F-010

文脈情報を活用した適応的時間間隔による Mamba の拡張
Extended Mamba with Context-Aware and Adaptive Time Intervals

武智 莉央 [1)]　小西 卓哉 [1) 2)]　河原 吉伸 [1) 2)]
Rio Takechi　Takuya Konishi　Yoshinobu Kawahara

1　概要

本研究は，効率的な深層学習モデルとして注目される Mamba に対し，文脈に依存して時間間隔を調整する新たな拡張手法を提案する．従来の Mamba では入力系列のみから時間間隔を決定するが，提案手法では内部状態に保持された過去の文脈情報を活用し，時間間隔を適応的に決定する．実験では，Induction Heads タスクとそれを発展させた 2 つの評価タスクを実施し，提案手法の有効性を検証する．

2　はじめに

機械学習は，現代社会における重要な技術基盤として，様々な分野で応用されている．大量のデータからパターンを学び，未知の状況においても予測や意思決定を可能にすることで，従来困難だった高度な問題解決や精密な分析を実現している．また，医療診断や自動運転，画像認識，音声処理，自然言語処理など，幅広い問題に応用され，社会全体の発展に寄与している．

近年，深層学習モデルである Transformer [9] が機械学習の分野に画期的な進展をもたらしている．Transformer は系列データのためのニューラルネットワークであり，自己注意機構を活用することで，長い文脈間の依存関係を正確に捉える能力を備えている．この特性により，自然言語処理における機械翻訳，自動要約，質問応答といったタスクで高い性能を実現している．また，並列計算が可能な構造により大規模データからの学習にも適している．Transformer を発展させた BERT [2] や GPT [7] が，事前学習と微調整を組み合わせることで多くのタスクで採用されている．

一方，Transformer には課題も存在する．その 1 つとして，自己注意機構の計算コストが系列長に対して 2 次的に増大する点が挙げられる．特に，長い系列データを扱う場合，メモリの使用量と計算時間の大きさが実用上大きな問題になる．また，現代では，モデルのパラメータ数を増やすことで性能が向上するというスケーリング則 [5] に基づき大規模なモデルの開発が進められている．モデルのパラメータ数が増加すると学習に必要な訓練量も増加するため，Transformer の計算コストは大規模言語モデルをはじめとする最先端の研究開発における主要な課題の一つとなっている．

この問題を解決するため，近年 Mamba [3] が注目されている．Mamba は，状態空間モデルを応用したニューラルネットワークであり，系列長に対して線形な計算量を達成する構造を採用しており，長い系列データでも効率的に処理可能である．また，計算コストを抑えながら Transformer に匹敵する性能を達成できると報告されている．性能向上のために，パラメータ数が爆発的に増加し続ける現代の深層学習モデルにおいて，Mamba のような効率的なモデルは持続可能な発展を支える重要な役割を果たすと期待されている．

多くの既存研究において Mamba の有効性を示す結果が示されているが，依然として多くの改良の余地が残されていると考えられる．本研究では，特に Mamba の時間間隔を決定する方法に着目する．従来の Mamba では時間間隔を入力系列から決定するが，本研究では入力系列に加え，状態空間モデルの内部状態として表現される文脈情報も活用する拡張を提案する．この拡張した Mamba を Induciton Heads タスクとそれを発展させた 2 つのタスクを通して評価を行い，文脈把握能力を実験的に検証する．

本論文の構成は以下のとおりである．まず第 3 章において，本研究の基礎となる状態空間モデルと Mamba の概要について述べる．第 4 章では，本研究で提案する Mamba の拡張について述べる．第 5 章では，実験内容および実験結果について述べる．最後に第 6 章では，本研究をまとめる．

