# ResNet-BKプロジェクトのための戦略的研究開発レポート: SOTA（最先端技術）との融合によるO(N)アーキテクチャの実現

## 序章: ResNet-BKの現状とSOTA研究の交差点

本レポートの目的: 提示されたResNet-BKプロジェクトの内部レビューおよび研究タスクマップ [1] は、Transformerに依存しないO(N)言語モデルの実現に向けた、野心的かつ体系的な計画を示している。そこでは、BKコア、MoE、量子化、そしてKoopman作用素といった先進的な要素が組み込まれている。本レポートの目的は、このロードマップ（特にStep 5〜7）を、2024年から2025年にかけて発表された最先端（SOTA）のAI研究（Mamba, Structured State Space Models (SSMs), Triton最適化, Koopman理論の応用など）の文脈で徹底的に分析することである。これにより、既存タスクの妥当性を検証し、研究開発のギャップを特定し、プロジェクトの成功確率を最大化するための戦略的裏付けと実行可能な提言を提供する。

初期の核心的所見: ResNet-BKの核となる「BKコア」（三対角行列の逆行列計算 [1]）は、単なる独自のアルゴリズムではなく、O(N)シーケンスモデリングのSOTAとして広く認められているMamba [2] やStructured State Space Models (SSM) [4] の数学的基盤と表裏一体の関係にある可能性が極めて高い。数値線形代数の分野では、三対角行列の逆行列と半可分行列（Semiseparable Matrix）との間に密接な関係があることが知られている [6]。この関連性の解明は、ResNet-BKがSSMコミュニティの膨大な知見を活用し、開発を飛躍的に加速させるための最初のブレークスルーとなる。

## 第1部: アーキテクチャの核心的アイデンティティの確立 - "BKコア" vs. Mamba (SSM)

### 1.1. O(N)シーケンスモデリングの二大潮流

Transformerアーキテクチャは、その自己注意（Self-Attention）メカニズムにより、言語モデリングに革命をもたらした [2]。しかし、Attentionはシーケンス長$N$に対して$O(N^2)$の計算量とメモリ消費を必要とするため、特に100ページ文書の要約のような超長文コンテキストの処理において、現実的でない計算コストの壁に直面している [2]。

この$O(N^2)$問題を解決するため、Structured State Space Models (SSMs) [4] およびその発展形であるMamba [3] が、SOTAのO(N)アーキテクチャとして台頭した。これらのモデルは、RNN（Recurrent Neural Networks）の持つ効率的な逐次計算（推論時）と、CNN（Convolutional Neural Networks）の持つ効率的な並列計算（学習時）の特性を両立させ、さらに長距離依存性を捉える能力を持つ [3]。

## 1.2. Mamba / S4 (SSM) の数学的基盤

SSMの理論的基盤は、連続時間システムを記述する以下の状態空間方程式にある 5。

$$x'(t) = Ax(t) + Bu(t), y(t) = Cx(t) + Du(t)$$

ここで、$x(t)$は状態ベクトル、$u(t)$は入力、$y(t)$は出力、そして$A, B, C, D$はシステムを定義する行列である。
S4モデル [5] の革新は、この方程式を効率的に計算するため、状態行列$A$をHiPPO (High-order Polynomial Projection Operator) 理論 [8] に基づいて特定の構造（例：対角＋低ランク補正 [10]）で初期化・パラメータ化した点にある。これにより、システムは効率的な畳み込みカーネル（例：Cauchyカーネル [5]）として表現でき、学習時の並列計算が可能となった。

Mamba [2] は、このSSMのアイデアをさらに発展させたものである。S4の$A, B, C$行列が静的（時間不変）であるのに対し、Mambaはこれらを入力データに応じて動的に変化させる「選択的メカニズム (Selective Scan)」 [11] を導入した。さらに、Triton言語 [12] を用いたハードウェア最適化（カーネル融合）を徹底することで、IOバウンド（メモリ律速）な処理をComputeバウンド（計算律速）に変え、Transformerを凌駕するスループットと性能を達成した [4]。

## 1.3. BKコアの数学的基盤とSSMとの接続

ResNet-BKのBKコアは、「三対角行列の逆行列の対角要素」を求めるO(N)アルゴリズムに基づいている [1]。これは、順方向と逆方向の漸化式を用いており、数値線形代数におけるThomasアルゴリズ

ム（三対角行列の高速な解法）[14] の一種、あるいはその変種であると推察される。

ここで、プロジェクトの独自性とSOTA研究との間に、決定的な理論的接続が存在する。

1. ResNet-BKのBKコアは「三対角行列の逆行列」を扱っている[1]。
2. 一方、SSMのS4モデルが用いるHiPPO行列や関連する構造化行列は、「半可分行列（Semiseparable Matrix）」またはそれに類する構造を持つことが知られている[6]。
3. 数値線形代数の古典的な結果として、「（既約な）三対角行列の逆行列は、半可分行列である」という定理が存在する[6]。

この数学的事実は、ResNet-BK（三対角行列の逆行列の視点）とS4/SSM（半可分行列＝状態空間の視点）が、同じ**O(N)**の数学的対象を異なる側面から定式化している可能性が極めて高いことを示している。

