非構造的枝刈り手法の構造的手法への変換によるLLMの軽量化

小林　亮太*，平川　翼，山下　隆義，藤吉　弘亘 (中部大学)

LLM Compression through Conversion of Unstructured Pruning to Structured Pruning Methods

Ryota Kobayashi, Tsubasa Hirakawa, Takayoshi Yamashita, Hironobu Fujiyoshi (Chubu University)

１．はじめに

大規模言語モデル（LLM）は，様々な自然言語処理タスクにおいて高い性能を示しているが，その計算コストの高さが障壁となっている．このため，モデルの軽量化は不可欠であり，枝刈り（pruning）はその有力な手法の一つである．枝刈りは非構造的枝刈りと構造的枝刈りに大別できる．前者は個々の重みを独立に削除できるため高い圧縮率が得られるが，不規則にスパースな行列演算により推論速度の改善が困難である．後者はニューロン単位での削除により，ハードウェア上での高速化が可能である一方で，重要度評価の自由度が低く，性能劣化が大きくなりやすい．本研究では，非構造的手法で得られた精緻な重要度評価を構造的枝刈りに応用することで，精度と高速化の両立を図る．具体的には，既存の非構造的手法（SNIP, ReFer-L1，ReFer-SVD，AFR）をスコアベースで構造的に変換するアプローチを提案し，Llama-3-8B に対して評価を行う．

２．関連手法

大規模言語モデルの軽量化において，学習前に枝刈りを行う foresight pruning 手法は計算効率の観点から重要である．本節では，foresight pruning 手法と，本研究で構造的枝刈りの対象とする代表的な非構造的 foresight pruning 手法について述べる．

<2・1>Foresight pruning 手法　ニューラルネットワークの枝刈りは適用タイミングによって学習後枝刈り [5, 6, 7], 学習中枝刈り [1, 2, 8, 9]，学習前枝刈り (foresight pruning)[12, 13, 15] に分類される．Foresight pruning は，従来の学習・枝刈り・再学習の反復プロセスを経ずに，学習前にスパースネットワークを特定することを目的とし [3, 4, 11]，lottery ticket 仮説 [3] に基づいている．

初期の foresight pruning 手法である Single-shot Network Pruning (SNIP) [15] は目的関数に対するパラメータの感度を重みの評価値とする手法である．GraSP [13] はネットワーク内の勾配フローの維持を目的とし，SynFlow [12] はレイヤー崩壊問題を解決するために提案された．しかし，これらはスクラッチモデル用であり，事前学習済みモデルでの有効性は十分に議論されていない．

事前学習済みモデル向けの手法においては，事前学習済みモデルは多くのパラメーターが既に収束しているため，勾配ベースの評価は困難である [10, 14]．この課題に対し，SNIP-Mag [14] や ReFer [16] などの手法が提案されている．

<2・2>既存の Foresight Pruning 手法　SNIP[15] は，目的関数に対するパラメータの感度を重みの評価値として算出する代表的な foresight pruning 手法である．n 番目の重みパラメータ θ_n に対する評価値 $S_{SNIP}(\theta_n)$ は式 (1) のように定義される．

$$S_{SNIP}(\theta_n) = \left| \frac{\partial \mathcal{L}(\theta)}{\partial \theta_n} \theta_n \right| \tag{1}$$

ここで，$\mathcal{L}$ は目的関数である．しかし，事前学習済みモデルでは多くのパラメータが既に収束しているため，$\frac{\partial \mathcal{L}(\theta)}{\partial \theta_n} = 0$ となり，勾配ベースの評価では適切な重要度評価が困難である．この課題に対し，事前学習済みモデルの特徴表現の維持に着目した手法 ReFer[16] が提案されている．ReFer は，特徴表現の維持を重視し，L1 ノルムや SVD（特異値分解）を用いて各層の重要度を評価する．L1 ノルムを用いる ReFer (以下，ReFer-L1) は，各層の特徴量の L1 ノルムへの重みの感度を評価することで特徴表現を維持する．l 層目の特徴量 F^l に対する L1 ノルム $||F^l||_1$ への重みの感度を評価する．重み θ^l_n に摂動 δ を与えた際の $||F^l||_1$ の変化量 $\Delta||F^l||_1$ は式 (2) のように表される．

$$\Delta||F^l||_1 \approx -\frac{\partial ||F^l||_1}{\partial \theta^l_n} \theta^l_n \tag{2}$$

