して微調整学習を行った場合の影響を調査する．後段タスクとしては，指示調整を採用する．

4.1 事前学習実験

事前学習での学習量を，一定の FLOPs を上限に設定し，その中でベースライン手法と STEP での学習効率について調査する．[1)] ベースライン手法としては，PET やモデル拡張を適用しない標準的な事前学習法（標準事前学習）と，既存のパラメータ効率的な事前学習方法である ReLoRA と GaLore を採用した．

1) FLOPs の詳細な計算方法は付録 A にて述べる．

表 1 実験で使用した STEP の設定．各段階におけるモデルのパラメータ数と層数を示す．最下段は 3 段階の拡張過程を表す．

	モデルサイズ	層数
STEP-3stages	215M →368M	7 →12
STEP-2stages	396M →680M	14 →24
STEP-2stages	704M →1.2B	14 →24
STEP-3stages	553M →956M →1.2B	11 →19 →24

学習データセットとモデル　事前学習用のデータセットとして FineWeb-Edu データセット [18] を用いた．モデルのアーキテクチャは LLaMA [4] の設定に従った．異なるモデルのサイズでも同様の結果が導かれるかを検証するため，368M，680M，1.2B の 3 つのサイズに対して実験を行った．詳細な訓練設定について付録 B にて述べる．

評価方法　各手法を評価するため，評価用データにおける Perplexity を計算した．1 つは訓練データセットである FineWeb-Edu (10M トークン) から，もう一つは Wiki-Text (0.3M トークン) [19] からのものである．加えて，事前学習モデルの標準的な評価指標である，複数の下流タスクにおけるゼロショット性能を測定した．言語モデリングタスクとして，LAMBADA [20] を，常識推論タスクとして WinoGrande [21]，PIQA [22]，HellaSwag [23] を，質問応答タスクとして ARC [24] と OBQA [25] を採用し，これらに対する Accuracy の平均値を算出した．

STEP の設定　本実験では，STEP の手順 2 から 4 を 1 回のみ，すなわち訓練を 2 段階かけて行う場合 ($K = 2$) の評価を行う．これを STEP-2stages と表現する．さらに，1.2B サイズのモデルについてのみ，手順 2 から 4 を 2 回繰り返し，訓練を 3 段階かけて行う場合 ($K = 3$) の評価も行う．これを STEP-3stages と表現する．Transformer 層の次元数を固定した状態で層数 L が与えられた場合，式 3 により，最大メモリ要求量を最小化できる追加層数を計算する．例えば，1.2B サイズのモデルにおいて，STEP-2stages の場合，$\{n_1 = 14, n_2 = 10\}$，STEP-3stage の場合，$\{n_1 = 11, n_2 = 8, n_3 = 5\}$ である．表 1 は，最終的に構築する目標のモデルサイズが{368M, 680M, 1.2B}のいずれかの場合における，計算された追加層数を考慮した全層数を示している．図 2 は，標準事前学習と STEP-3stages の各段階における，目標モデルサイズが 1.2B の場合のメモリ要求量の例を示

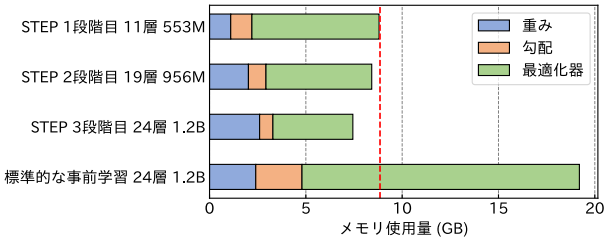

図 2　1.2B モデルの 3 段階での STEP 適用時のメモリ要求量（表 1 の最下段）．STEP では各段階でメモリ要求量を一定にしながらモデルサイズを拡張することが可能である．