この接続は、ResNet-BKプロジェクトにとって極めて戦略的な意味を持つ。プロジェクトの独自性を「三対角行列」と定義することは可能だが、SSMコミュニティが過去数年間で蓄積した膨大な知見（例：HiPPO初期化による長距離依存性の獲得[9]、ZOHやBilinear変換による安定した離散化手法[10]）を無視することは非効率である。BKコアの行列の初期化や、現在グリッドサーチで探索されているGRAD_BLENDパラメータ[1]の安定化において、SSMの理論的知見を積極的に導入すべきである。

### 1.4. Mamba-2と「構造化状態空間双対性 (SSD)」

2024年に発表されたMamba-2[18]は、SSMとAttentionの間の理論的関係性を「Structured State Space Duality (SSD)」[19]として明示的に定式化した。これは、SSMがAttentionのメカニズム（Q, K, Vの生成）と類似の操作として解釈できることを示している[19]。

このSOTAの動向は、ResNet-BKが「SSMファミリーの高性能亜種」として自身のアーキテクチャを再定義し、Mamba-2[12]のような最新の成果から学ぶことが可能であることを強く示唆している。

## 第2部: ハードウェアの限界突破 (Step 5)

### 2.1. タスクA-01/A-02 (CUDA/Tritonカーネル) へのSOTA解: "FlashRNN" に学ぶ漸化式計算の最適化

プロジェクトの課題: ResNet-BKのパフォーマンスボトルネックは、BKコア内部の「順方向と逆方向の漸化式計算」にある[1]。この種の計算は本質的に逐次的であり、GPUの得意とする大規模な並列性に反するため、メモリ律速（IO-bound）になりやすい。

**Mamba**の解法: Mambaは、この逐次的な漸化式（Scan）を並列化するための「Parallel Scan」アルゴリズム[11]を採用した。さらに、Triton[12]を用いて、Scan操作、活性化関数、畳み込みなどを単一のカーネルに「融合（fusion）」することで、GPUのHBM（高帯域幅メモリ）との間のデータ往復を最小限に抑え、高速なSRAM（共有メモリ）を最大限に活用する[13]。これにより、処理はメモリ律速から計算律速（Compute-bound）へと転換され、劇的な速度向上を達成した。

**BK**コアへの最適解: タスクA-01（CUDA）およびA-02（Triton）[1]にとって、このMambaのアプローチは直接的な指針となる。さらに、2024年12月の論文「FlashRNN: Optimizing traditional RNNs on modern hardware」[21]は、BKコア（一種のRNN）の最適化に特化した、より具体的なSOTA解を提供する。

FlashRNNの核心技術は、RNNの漸化式計算（行列積）と要素ごとの操作（pointwise operations、例：活性化関数）を、**単一の永続カーネル（persistent kernel）に融合（fusing）**することにある[21]。さらに重要なのは、漸化式の重み（BKコアの場合、三対角行列の要素に相当）を、HBMから毎回ロードするのではなく、**GPU**のレジスタにカスタムキャッシングする戦略である[21]。

行動計画: タスクA-01/A-02は、FlashRNN[21]およびMamba[11]の公開実装をリファレンスとして開始すべきである。

- **A-02 (Triton):** Tritonは、CUDA C++よりも迅速なPoC（概念実証）に適している[23]。OpenAIが主張するように、メモリのCoalescing（合体）やShared Memory管理といった低レベルな最適化の多くをコンパイラが自動化してくれる[25]。
- **A-01 (CUDA):** FlashRNNが示したように、レジスタレベルでの手動キャッシングなど、Tritonの抽象化では不可能なレベルの最適化を行うには、最終的にCUDA C++によるカーネル実装が必要になる[21]。

## 2.2. T4環境におけるメモリ効率の最大化 (Step 5B)

課題: T4 GPU（無料枠）での実験において、OOM（Out of Memory）が深刻な制約となっている[1]。

タスクB-01 (勾配チェックポイント) の高度化:
torch.utils.checkpoint 27 を適用するというユーザーの計画（B-01）は、アクティベーションの保存に必要なメモリを削減するための有効な第一歩である 1。
しかし、標準的なチェックポイントは、順伝播の一部を破棄し、逆伝播時にそれを再計算するため、

計算オーバーヘッドが伴う。NVIDIAの研究 [30] や2024年の最新研究 [31] は、このオーバーヘッドを最小化する「選択的活性化再計算 **(Selective Activation Recomputation)**」[32] を提案している。

この戦略は、Transformerにおいて、メモリ消費が激しく計算コストが低い操作（例：Softmax, Dropout）のアクティベーションのみを保存し、計算コストが高い操作（例：MatMul）は再計算するというものである [32]。

ResNet-BKに適用する場合、ResNetBKBlock [1] 全体をチェックポイントするのではなく、BKCoreFunction [1] の「入力」のみを保存し、Compute集約型である漸化式計算自体は逆伝播時に再計算するという、より洗練された戦略（B-02のtorch.compileのmin-cut partitioner [1] に近い思想）が、メモリ削減と速度オーバーヘッドの最適なトレードオフとなる可能性が高い。