全 L 層について総和することで，L1 ベース手法の評価基準を式 (3) のように定義する．

$$S_{ReFer-L1}(\theta_n) = \sum_{l=1}^{L} \left| \frac{\partial ||F^l||_1}{\partial \theta^l_n} \theta^l_n \right| \tag{3}$$

SVD ベースの ReFer (以下，ReFer-SVD) は，L1 ノルムの代わりに SVD（特異値分解）による特異値の平均を用いる．l 層目の出力特徴量 F^l を SVD により $F^l = U\ \Sigma\ V^T$ に分解し，特異値の平均 F^l_{SVD} を式 (4) で算出する．

$$F^l_{SVD} = \frac{1}{N} \sum_{i=1}^{N} \sigma_i \tag{4}$$

ここで，σ_i は特異値，N はサンプル数である．SVD ベース手法の評価基準は式 (5) のように定義される．

$$S_{ReFer-SVD}(\theta_n) = \sum_{l=1}^{L} \left| \frac{\partial F^l_{SVD}}{\partial \theta^l_n} \theta^l_n \right| \tag{5}$$

しかし，特徴表現の維持のみに焦点を当てた手法では，下流タスクへの適応に必要な重みを誤って削除する可能性がある．この問題を解決するため，Adaptive Feature Retention (AFR) [17] は，ReFer-SVD と SNIP を組み合わせることで，事前学習で獲得した特徴空間の維持と下流タスクへの適応を両立する手法である．それぞれを標準化した後に加算することで，式 (6) のように定義される．

$$S_{AFR}(\theta_n) = \hat{S}_{ReFer-SVD}(\theta_n) + \hat{S}_{SNIP}(\theta_n) \tag{6}$$

ここで，$\hat{S}_{ReFer-SVD}$ と $\hat{S}_{SNIP}$ はそれぞれ標準化後の ReFer-SVD と SNIP の評価値である．AFR によって，単一の評価基準では困難であった事前学習知識の保持と下流タスク適応の両立が可能となる．

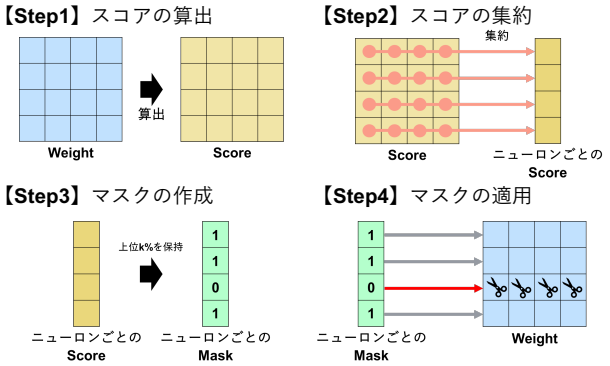

Fig. 1　Score Calculation and aggregation for structured pruning

３．非構造的枝刈りから構造的枝刈りへの変更

本研究では，既存の非構造的 foresight pruning 手法を構造的手法に適用し，LLM の軽量化における有効性を検証する．従来の非構造的枝刈りは個々の重みを独立に削除するため高い圧縮率を達成できるが，不規則なスパースパターンを生成するため，実際の推論高速化が困難である．一方，構造的枝刈りはニューロン単位でパラメータを削除するため，推論高速化を実現できる．

<3・1>構造的枝刈りへの変換の実現方法　非構造的手法を構造的手法に変換するため，Fig. 1 に示す 4 つのステップからなるスコアベースアプローチを採用する．

まず，Step1 として既存の非構造手法（AFR, ReFer-SVD, ReFer-L1, SNIP）を用いて，重み行列の各要素に対して個別にスコアを算出する．この段階では，重み行列と同じサイズのスコア行列が生成される．次に，Step2 においてスコアをニューロンごとに集約する．具体的には，各ニューロンに対応する列・行方向のスコアを平均化することで，ニューロン単位での重要度を表す 1 次元のスコアベクトルを生成する．この集約により，個々の重みレベルでの評価をニューロンレベルでの評価に変換できる．Step3 では，集約されたニューロンごとのスコアに基づいてマスクを作成する．上位 k% のスコアを持つニューロンを保持対象 (マスク値 1)，それ以外を削除対象 (マスク値 0) として二値のマスクベクトルを生成する．最後に，Step4 でマスクを元の重み行列に適用する．マスク値が 0 のニューロンに対応する列・行全体を削除することで，ニューロン単位での構造的枝刈りが可能となる．