表 2　事前学習実験の結果．括弧内の数値は，各手法の事前学習に必要な最大メモリ量を示している．

	Perplexity ↓		Accuracy ↑
	Validation	Wikitext	下流タスク
368M			
標準事前学習 (5.9G)	16.9	32.1	41.9
ReLoRA (5.9G)	17.4	33.1	41.7
GaLore (3.3G)	21.6	43.1	39.0
STEP-2stages (3.4G)	**16.7**	**31.5**	**42.5**
680M			
標準事前学習 (10.9G)	**14.6**	**26.0**	**46.0**
ReLoRA (10.9G)	15.1	27.3	44.6
GaLore (6.0G)	19.4	37.5	39.6
STEP-2stages (6.3G)	**14.6**	**26.0**	**46.0**
1.2B			
標準事前学習 (19.3G)	**12.9**	**22.1**	48.5
ReLoRA (19.3G)	13.5	23.6	47.4
GaLore (10.4G)	17.4	35.3	41.6
STEP-2stages (10.6G)	**12.9**	22.3	**49.3**
STEP-3stages (8.9G)	**12.9**	**22.1**	48.8

している．層拡張演算子の適用タイミングは，各段階の総訓練ステップの 75%が完了した時点で実行されるように設定した．

結果　表 2 は，標準事前学習, ReLoRA，GaLore，および STEP の評価データに対する Perplexity や Accuracy を示している．STEP は ReLoRA と GaLore の両方を上回る性能を示した．さらに，STEP は最大メモリ要求量を 368M モデルで 5.9G から 3.4G（42.3%），680M モデルで 10.9G から 6.3G（42.2%），1.2B モデルで 19.3G から 8.9G (53.9%) と大幅に削減しながら，標準事前学習と同等の性能を達成した．さらに，1.2B モデルにおける STEP-2stages と STEP-3stages による結果は，段階数を増やすことで，性能を劣化することなく，メモリ要求量をさらに削減可能なことを示している．これらの結果は，STEP がメモリ要求量を削減しながら，効率的に LLM を事前学習可能なことを示唆している．

表 3　指示調整実験の結果．MT-Bench スコアは 1 から 10 の範囲で，高いほど良い応答品質を示す．

	MT-Bench ↑
標準事前学習モデル 1.2B	2.26
STEP-2stages 1.2B	**2.30**
STEP-3stages 1.2B	2.26

4.2　指示調整実験

節 4.1 にて事前学習を行ったモデルに対して，指示調整を行い，その性能を評価し，STEP が後段のタスクに悪影響を及ぼさないことを検証する．

実験設定　指示調整用のデータセットとして Alpaca データセット [26] を使用し，評価には MT-Bench [27] を使用する．標準事前学習モデル，STEP-2stages，および STEP-3stages のモデルに対して指示調整を行なった．いずれもモデルサイズは 1.2B である．指示調整の詳細な訓練設定は付録 C にて述べる．

結果　表 3 は標準事前学習モデル，STEP-2stages，STEP-3stage の MT-Bench スコアを示している．この表から，STEP で訓練されたモデルのスコアが，事前学習モデルのスコアと同等であることがわかる．これらの結果は STEP が下流タスクに悪影響を及ぼさず，標準事前学習と同様の効果をもたらしていることを示唆している．

5　おわりに

本研究では，GPU の最大メモリ要求量に着目し，これを一定量に抑えて事前学習ができる効率的な学習法として STEP を提案した．STEP では，モデル拡張の考えを導入し段階的にモデルを拡張しながら事前学習する．その際に，各段階の最大メモリ要求量を最小化する整数計画問題をモデル構成の設定値を変数として定式化する．そして，その解を得ることで，最大メモリ要求量を一定量以下に抑制する各段階でのモデル設定を得る．最終的に，各ステージのモデル設定に適用することで，最大メモリ要求量を一定量以下に抑制できる．実験により，STEP は最大メモリ要求量を事前に定義した一定量以下に制限しながら，標準的な事前学習，および，その後の指示調整をしたモデルと同等のタスク性能を達成可能であることを示した．

本研究の成果が，LLM の事前学習の民主化に向けた一助になることを期待している．

謝辞

本研究は，JST ムーンショット型研究開発事業 JPMJMS2011-35 (fundamental research)，および，文部科学省の補助事業「生成 AI モデルの透明性・信頼性の確保に向けた研究開発拠点形成」の支援を受けたものです．また，本研究成果 (の一部) は，データ活用社会創成プラットフォーム mdx を利用して得られたものです．

参考文献

[1] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models, 2020.