T4環境向けメモリ最適化戦略の比較:
T4環境でのOOMに対処するため、タスクマップ 1 に記載されている技術と、SOTAの技術を比較検討する必要がある。

| 戦略 **(**関連タスク**)** | 引用 | 概要 | メモリ削減効果 | 計算オーバーヘッド | 実装難易度 |
|---|---|---|---|---|---|
| 標準チェックポイント (B-01) | [27] | torch.utils.checkpointでモジュールをラップ | 大 (アクティベーション) | 中 (再計算) | 低 |
| 選択的チェックポイント (B-02) | [30] | 計算グラフを分析し、メモリ消費大/計算小の部分のみ保存 | 中～大 | 低～中 | 高 (要torch.compile) |
| **8-bit** オプティマイザ | [33] | AdamWの代わりにAdamW-8bit等を使用 | 中 (オプティマイザ状態) | ほぼゼロ | 低 (ライブラリ) |
| **ZeRO-Offload** | [36] | 勾配・オプティマイザ状態をCPU/NVMeにオフロード | 甚大 | 高 (PCIe転送) | 中 (DeepSpeed) |

タスクマップにない緊急の対策: 8-bit オプティマイザの導入
T4環境でのOOMの主要因は、(1) モデルパラメータ、(2) アクティベーション、そして (3) オプティマイザ状態 の3つである。特にAdamWオプティマイザ 1 は、パラメータ毎に2つの状態（1次および2次モーメント）をFP32で保持するため、モデルパラメータ自体の2倍のメモリを消費する。
タスクマップ [1] にはこのオプティマイザ状態の削減策が欠落している。2024年-2025年のSOTA研究 [33] は、AdamW-8bitのような8-bitオプティマイザが、32-bit版とほぼ同等の収束性を保ちつつ、オプティマイザが消費するGPUメモリを劇的に削減することを示している [33]。これはQLoRA [40] のようなPEFT（Parameter-Efficient Fine-Tuning）だけでなく、フルパラメータートレーニングにも適用可能である [39]。

この対策は、計算オーバーヘッドがほぼゼロであり、ライブラリの差し替え（train.py [1] の変更）のみで実現できるため、タスクB-01（チェックポイント）よりも優先度が高い、即時実行可能な施策である。

# 第3部: アルゴリズムによる計算効率の追求 (Step 6)

## 3.1. スパース化の深化 (Step 6D) - MoEの先へ

タスク**D-01 (MoE):** ユーザーの計画通り [1]、MoEのRouter（Gating Network）の温度（softmaxのtau）制御と、専門家の使用率を均一化する負荷バランス損失の導入は、MoEの性能を安定させるための標準的かつ重要な改善策である。

タスクD-02 (構造化剪定) vs. D-03 (N:Mスパース化):
タスクマップでは、「構造化剪定」（D-02）と「N:Mスパース化」（D-03）が別タスクとして挙げられている 1。2024-2025年のハードウェア・アウェアな圧縮の観点からは、この2つは明確に優先順位付けされるべきである。

- 構造化剪定（チャネルや層を丸ごと削除 [1]）は、理論的には高速化が見込めるが、ハードウェア・カーネルがその特殊な形状に対応していない限り、実際には速度向上に繋がらないことが多い [43]。
- 非構造化剪定（個々の重みを削除）は、精度維持には優れるが、重み行列が不規則なスパースパターンとなるため、ハードウェア（特にGPU）での加速は絶望的である [44]。
- 半構造化剪定（**Semi-structured Pruning**） [44]、すなわち**N:M**スパース性（例：4つの重みのうち2つだけを非ゼロにする「2:4スパース」）は、NVIDIA AmpereおよびHopperアーキテクチャ [46] においてハードウェア・レベルでサポートされている。

このため、N:Mスパース化は、精度のトレードオフと実際の大幅な速度向上（SpMM: Sparse

Matrix-Matrix Multiplication）を両立する、現実的かつSOTAのアプローチである [47]。したがって、リソースはD-02からD-03 (N:Mスパース化) に集中すべきである。

タスクマップにない新フロンティア: "活性化スパース化"
プロジェクトの現在のスパース化の焦点は、MoE（活性化のルーティング）1 と、重み（Weight）の剪定（D-02, G-04）1 にある。
しかし、2025年のSOTA研究は、「活性化（**Activation**）の**N:M**スパース化」という新しいフロンティアを開拓している [47]。具体的には、arXiv:2508.02128の「Amber Pruner」 [47] などの研究がこれにあたる。

- 重みスパース化: 静的であり、学習後や学習中に一度実行されると、モデルの容量（表現能力）が不可逆的に失われる [49]。
- 活性化スパース化: 動的であり、入力データに依存して計算時に適用される。モデルの全容量を維持しつつ、特定の入力に対して不要な計算のみを（N:Mパターンで）スキップする [48]。

Amber Pruner [47] は、これを「トレーニング不要（training-free）」で実現し、特にLLMのPrefillステージ（入力トークンの初期処理）を高速化する手法として提示している。

ResNet-BKのMoE [1] は、それ自体が「トークン」レベルでの活性化スパース化（選択されたExpertのみが活性化する）である。これに加えて、Amber Prunerの手法をMoEの各Expert内の線形層に適用することで、スパース性を「積層」し、計算量をさらに削減できる可能性がある。これは、タスクマップにはない、新しい研究の方向性を示唆する。