3　関連研究

3.1　状態空間モデル

状態空間モデル (State Space Model，SSM) [4] は系列データを隠れた内部状態によって記述する数理モデルであり，時系列データの解析や予測を中心に広く用いられている．従来から制御理論や信号処理など様々な分野の問題に応用されているが，近年ニューラルネットワークとの統合により注目を集めている．状態空間モデルは，入力系列に対して内部状態を更新する「状態方程式」と，内部状態から観測データを生成する「観測方程式」によって構成され，一般に次の連続時間形式で表される．

$$\frac{d}{dt}h(t) = Ah(t) + Bx(t) \tag{1}$$

$$y(t) = Ch(t) \tag{2}$$

ここで，式 (1) が状態方程式，式 (2) が観測方程式である．t は時刻を表す変数であり，$h(t) \in \mathbb{R}^N$ は時刻 t の内部状態，$x(t) \in \mathbb{R}$ は入力信号，$y(t) \in \mathbb{R}$ は出力信号である．行列 $A \in \mathbb{R}^{N\times N}, B \in \mathbb{R}^{N\times 1}, C \in \mathbb{R}^{1\times N}$ はそれぞれ状態遷移，入力変換，出力変換のための係数行列である．内部状態に系列の過去の情報を保持することで長期依存関係を考慮した効率的なモデル化を実現することができる．

連続時間形式のままでは扱いづらいため，実用上は離散化されたモデルが利用される．離散化された状態空間モデルを求めるために，まず状態方程式 (1) を解くこと

1)　大阪大学大学院情報科学研究科 Graduate School of Information Science and Technology, The University of Osaka
2)　理化学研究所革新知能統合研究センター Center for Advanced Intelligence Project, RIKEN

で次式を得る．

$$h(t) = e^{At}h(0) + e^{At}\int_0^t e^{-Ar}Bx(t)dr \tag{3}$$

ここで時間間隔 Δ ごとに離散化すると，t 番目の時刻は $t\Delta$ となり，時刻 $t\Delta$ に対応する値を h_t と定義することで，h_t と h_{t+1} は次のように表現することができる．

$$\begin{aligned} h_t &= e^{At\Delta}h(0) + e^{At\Delta}\int_0^{t\Delta} e^{-Ar}Bx(t)dr \\ h_{t+1} &= e^{A(t+1)\Delta}h(0) + e^{A(t+1)\Delta}\int_0^{(t+1)\Delta} e^{-Ar}Bx(t)dr \end{aligned} \tag{4}$$

さらに両辺の差をとることで次式を得る．

$$h_{t+1} - e^{A\Delta}h_t = e^{A(t+1)\Delta}\int_{t\Delta}^{(t+1)\Delta} e^{-Ar}Bx(t)dr \tag{5}$$

ここで，$x(t)$ が時刻 $t\Delta$ から $(t+1)\Delta$ まで定数 x_t であると仮定すると，式 (5) の右辺を次のように変形できる．

$$e^{A(t+1)\Delta}\int_{t\Delta}^{(t+1)\Delta} e^{-Ar}Bx(t)dr = \int_{t\Delta}^{(t+1)\Delta} e^{A((t+1)\Delta-r)}drBx_t \tag{6}$$

$v = (t+1)\Delta - r$ で置換すると

$$\begin{aligned} \int_{t\Delta}^{(t+1)\Delta} e^{A((t+1)\Delta-r)}dr &= -\int_{\Delta}^{0} e^{Av}dv \\ &= A^{-1}(e^{A\Delta} - I) \end{aligned} \tag{7}$$

以上より，離散化された状態空間モデルを次のように表すことができる．

$$h_{t+1} = \bar{A}h_t + \bar{B}x_t \tag{8}$$

$$y_t = Ch_t \tag{9}$$

ただし，y_t は t 番目の出力を表し，$\bar{A}$ と $\bar{B}$ は次式で定義される．

$$\begin{aligned} \bar{A} &= e^{A\Delta} \\ \bar{B} &= A^{-1}(e^{A\Delta} - I)B \end{aligned} \tag{10}$$

なお，式 (9) は式 (8) より次のように書き直すことができる．

$$\begin{aligned} y_t &= Ch_t \\ &= C(\bar{A}'h_{t-1} + \bar{B}'x_{t-1}) \\ &= C(\bar{A}'(\bar{A}'h_{t-2} + \bar{B}'x_{t-2}) + \bar{B}'x_{t-1}) \\ &= C\bar{A}'^2h_{t-2} + C\bar{A}'\bar{B}'x_{t-2} + C\bar{B}'x_{t-1} \\ &\cdots \\ &= C\bar{A}'^t\bar{B}'x_0 + C\bar{A}'^{t-1}\bar{B}'x_1 + \cdots + C\bar{A}'\bar{B}'x_{t-1} + C\bar{B}'x_t \end{aligned} \tag{11}$$