この手法により，既存の非構造手法の重要度評価機能を保持しつつ，実装が簡素で推論高速化が可能な構造的枝刈りの利点を享受できる．

４．実験

構造的枝刈りに変換した手法の有効性を検証するために Llama-3-8B モデルに対して枝刈りを実施し，複数の下流タスクでの性能を比較した．また，構造的枝刈りによる推論速度の改善効果も評価する．

<4・1>実験設定　実験では，Llama-3-8B を対象モデルとし，20%,50% の枝刈り率を適用した．比較手法として，AFR, ReFer-L1, ReFer-SVD, SNIP の構造的版を用いた．評価タスクには，WinoGrande, PIQA, HellaSwag, ARC-e, ARC-c を使用し，Feed-Forward Network(FFN) のみを枝刈り対象とした．評価指標には accuracy を用い，LM-Evaluation-Harness を使用して評価を行った．

推論速度の測定は，NVIDIA A6000 GPU 1 基を用いて，バッチサイズ 1 での推論時間を計測した．同時に，パラメータサイズと VRAM 使用量についても比較を行った．

Table 1　Accuracy of structural pruning methods (%)

Method	WinoG	PIQA	HellaS	ARC-e	ARC-c	Ave
Sparsity 0%						
Llama3-8B	70.93	80.96	79.17	45.00	53.16	65.84
Sparsity 20%						
AFR	59.19	63.17	53.94	43.27	29.27	49.77
ReFer-L1	51.78	52.50	26.29	24.83	26.11	36.30
SNIP	50.83	52.61	26.30	24.49	26.62	36.17
ReFer-SVD	49.41	52.50	26.31	25.08	26.71	36.00
Sparsity 50%						
AFR	52.25	52.88	29.29	29.92	26.02	38.07
ReFer-L1	49.88	52.72	27.34	28.32	27.05	37.07
SNIP	49.57	51.41	27.31	25.29	25.68	35.57
ReFer-SVD	49.25	51.63	25.89	24.62	26.45	35.85

Table 2　Accuracy of Unstructural pruning methods (Sparsity 50%)

Method	WinoG	PIQA	HellaS	ARC-e	ARC-c	Ave
AFR	60.62	67.46	50.64	55.35	32.94	53.40
ReFer-L1	57.06	67.25	46.42	50.04	28.84	49.92
SNIP	59.12	65.56	45.46	50.17	30.55	50.17
ReFer-SVD	60.46	66.92	49.13	52.06	29.10	51.53

<4・2>構造的枝刈りの精度評価　構造的枝刈り手法の性能比較結果を Table 1 に示す．20% 枝刈りにおいて，AFR が平均 accuracy 49.77% で最良の性能を示し，ReFer-L1（36.30%），ReFer-SVD（36.00%），SNIP（36.17%）を大幅に上回った．50% 枝刈りでも同様の傾向が見られ，AFR が 38.07% で最良を維持し，ReFer-L1（37.07%），SNIP（35.57%），ReFer-SVD（35.85%）が続いた．

注目すべきは，AFR と他手法の性能差が枝刈り率によって大きく変化することである．20% 枝刈りでは AFR が他手法を約 13 ポイント上回っているが，50% 枝刈りではその差は約 1-3 ポイントに縮小した．これは，高い枝刈り率において各手法の性能が収束する傾向を示しており，極端な軽量化条件下では手法間の差異が相対的に小さくなることを示唆している．

タスク別の分析では，全手法で HellaSwag と ARC タスクにおける大幅な性能低下が確認された．一方，WinoGrande と PIQA では比較的性能が保持されており，タスクの性質による枝刈り耐性の違いが明確に現れている．

<4・3>非構造的手法との比較　構造的枝刈りの性能を評価するため，同一の手法を非構造的枝刈りとして適用した場合との比較を行った．Table 2 に非構造的枝刈りの結果を示す．AFR の非構造的枝刈りでは平均 accuracy 53.40% を達成し，構造的枝刈り（38.07%）と比較して約 15 ポイントの性能差が見られた．この性能低下は，構造的枝刈りの制約に起因するものである．非構造的枝刈りでは個々の重みを独立に評価・削除できるため，重要な重みを細かく保持できる．一方，構造的枝刈りではニューロン単位での削除が必要なため，重要な重みが含まれるニューロンであっても，そのニューロン内の他の重みの影響により削除される場合がある．しかし，AFR は構造的変換後も他手法に対する相対的優位性を保持している．非構造的枝刈りにおける AFR の優位性が，構造的枝刈りにおいても維持され