### 3.2. 適応的計算 (Step 6C) - Early Exit

タスク**C-01 (Early-Exit):** 各ResNet-BKブロックの後 [1] に分類器（exit head）を設け、信頼度（confidence）が閾値を超えた場合に計算を打ち切るという計画 [1] は、標準的かつ効果的なアプローチである [50]。

Adaptive Computation Time (ACT) [51] は依然として活発な研究分野である。より高度な手法として、PonderNet [53] のように、終了するレイヤー（ステップ数）自体を確率的な潜在変数として学習するアプローチも存在する。タスクC-01の実装は、推論速度（特に「簡単な」入力に対するレイテンシ）を劇的に改善する可能性があるため、A-01（カーネル最適化）と並行して進める価値が十分にある。

## 第4部: 理論的フロンティア (Step 7) - Koopman作用素による学習理論の革新

## 4.1. Koopman作用素とは何か (タスク F-01)

プロジェクトの野望: ロードマップのStep7は、最終目標として「バックプロパゲーションを Koopman 作用素に基づく物理学的学習に置き換える」[1] ことを掲げている。

理論的背景: Koopman作用素理論 [54] は、非線形な力学系を、その系の「観測量（Observables）」全体の無限次元関数空間における「線形」作用素として記述する手法である [55]。つまり、状態空間では $x_{k+1} = F(x_k)$という非線形な遷移が、観測量空間では $g(x_{k+1}) = K[g(x_k)]$ という線形な遷移として表される [55]。

**ML**への応用: この「非線形の線形化」という強力な特性を利用し、ニューラルネットワークを用いてKoopman作用素の有限次元近似（固有関数）を学習し [57]、システムの未来予測 [61]、時系列分析 [62]、さらには制御 [28] に応用する研究が2024年以降、活発化している。

## 4.2. F-04 "勾配フリー学習" はSOTA研究と完全に一致する

タスクF-04 [1] で示された「バックプロパゲーションを不要にする物理学的学習」という野心的な目標は、単なる空想ではなく、2024年-2025年のSOTA研究と完全に軌を一にしている。

2024年10月（v2: 2025年1月）に発表されたarXiv論文「**Gradient-free training of recurrent neural networks**」[27] は、まさにこの問題を扱っている。

この論文は、「勾配ベースの手法を用いずに（without using gradient-based methods）、リカレントニューラルネットワーク（RNN）の全ての重みとバイアスを構築する計算的アプローチ」[65] を導入している。その核心的手法は、「ランダム特徴ネットワークと**Koopman**作用素理論（具体的にはExtended Dynamic Mode Decomposition, EDMD [68]）の組み合わせ」[65] である。

この研究は、Koopman理論が、BPTT（Backpropagation Through Time）の勾配爆発・消失問題 [67] を根本的に回避する「勾配フリー」な学習パラダイムを提供可能であることを示している。

ResNet-BKのBKコア [1] は、その構造（順方向・逆方向の漸化式）からして、本質的にRNNの一種である。したがって、ResNet-BKは、このKoopman作用素による勾配フリー学習 [65] のアプローチを検証するための、理想的なテストベッドとなる。タスクF-01〜F-04は、プロジェクトの最も革新的な側面として、最優先の基礎研究として推進すべきである。

# 第5部: 圧縮と評価の統合戦略 (Step 4 & G)

## 5.1. 2025年の量子化SOTA (タスク G-05)

**課題:** ユーザーはINT8（FP32の重みを8bit整数にマッピング）を試験的に実装済み [1] であるが、INT4については「校正が難しい」[1] と懸念している。

**SOTAによる回答:** タスクG-05「量子化スキームと校正戦略の評価」[1] にとって、極めて重要な実証研究が2024年11月に発表されている（ACL 2025採択）[69]。

論文「"Give Me BF16 or Give Me Death"? Accuracy-Performance Trade-Offs in LLM Quantization」[69] は、Llama-3.1ファミリー全体を用いて50万件以上の評価を行った、過去最大級の量子化ベンチマークである。その主な結論は以下の通りである [69]:

1. **FP8 (W8A8-FP):** ほぼロスレス（精度低下なし）。
2. **INT8 (W8A8-INT):** 驚くほど良好（精度低下1-3%）。
3. **INT4 (W4A16-INT):** 重み（Weight）のみを4bit化し、活性化（Activation）を16bit（BF16/FP16）に保つW4A16は、期待以上に競争力があり、**8-bit**量子化（**W8A8**）に匹敵する。
4. **展開シナリオ:** T4のような同期セットアップ（バッチ処理）において、**W4A16**が最もコスト効率（速度と精度のバランス）が高い。

**行動計画:** ユーザーの「INT4は難しい」という懸念 [1] は、W4A4（重みも活性化も4bit）の場合には正しいが、SOTA [69] は、より実用的な**W4A16**という選択肢がINT8に匹敵する有力なターゲットであることを示している。タスクG-05では、BKコアとMoE層に対し、INT8 (W8A8) [70] と W4A16 (INT4) [71] の両方を評価すべきである。