[2] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, **Advances in Neural Information Processing Systems**, Vol. 30. Curran Associates, Inc., 2017.

[3] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. **Advances in neural information processing systems**, Vol. 33, pp. 1877–1901, 2020.

[4] 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, 2023.

[5] Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. Zero: Memory optimizations toward training trillion parameter models. In **SC20: International Conference for High Performance Computing, Networking, Storage and Analysis**, pp. 1–16. IEEE, 2020.

[6] Vijay Anand Korthikanti, Jared Casper, Sangkug Lym, Lawrence McAfee, Michael Andersch, Mohammad Shoeybi, and Bryan Catanzaro. Reducing activation recomputation in large transformer models. **Proceedings of Machine Learning and Systems**, Vol. 5, , 2023.

[7] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In **International conference on machine learning**, pp. 2790–2799. PMLR, 2019.

[8] Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In **International Conference on Learning Representations**, 2022.

[9] Cheng Chen, Yichun Yin, Lifeng Shang, Xin Jiang, Yujia Qin, Fengyu Wang, Zhi Wang, Xiao Chen, Zhiyuan Liu, and Qun Liu. bert2BERT: Towards reusable pretrained language models. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, **Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)**, pp. 2134–2148, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[10] Yu Pan, Ye Yuan, Yichun Yin, Jiaxin Shi, Zenglin Xu, Ming Zhang, Lifeng Shang, Xin Jiang, and Qun Liu. Preparing lessons for progressive training on language models. In **Proceedings of the AAAI Conference on Artificial Intelligence**, Vol. 38, pp. 18860–18868, 2024.

[11] Vladislav Lialin, Sherin Muckatira, Namrata Shivagunde, and Anna Rumshisky. ReloRA: High-rank training through low-rank updates. In **The Twelfth International Conference on Learning Representations**, 2024.

[12] Jiawei Zhao, Zhenyu Zhang, Beidi Chen, Zhangyang Wang, Anima Anandkumar, and Yuandong Tian. Galore: Memory-efficient LLM training by gradient low-rank projection. In **Forty-first International Conference on Machine Learning**, 2024.

[13] Sheng Shen, Pete Walsh, Kurt Keutzer, Jesse Dodge, Matthew Peters, and Iz Beltagy. Staged training for transformer language models. In **International Conference on Machine Learning**, pp. 19893–19908. PMLR, 2022.

[14] Bo Chang, Lili Meng, Eldad Haber, Frederick Tung, and David Begert. Multi-level residual networks from dynamical systems view. In **International Conference on Learning Representations**, 2018.

[15] Chengyu Dong, Liyuan Liu, Zichao Li, and Jingbo Shang. Towards adaptive residual network training: A neural-ODE perspective. In Hal Daumé III and Aarti Singh, editors, **Proceedings of the 37th International Conference on Machine Learning**, Vol. 119 of **Proceedings of Machine Learning Research**, pp. 2616–2626. PMLR, 13–18 Jul 2020.

[16] Changlin Li, Bohan Zhuang, Guangrun Wang, Xiaodan Liang, Xiaojun Chang, and Yi Yang. Automated progressive learning for efficient training of vision transformers. In **CVPR**, 2022.

[17] James O'Neill, Greg V. Steeg, and Aram Galstyan. Layer-wise neural network compression via layer fusion. In Vineeth N. Balasubramanian and Ivor Tsang, editors, **Proceedings of The 13th Asian Conference on Machine Learning**, Vol. 157 of **Proceedings of Machine Learning Research**, pp. 1381–1396. PMLR, 17–19 Nov 2021.

[18] Anton Lozhkov, Loubna Ben Allal, Leandro von Werra, and Thomas Wolf. Fineweb-edu, May 2024.

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

[20] 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, Berlin, Germany, August 2016. Association for Computational Linguistics.

[21] Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. **Communications of the ACM**, Vol. 64, No. 9, pp. 99–106, 2021.