よって t 番目の出力 y_t は以下のように畳み込みで記述できる．

$$y_t = \sum_{i=0}^{t} C\bar{A}^i\bar{B}x_{t-i} \tag{12}$$

この畳み込みの関係式を利用することで，例えば長さ L の入力系列 $x = (x_1, x_2, ..., x_L)^\top \in \mathbb{R}^L$ が与えられたとき，出力系列 $y = (y_1, y_2, ..., y_L)^\top \in \mathbb{R}^L$ を得ることができる．この畳み込みの計算は，高速フーリエ変換により系列長 L に対して $\mathcal{O}(L\log L)$ で実行できる．

3.2 Mamba

Mamba [3] は 3.1 節の状態空間モデルを発展させることで，長距離依存関係を効率的に捉える能力と高い計算効率を引き継ぎつつ，より柔軟で適応的なアプローチを実現する．3.1 節の状態空間モデルでは，モデルのパラメータは入力系列全体にわたって固定されており，系列の特性に応じた適応が難しい．Mamba では，パラメータを入力と時刻に依存して動的に変更することで，この問題を解決する．

まず，Mamba では 3.1 節の離散化された状態空間モデルを表す式 (8) と式 (9) を次のように変更する．

$$h_{t+1} = \bar{A}_t h_t + \bar{B}_t x_t \tag{13}$$

$$y_t = C_t h_t \tag{14}$$

ここで $\bar{A}_t$ と $\bar{B}_t$ は次のように定義される．

$$\bar{A}_t = e^{A\Delta_t} \tag{15}$$

$$\bar{B}_t = A^{-1}(e^{A\Delta_t} - I)\,B_t \tag{16}$$

つまり，式 (9) と式 (10) における B，C，Δ を時刻 t に依存する B_t，C_t，Δ_t にそれぞれ変更する．これら 3 つのパラメータは入力系列 x から次の変換によって得られる．

$$B = s_B(x) \tag{17}$$

$$C = s_C(x) \tag{18}$$

$$\Delta = \tau_\Delta(\theta_\Delta + s_\Delta(x)) \tag{19}$$

ここで s_B，s_C，s_Δ はいずれも線形層，τ_Δ は softplus 関数を要素ごとに作用させる変換を表す．また，θ_Δ は入力系列に依存しない学習可能なパラメータである．これら変換によって得られる B，C，Δ は各時刻における係数行列と時間間隔を並べた列に対応する．

$$B = (B_1, ..., B_t, ..., B_L)^\top \in \mathbb{R}^{L\times N} \tag{20}$$

$$C = (C_1, ..., C_t, ..., C_L)^\top \in \mathbb{R}^{L\times N} \tag{21}$$

$$\Delta = (\Delta_1, ..., \Delta_t, ..., \Delta_L)^\top \in \mathbb{R}^L \tag{22}$$

以上のように，Mamba における状態空間モデルでは時刻 t ごとに異なるパラメータ B_t，C_t，Δ_t を仮定し，それらを入力系列 x に応じて動的に調整することで，入力系列の変化をより細かく捉えることができる．

また，入力系列と出力系列の各要素がベクトルからなる系列に一般化した場合の状態空間モデルも考えることができる．系列の各要素が D 次元のベクトルであるとき，長さ L の入力系列と出力系列はそれぞれ行列 $x \in \mathbb{R}^{L\times D}$ と $y \in \mathbb{R}^{L\times D}$ になる．特に，Mamba ではベクトルの各要素について上記の状態空間モデルを独立に計