## 5.2. 統合的圧縮パイプライン (タスク G-03)

**課題:** タスクマップでは、知識蒸留 (G-02)、剪定 (G-04)、量子化 (G-05) が、個別のタスクとしてリストされている [1]。

**SOTAのアプローチ:** 2024年-2025年のSOTA研究 [72] は、これらの技術を個別に適用するのではな

く、統合的な「圧縮パイプライン」として扱うのが一般的である。

特に、「構造化剪定」と「知識蒸留」は強力な組み合わせである[73]。[72]の研究によれば、剪定（Pruning）によるモデルの精度低下から回復させる際、その性能回復の度合いは、再訓練時の損失関数（標準的なCross-Entropy vs 蒸留損失KL-divergence）に強く依存する。

これは、剪定後のファインチューニングは、タスクG-02で計画されている「知識蒸留」として行うのが最適である可能性が高いことを示唆している。量子化は、一般的にこのパイプラインの最終ステップとして適用される（Post-Training Quantization (PTQ) または Quantization-Aware Training (QAT)）。

行動計画: タスクG-03（「組合せ効果」）[1] は、G-02, G-04, G-05を束ねる最重要タスクとして再定義すべきである。推奨されるSOTAパイプラインは、

1. 剪定: ハードウェア・アウェアなD-03（N:Mスパース化）を実行する。
2. 蒸留: G-02（知識蒸留）を用いて、剪定による精度低下からモデルを回復させる。
3. 量子化: G-05（量子化）を適用し、ターゲット（例：W4A16 [69]）で最終的なモデルを得る。

# 第6部: ResNet-BKプロジェクトへの戦略的提言

提供されたタスクマップ [1] と、2024年-2025年のSOTA研究との詳細な比較分析に基づき、ResNet-BKプロジェクトの価値を最大化し、O(N)アーキテクチャの実現を加速するために、以下の5つの戦略的提言を行う。

提言1: アーキテクチャのアイデンティティ再定義（第1部より）

- 行動: 「Transformerの代替」という孤立したアプローチから、「**Mamba/SSM**ファミリーの高性能亜種（三対角行列ソルバー・ビュー）」へとポジショニングを再定義する。
- 根拠: BKコア（三対角行列の逆行列）とSSM（半可分行列）は数学的に双対の関係にある [6]。この再定義により、S4/Mambaコミュニティの膨大な知見（HiPPO初期化による長距離依存性 [9]、Tritonによるハードウェア最適化 [13]）を、「競合技術の模倣」ではなく「同一理論基盤の適用」として正当かつ迅速に取り込むことが可能となる。

提言2: ハードウェア最適化（**Step 5**）の即時着手と戦略変更（第2部より）

- 行動: タスクA-01 (CUDAカーネル) とB-01 (メモリ) を最優先で着手する。
- 根拠: T4 GPU環境でのOOMを回避し [1]、実験の継続性を確保するために不可欠である。
- 戦略変更**A (A-01):** CUDA/Tritonカーネルは、単なるPoCに留めず、「**FlashRNN**」[21] の設計思想（永続カーネルへの融合、レジスタキャッシング）をBKコアの漸化式に適用する、高度な最適化を目指す。
- 戦略変更**B (Memory):** メモリ対策として、train.py [1] に、計算オーバーヘッドのない「**8-bit**オプ

ティマイザ」[33] を即時導入する。これはチェックポイント (B-01) よりも優先度が高い。
- 戦略変更**C (B-01):** チェックポイントは、標準的なtorch.utils.checkpoint [27] に加え、「選択的活性化再計算」[30] の導入を検討し、計算オーバーヘッドを最小化する。

提言**3:** スパース化（**Step 6**）の**SOTA**への更新（第**3**部より）

- 行動**:** タスクD-02（汎用構造化剪定）は中止し、リソースをD-03（**N:M**スパース化）に集中させる。
- 根拠**:** N:Mスパース性 (例: 2:4) は、NVIDIA GPUでハードウェア・サポートされる、実用的なSOTAである [44]。
- 新規提案**:** MoE (D-01) と並行し、2025年のSOTAである「活性化スパース化」[47] の導入を検討する。これはBKコアの計算量を動的に削減する新しいアプローチとなり得る。

提言**4:** 理論的飛躍（**Step 7**）の具体的な実行（第**4**部より）

- 行動**:** タスクF-04（Koopmanによる物理学習）を、高リスク・高リターンの基礎研究として即時開始する。
- 根拠**:** このタスクは、arXiv:2410.23467「**Gradient-free training of recurrent neural networks**」[65] によって、具体的な研究対象としてSOTAで検証されたばかりである。ResNet-BK（漸化式モデル）は、この「勾配フリー学習」の完璧な実験台となる。

提言**5:** 圧縮（**Step 4**）のパイプライン化（第**5**部より）

- 行動**:** タスクG-03を最上位タスクとし、剪定・蒸留・量子化を統合したパイプラインを設計する。
- 根拠**:** SOTAではこれらの技術は組み合わせて使用される [72]。量子化ターゲットとして、INT8 (W8A8) に加え、SOTAで有効性が示された**W4A16** [69] を積極的に評価する。