[22] Yonatan Bisk, Rowan Zellers, Ronan bras, Jianfeng Gao, and Choi Yejin. Piqa: Reasoning about physical commonsense in natural language. **Proceedings of the AAAI Conference on Artificial Intelligence**, Vol. 34, pp. 7432–7439, 04 2020.

[23] Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In Anna Korhonen, David Traum, and Lluís Màrquez, editors, **Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics**, pp. 4791–4800, Florence, Italy, July 2019. Association for Computational Linguistics.

[24] Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge, 2018.

[25] Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors, **Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing**, pp. 2381–2391, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[26] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca, 2023.

[27] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. **Advances in Neural Information Processing Systems**, Vol. 36, , 2024.

表 4　モデル設定および手法ごとの固有ハイパーパラメータを Table 2 に示す．バッチサイズはトークン単位で指定している．

	学習率	更新回数	バッチサイズ	訓練トークン量	FLOPs
368M					
標準事前学習	5e-4	20K	360K	7B	1.63e+19
ReLoRA	5e-4	43K	360K	13B	1.63e+19
GaLore	1e-2	20K	360K	7B	1.63e+19
STEP-2stages	5e-4	33K	360K	11B	1.63e+19
680M					
標準事前学習	4e-4	20K	688K	14B	5.55e+19
ReLoRA	4e-4	43K	688K	23B	5.55e+19
GaLore	1e-2	20K	688K	14B	5.55e+19
STEP-2stages	4e-4	33K	688K	21B	5.55e+19
1.2B					
標準事前学習	3e-4	20K	1179K	24B	1.73e+20
ReLoRA	3e-4	43K	1179K	43B	1.73e+20
GaLore	1e-2	20K	1179K	24B	1.73e+20
STEP-2stages	3e-4	33K	1179K	39B	1.73e+20
STEP-3stages	3e-4	45K	1179K	53B	1.73e+20

A　FLOPs の計算方法

FLOPs を C，埋め込み層以外のパラメータ数を N，学習に使用する総トークン数を T とする．このとき，$C \approx 6NT$ が成り立つ．係数 6 は 1 ステップあたりの浮動小数点演算の回数を表し，順伝播に 2 回．逆伝播などその他の計算に 4 回の浮動小数点演算が必要となる．したがって，学習対象のパラメータ数を $N_{\text{trainable}}$，学習対象ではないパラメータ数を $N_{\text{untrainable}}$ とすると，FLOPs は $C \approx (6N_{\text{trainable}} + 2N_{\text{untrainable}})T$ と計算できる．

B　事前学習の詳細な設定

表 5　項 4.1 の事前学習の実験において，全モデルサイズに共通する学習設定の一覧．

訓練設定	設定した値
共通設定	
最適化器	AdamW ($\beta_1 = 0.9, \beta_2 = 0.95$)
重み減衰	0.1
学習率スケジューラ	cosine
ウォームアップステップ数	1000
文長	1024
ReLoRA の設定	
LoRA ランク	128
ReLoRA リセットステップ	5000
再開始時のウォームアップステップ数	500
GaLore の設定	
GaLore ランク	128
射影間隔の更新	200
Galore スケール	0.25

語彙は GPT-2 のものを使用した．すべてのモデル設定（368M，680M，1.2B）に共通する学習設定を Table 5 に，各モデル設定に固有の学習設定を Table 4 に示す．ReLoRA [11] および GaLore [12] については，論文で報告されているハイパーパラメータ設定に従った．

学習率スケジューラの再初期化　STEP の手順 2 で層を追加する際，既存の層に対して PET を適用することで最適化器の状態を初期化する．さらに手順 4 では，新しい層のより効率的な学習を促進するため，学習率を手順 1 で使用した値まで再度上昇させる．

C　指示学習の詳細な設定

表 6　項 4.2 における指示調整の学習設定

訓練設定	設定した値
最適化器	AdamW ($\beta_1 = 0.9, \beta_2 = 0.95$)
学習率	0.0001
学習率スケジューラ	cosine
ウォームアップステップ数	100
エポック数	2

指示調整で使用した学習設定を Table 6 に示す．表 4.2 に示した 3 つの指示調整済みモデルは，いずれも全パラメータのチューニングを行っている．