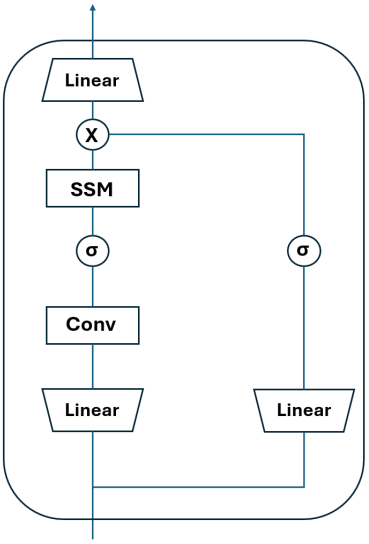

図 1　Mamba の構造

算する．このとき，対応する状態空間モデルの内部状態をまとめると 3 階テンソル $h \in \mathbb{R}^{L\times D\times N}$ として表現できる．また，時間間隔もベクトルの各要素ごとに計算されることで行列 $\Delta \in \mathbb{R}^{L\times D}$ となる．さらに，Mamba ではパラメータ A に構造化された行列を採用することで，実行的なパラメータ数を削減してモデル化する．

係数行列を入力に依存するように変更することで，出力を畳み込みとして表現できず，高速な計算手法を適用できなくなる．この問題に対処するために，Mamba は Parallel Scan と呼ばれる手法を用いて並列計算を実現している．Parallel Scan は再帰的な計算を累積和として表現することで，並列化可能な計算に変換する手法であり，再帰的な計算のボトルネックを解消する．この工夫により，Mamba は適応的な推論能力を保ちながら，高い計算効率を実現している．

Mamba の構造を図 1 に示す．図の下から上にかけて入力を出力へ変換する．Linear は線形層，Conv は系列方向に沿った 1 次元の畳み込み層による変換を表す．σ は活性化関数であり，swish 関数 [8] を用いる．$\times$ は要素積，SSM は状態空間モデルを表す．まず，入力を線形層，畳み込み層，活性化関数で順に変換することでパターンを抽出し，状態空間モデルの入力系列 x を得る．次に，x に上記状態空間モデルを適用することで，出力系列 y を計算する．また，入力を別の線形層と活性化関数で変換しておき，その結果と y との要素積をとり，最後に線形層によって変換することで出力を得る．この一連の変換を Mamba の一つの層とみなし，同様の層を連結することでモデル全体を構成する．

4　提案手法

本章では，Mamba の時間間隔を拡張する手法を提案する．従来の Mamba と区別するため，以降では提案手法を extended Mamba (exMamba) と呼称する．

まず，時間間隔に注目する理由を説明する．第 3 章で導入した状態方程式 (1) に対して，$N = 1, A = -1, B = 1$ とし，Mamba の式 (19) において，$\theta_\Delta + s_\Delta(x)$ の t 番目の要素を a_t と書くことにすると，式 (15)，(16) をそれぞれ次のように変形することができる．

$$\begin{aligned}\bar{A}_t &= \exp(A\Delta_t)\\ &= \frac{1}{1+\exp(a_t)}\\ &= \mathrm{sigmoid}(-a_t)\\ &= 1-\mathrm{sigmoid}(a_t)\end{aligned} \tag{23}$$

$$\begin{aligned}\bar{B}_t &= A^{-1}(\exp(A\Delta_t)-I)B_t\\ &= -(\exp(A\Delta_t)-I)\\ &= 1-\bar{A}_t\\ &= \mathrm{sigmoid}(a_t)\end{aligned} \tag{24}$$

ここで，sigmoid はシグモイド関数であり，$g_t = \mathrm{sigmoid}(a_t)$ とおくと，式 (13) は次のように書くことができる．

$$h_{t+1} = (1-g_t)h_t + g_t x_t \tag{25}$$

これは GRU [1] のゲート機構と同様の形であり，時間間隔 Δ_t が現在の入力をどの程度無視するか，または使用するか制御していると解釈することができる．Δ_t の値が大きければ内部状態 h_{t-1} を無視して入力 x_t を使用するように働き，Δ_t の値が小さければ h_{t-1} を使用して入力 x_t を無視する．このように時間間隔 Δ_t は Mamba において重要な役割を果たしていることがわかる．

本研究では，時間間隔に内部状態を反映させる方法を検討する．内部状態は過去の情報を圧縮して表現しており，系列の文脈に相当する情報を含んでいると期待できる．従来の Mamba では，時間間隔は入力系列のみに依存するが，内部状態を反映させることで文脈をより直接的に反映させた推論が可能になると考えられる．具体的には，h_{t-1} を用いて Δ_t を決定し，h_t や出力 y_t の計算に活用することで文脈を考慮した予測を行うモデルを設計する．