引用文献


1. ResNet-BK 研究タスク.pdf
2. Mamba: The Easy Way - Jack Cook, 11月 16, 2025にアクセス、https://jackcook.com/2024/02/23/mamba.html
3. Mamba: Linear-Time Sequence Modeling with Selective State Spaces - arXiv, 11月 16, 2025にアクセス、https://arxiv.org/pdf/2312.00752
4. From S4 to Mamba: A Comprehensive Survey on Structured ... - arXiv, 11月 16, 2025にアクセス、https://arxiv.org/abs/2503.18970
5. [2111.00396] Efficiently Modeling Long Sequences with Structured State Spaces - arXiv, 11月 16, 2025にアクセス、https://arxiv.org/abs/2111.00396
6. (PDF) On Structured State-Space Duality - ResearchGate, 11月 16, 2025にアクセス、https://www.researchgate.net/publication/396250327_On_Structured_State-Space_Duality
7. A small note on the scaling of symmetric positive definite semiseparable matrices - SciSpace, 11月 16, 2025にアクセス、https://scispace.com/pdf/a-small-note-on-the-scaling-of-symmetric-positive-definite-1udri9ivg5.pdf

8. State-Space Modeling in Long Sequence Processing: A Survey on Recurrence in the Transformer Era - arXiv, 11月 16, 2025にアクセス、https://arxiv.org/html/2406.09062v2
9. State-Space Modeling in Long Sequence Processing: A Survey on Recurrence in the Transformer Era - ResearchGate, 11月 16, 2025にアクセス、https://www.researchgate.net/publication/381404773_State-Space_Modeling_in_Long_Sequence_Processing_A_Survey_on_Recurrence_in_the_Transformer_Era
10. state-space modeling in long sequence processing - ResearchGate, 11月 16, 2025にアクセス、https://www.researchgate.net/publication/381404773_State-Space_Modeling_in_Long_Sequence_Processing_A_Survey_on_Recurrence_in_the_Transformer_Era/fulltext/666bcff5de777205a32c44fc/State-Space-Modeling-in-Long-Sequence-Processing-A-Survey-on-Recurrence-in-the-Transformer-Era.pdf
11. Mamba and S4 Explained: Architecture, Parallel Scan, Kernel Fusion, Recurrent, Convolution, Math - YouTube, 11月 16, 2025にアクセス、https://www.youtube.com/watch?v=8Q_tqwpTpVU
12. state-spaces/mamba: Mamba SSM architecture - GitHub, 11月 16, 2025にアクセス、https://github.com/state-spaces/mamba
13. Comprehensive Breakdown of Selective Structured State Space Model — Mamba (S5). | by Freedom Preetham | Autonomous Agents | Medium, 11月 16, 2025にアクセス、https://medium.com/autonomous-agents/comprehensive-breakdown-of-selective-structured-state-space-model-mamba-s5-441e8b94ecaf
14. Tridiagonal matrix algorithm - Wikipedia, 11月 16, 2025にアクセス、https://en.wikipedia.org/wiki/Tridiagonal_matrix_algorithm
15. Tridiagonal M-matrices with tridiagonal Moore-Penrose inverse - ResearchGate, 11月 16, 2025にアクセス、https://www.researchgate.net/publication/393901434_Tridiagonal_M-matrices_with_tridiagonal_Moore-Penrose_inverse
16. arXiv:2304.06100v2 [math.RA] 28 Feb 2024, 11月 16, 2025にアクセス、https://arxiv.org/pdf/2304.06100
17. CLOSED-FORM EXPRESSION FOR THE INVERSE OF A CLASS OF TRIDIAGONAL MATRICES Sigve Hovda (Communicated by Hua Dai) 1. Motivation. I - Semantic Scholar, 11月 16, 2025にアクセス、https://pdfs.semanticscholar.org/fa2c/e6c4764c012505bfe46d06ea8a140a3e107e.pdf
18. [2405.21060] Transformers are SSMs: Generalized Models and Efficient Algorithms Through Structured State Space Duality - arXiv, 11月 16, 2025にアクセス、https://arxiv.org/abs/2405.21060
19. XAMBA: Enabling Efficient State Space Models on Resource-Constrained Neural Processing Units - arXiv, 11月 16, 2025にアクセス、https://arxiv.org/html/2502.06924v1
20. State Space Duality (Mamba-2) Part I - The Model | Tri Dao, 11月 16, 2025にアクセス、https://tridao.me/blog/2024/mamba2-part1-model/
21. FlashRNN: Optimizing traditional RNNs on modern hardware - arXiv, 11月 16, 2025