Table 3　Comparison of parameters size, memory usage, and inference speed-up on GPU

Model	Parameters	VRAM	Speed-up GPU
Llama3-8B	8.03B	16.06GB	1.0x
+AFR	5.21B	10.42GB	1.57x

ており，AFR の重要度評価機能の有効性が確認された．構造的枝刈りによる精度低下は見られるものの，実装の簡素化と実際の推論高速化という実用上の大きなメリットが期待される．この実用性について，次節で評価する．

<4・4>推論速度の評価　構造的枝刈りによる推論効率の改善を評価した．Table 3 に示すように，50% 枝刈り後の AFR モデルは，元モデル（8.03B パラメータ）と比較してパラメータサイズが 5.21B に削減され，約 35% の圧縮を達成した．VRAM 使用量も 16.06GB から 10.42GB に減少し，同様に約 35% の削減効果が得られた．推論速度については 1.57 倍の高速化を実現し，構造的枝刈りの実用的価値を実証した．

５．まとめ

本研究では，既存の非構造的枝刈り手法を構造的手法に変換した手法を提案し，LLM に対する有効性を検証した．Llama-3-8B を用いた枝刈り実験の結果，構造的変換後も AFR が最良の性能を示し，他手法に対する優位性を保持することを確認した．また，パラメータサイズを約 35% 削減，VRAM 使用量を約 35% 削減，推論速度を 1.57 倍高速化することを実証し，精度と実用性のバランスが取れた軽量化手法としての有効性を示した．今後の課題として，Attention モジュールへの適用，平均以外の集約方法の検討，ワンショット以外の枝刈り手法への拡張が挙げられる．

文　献

[1] X. Dai, et al.: "A neural network synthesis tool based on a grow-and-prune paradigm", IEEE 2019, pages 1487-1497, 2019.

[2] T. Dettmers and L. Zettlemoyer : "Sparse networks from scratch: Faster training without losing performance." ArXiv 1907.04840, 2019.

[3] J. Frankle and M. Carbin.: "The lottery ticket hypothesis: Finding sparse, trainable neural networks.", ICLR 2019, 2019.

[4] S. Girish, et al.: "The lottery ticket hypothesis for object recognition.", CVPR, pages 762 – 771, 2021.

[5] S. Han, et al.: "Learning both weights and connections for efficient neural network.", NeurIPS 2015, 2015.

[6] B. Hassibi, et al.: "Wolff. Optimal brain surgeon and general network pruning.", IEEE, pages 293 – 299 vol.1, 1993.

[7] Y. LeCun, et al.: "Optimal brain damage.", NeurIPS, 1989.

[8] T. Lin, et al.: "Dynamic model pruning with feedback.", ArXiv 2006.07253, 2020.

[9] S. Liu, et al.: "Sparse training via boosting pruning plasticity with neurogeneration.", 2021.

[10] S. Liu, et al.: "Sparsity may cry: Let us fail (current) sparse neural networks together!", ICLR 2023, 2023.

[11] A. Morcos, et al.: "One ticket to win them all: generalizing lottery ticket initializations across datasets and optimizers.", NeurIPS, 2019.

[12] H. Tanaka, et al.: "Pruning neural networks without any data by iteratively conserving synaptic flow.", NeurIPS, pages 6377 – 6389, 2020.

[13] C. Wang, et al.: "Picking winning tickets before training by preserving gradientflow.", ICLR 2020, 2020.

[14] H. Kohama, et al.: "Single-shot pruning for pre-trained models: Rethinking the importance of magnitude pruning.", ICCV Workshops, pages 1433 – 1442, 2023.

[15] N. Lee, et al.: "SNIP: Single-shot Network Pruning based on Connection Sensitivity", International Conference on Learning Representation (ICLR) 2019, 2019.

[16] H. Kohama, et al.: "ReFer: Single-shot Foresight Pruning for Retaining Feature Representation", Arxiv, 2024.

[17] T. Nitta, et al: "AFR: Adaptive Feature Retention for Single-shot Foresight Pruning", Arxiv, 2025.