にアクセス、https://arxiv.org/html/2412.07752v2
22. Sepp Hochreiter - SciProfiles, 11月 16, 2025にアクセス、https://sciprofiles.com/profile/author/QkI1R1JjdnJvYzEzU1J6WjVvY0pmeW9teHRENmswVXZ3NnkyM1g4ZUsvMD0=?utm_source=mdpi.com&utm_medium=website&utm_campaign=avatar_name
23. Unleashing the Power of Triton: Mastering GPU Kernel Optimization in Python, 11月 16, 2025にアクセス、https://towardsdatascience.com/unleashing-the-power-of-triton-mastering-gpu-kernel-optimization-in-python-160a3f52701e/
24. Learning Triton One Kernel At a Time: Vector Addition | Towards Data Science, 11月 16, 2025にアクセス、https://towardsdatascience.com/learning-triton-one-kernel-at-a-time-vector-addition/
25. Introducing Triton: Open-source GPU programming for neural networks - OpenAI, 11月 16, 2025にアクセス、https://openai.com/index/triton/
26. Fast LLM Inference From Scratch - Andrew Chan, 11月 16, 2025にアクセス、https://andrewkchan.dev/posts/yalm.html
27. Gradient-free training of recurrent neural networks - arXiv, 11月 16, 2025にアクセス、https://arxiv.org/html/2410.23467v1
28. Universal Learning of Nonlinear Dynamics - arXiv, 11月 16, 2025にアクセス、https://arxiv.org/html/2508.11990v1
29. Mini-Sequence Transformer: Optimizing Intermediate Memory for Long Sequences Training, 11月 16, 2025にアクセス、https://arxiv.org/html/2407.15892v2
30. [2205.05198] Reducing Activation Recomputation in Large Transformer Models - arXiv, 11月 16, 2025にアクセス、https://arxiv.org/abs/2205.05198
31. BurstEngine: an Efficient Distributed Framework for Training Transformers on Extremely Long Sequences of over 1M Tokens - arXiv, 11月 16, 2025にアクセス、https://arxiv.org/html/2509.19836v1
32. A Survey on Memory-Efficient Transformer-Based Model Training in AI for Science - arXiv, 11月 16, 2025にアクセス、https://arxiv.org/pdf/2501.11847
33. SANA 1.5: Efficient Scaling of Training-Time and Inference-Time Compute in Linear Diffusion Transformer - arXiv, 11月 16, 2025にアクセス、https://arxiv.org/html/2501.18427v1
34. Visualization of two scenarios where Adafactor updates have different stability., 11月 16, 2025にアクセス、https://www.researchgate.net/figure/sualization-of-two-scenarios-where-Adafactor-updates-have-different-stability_fig3_372918087
35. Effective Quantization of Muon Optimizer States - arXiv, 11月 16, 2025にアクセス、https://arxiv.org/html/2509.23106v1
36. Practical offloading for fine-tuning LLM on commodity GPU via learned subspace projectors, 11月 16, 2025にアクセス、https://arxiv.org/html/2406.10181v1
37. Deep Optimizer States: Towards Scalable Training of Transformer Models Using Interleaved Offloading - arXiv, 11月 16, 2025にアクセス、https://arxiv.org/html/2410.21316v1
38. Scaling Large Language Models with DeepSpeed ZeRO, ZeRO++, and

ZeRO-Offload — A Complete Guide | by Pratish Dewangan | Medium, 11月 16, 2025にアクセス、https://medium.com/@dpratishraj7991/scaling-large-language-models-with-deepspeed-zero-zero-and-zero-offload-a-complete-guide-70d393e311f4
39. Wavelet Meets Adam: Compressing Gradients for Memory-Efficient Training - arXiv, 11月 16, 2025にアクセス、https://arxiv.org/html/2501.07237v3
40. Debate, Train, Evolve: Self-Evolution of Language Model Reasoning - arXiv, 11月 16, 2025にアクセス、https://arxiv.org/html/2505.15734v1
41. On the Duality between Gradient Transformations and Adapters - arXiv, 11月 16, 2025にアクセス、https://arxiv.org/html/2502.13811v1
42. SubTrack++ : Gradient Subspace Tracking for Scalable LLM Training - arXiv, 11月 16, 2025にアクセス、https://arxiv.org/html/2502.01586v3
43. HAPE: Hardware-Aware LLM Pruning For Efficient On-Device Inference Optimization | Request PDF - ResearchGate, 11月 16, 2025にアクセス、https://www.researchgate.net/publication/392703282_HAPE_Hardware-Aware_LLM_Pruning_For_Efficient_On-Device_Inference_Optimization
44. Progressive Binarization with Semi-Structured Pruning for LLMs - arXiv, 11月 16, 2025にアクセス、https://arxiv.org/html/2502.01705v4
45. Progressive Binarization with Semi-Structured Pruning for LLMs - arXiv, 11月 16, 2025にアクセス、https://arxiv.org/html/2502.01705v1
46. Leave it to the Specialist: Repair Sparse LLMs with Sparse Fine-Tuning via Sparsity Evolution - arXiv, 11月 16, 2025にアクセス、https://arxiv.org/html/2505.24037v1
47. Amber Pruner: Leveraging N:M Activation Sparsity for Efficient Prefill in Large Language Models - arXiv, 11月 16, 2025にアクセス、https://arxiv.org/html/2508.02128v1
48. Amber Pruner: Leveraging N:M Activation Sparsity for Efficient Prefill in Large Language Models - arXiv, 11月 16, 2025にアクセス、https://arxiv.org/pdf/2508.02128
49. Lightweight error mitigation strategies for post-training N:M activation sparsity in LLMs - arXiv, 11月 16, 2025にアクセス、https://www.arxiv.org/pdf/2509.22166
50. (PDF) A Survey on Green Deep Learning - ResearchGate, 11月 16, 2025にアクセス、https://www.researchgate.net/publication/356083410_A_Survey_on_Green_Deep_Learning
51. Confident Adaptive Language Modeling | Request PDF - ResearchGate, 11月 16, 2025にアクセス、https://www.researchgate.net/publication/362011497_Confident_Adaptive_Language_Modeling
52. SOLAR 10.7B: Scaling Large Language Models with Simple yet Effective Depth Up-Scaling | Request PDF - ResearchGate, 11月 16, 2025にアクセス、https://www.researchgate.net/publication/382633529_SOLAR_107B_Scaling_Large_Language_Models_with_Simple_yet_Effective_Depth_Up-Scaling
53. Daily Papers - Hugging Face, 11月 16, 2025にアクセス、https://huggingface.co/papers?q=retry%20mechanisms
54. Notes on Koopman Operator Theory - UK Fluids Network, 11月 16, 2025にアクセ

ス、https://fluids.ac.uk/files/meetings/KoopmanNotes.1575558616.pdf
55. Recurrent neural networks and Koopman-based frameworks for temporal predictions in a low-order model of turbulence - arXiv, 11月 16, 2025にアクセス、https://arxiv.org/pdf/2005.02762
56. Modern Koopman Theory for Dynamical Systems | SIAM Review, 11月 16, 2025にアクセス、https://epubs.siam.org/doi/10.1137/21M1401243
57. KoopmanLab: Machine learning for solving complex physics equations - AIP Publishing, 11月 16, 2025にアクセス、https://pubs.aip.org/aip/aml/article/1/3/036110/2910717/KoopmanLab-Machine-learning-for-solving-complex
58. Deep Koopman operator framework for causal discovery in nonlinear dynamical systems, 11月 16, 2025にアクセス、https://arxiv.org/html/2505.14828v1
59. Deep Learning of Koopman Representation for Control - arXiv, 11月 16, 2025にアクセス、https://arxiv.org/pdf/2010.07546
60. Deep Learning-Based Construction of Koopman Observable Functions for Power System Nonlinear Dynamics - IEEE Xplore, 11月 16, 2025にアクセス、https://ieeexplore.ieee.org/document/10689140/
61. [2102.12086] Modern Koopman Theory for Dynamical Systems - arXiv, 11月 16, 2025にアクセス、https://arxiv.org/abs/2102.12086
62. [2312.00137] The Multiverse of Dynamic Mode Decomposition Algorithms - arXiv, 11月 16, 2025にアクセス、https://arxiv.org/abs/2312.00137
63. Koopman Operator Theory in Data-Driven Control - Emergent Mind, 11月 16, 2025にアクセス、https://www.emergentmind.com/topics/koopman-operator-theory
64. Koopman-Nemytskii Operator: A Linear Representation of Nonlinear Controlled Systems, 11月 16, 2025にアクセス、https://arxiv.org/html/2503.18269v2
65. [2410.23467] Gradient-free training of recurrent neural networks - arXiv, 11月 16, 2025にアクセス、https://arxiv.org/abs/2410.23467
66. (PDF) Scientific machine learning - ResearchGate, 11月 16, 2025にアクセス、https://www.researchgate.net/publication/395305012_Scientific_machine_learning
67. Learning & Dynamics 2024-2025 - Neuro group @ FEMTO-ST, 11月 16, 2025にアクセス、https://neuro-team-femto.github.io/learning2425/
68. gradient-free training of recurrent neural networks - arXiv, 11月 16, 2025にアクセス、https://arxiv.org/pdf/2410.23467
69. [2411.02355] "Give Me BF16 or Give Me Death"? Accuracy-Performance Trade-Offs in LLM Quantization - arXiv, 11月 16, 2025にアクセス、https://arxiv.org/abs/2411.02355
70. Doing more with less: LLM quantization (part 2) - Red Hat, 11月 16, 2025にアクセス、https://www.redhat.com/en/blog/doing-more-less-llm-quantization-part-2
71. Achieving Binary Weight and Activation for LLMs Using Post-Training Quantization - ACL Anthology, 11月 16, 2025にアクセス、https://aclanthology.org/2025.findings-acl.459.pdf
72. Constrained Edge AI Deployment: Fine-Tuning vs. Distillation for LLM Compression - arXiv, 11月 16, 2025にアクセス、https://arxiv.org/html/2505.18166v1
73. LLM Pruning and Distillation in Practice: The Minitron Approach - arXiv, 11月 16,

2025にアクセス、https://arxiv.org/pdf/2408.11796
74. LLM Pruning and Distillation in Practice: The Minitron Approach - arXiv, 11月 16, 2025にアクセス、https://arxiv.org/html/2408.11796v1
75. HuangOwen/Awesome-LLM-Compression - GitHub, 11月 16, 2025にアクセス、https://github.com/HuangOwen/Awesome-LLM-Compression
76. A survey of model compression techniques: past, present, and future - Frontiers, 11月 16, 2025にアクセス、https://www.frontiersin.org/journals/robotics-and-ai/articles/10.3389/frobt.2025.1518965/full
77. A Comprehensive Review: Model Compression for Large Language Models (LLMs) | by Doil Kim | Medium, 11月 16, 2025にアクセス、https://medium.com/@kimdoil1211/a-comprehensive-review-model-compression-for-large-language-models-llms-2c0c106d5dbe
78. Contemporary Model Compression on Large Language Models Inference - arXiv, 11月 16, 2025にアクセス、https://arxiv.org/html/2409.01990v1