一方，このアプローチを単純に適用すると，Parallel Scan で内部状態をどのように処理するかが課題となる．Parallel Scan は入力系列の全体に対して並列計算を行うため，各時刻の内部状態は全体の計算完了後に初めて取得可能となる．対して上記のアプローチでは，各時刻の内部状態を時間間隔に再帰的に反映させる必要があるため，Parallel Scan を適用することが困難となり，Mamba の重要な特徴である効率的な並列計算を実現できない．この課題を解決するため，提案手法である exMamba では状態空間モデルを 1 層内で 2 回適用する構造を採用する．具体的な計算手順は以下の通りである．

1. 1 回目の状態空間モデルの適用
   最初の状態空間モデル適用では，入力系列 x のみを用いて時間間隔を暫定的に決定する．この際，従来の Mamba と同様の方法で時間間隔を設定し，各時刻の内部状態を計算する．
2. 内部状態の取得
   Mamba の実装では CUDA による計算時に状態空間モデルを適用しており，特定の内部状態を直接取得することが難しい．そこで exMamba では CUDA による計算時に内部状態を別途保存して出

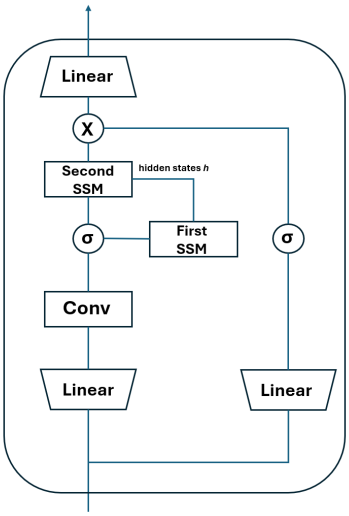

図 2　exMamba の構造

図 3　Induction Heads タスクの例

力する．これを 1 回目の適用時に実行することで，次の計算で内部状態を取得できるようにする．

3. 2 回目の状態空間モデルの適用

1 回目の状態空間モデルの適用で得られた内部状態 h を基に新たな時間間隔 Δ を決定する．入力系列が行列 $x \in \mathbb{R}^{L\times D}$ であるとき，式 (15) に内部状態 $h \in \mathbb{R}^{L\times D\times N}$ を加えて次のように変更する．

$$\Delta = \tau_\Delta(\theta_\Delta + s_\Delta(x, h)) \tag{26}$$

$s_\Delta(x,h)$ では，まず h の 2 番目と 3 番目のモードを展開することで $\hat{h} \in \mathbb{R}^{L\times DN}$ に変換する．次に，x と $\hat{h}$ からなるブロック行列

$$x' = \begin{bmatrix} x & h \end{bmatrix} \in \mathbb{R}^{L\times D(N+1)} \tag{27}$$

を計算する．さらに x' を線形層で変換することで，新たな時間間隔 Δ を更新する．最後に，更新された時間間隔 Δ を用いて再度状態空間モデルを適用し，最終的な出力を得る．

この 2 段階の手順により，並列計算を維持しつつ，文脈を考慮した適応的な時間間隔を実現する．

exMamba の構造を図 2 に示す．図の下から上にかけて入力系列を出力系列へ変換する．Linear，Conv，σ，× については Mamba と同様である．First SSM は 1 回目の状態空間モデル．Second SSM は 2 回目の状態空間モデルをそれぞれ表す．同様の構造を連結することでモデル全体を構成する．

5　実験

本章では実験を通して提案手法である exMamba を評価する．exMamba の実装にあたっては，Mamba の公式実装を参考にした．[1)]

1）　https://github.com/state-spaces/mamba

図 4　Selective Induction Heads タスクの例

図 5　Gap Induction Heads タスクの例

5.1　実験内容

実験では，Mamba と exMamba を比較することに焦点を当てた．評価タスクとして，Induction Heads[6] タスクを選定した．このタスクの概要を図 3 に図示する．まず，各正方形は系列を構成する要素であるトークンを表し，左側に系列の前方のトークン，右側に後方のトークンが順に配置されているとする．ここで，黒色のトークンをスペシャルトークンと呼ぶ．このスペシャルトークンが初めて現れたとき，その直後に配置された黄色のターゲットトークンを記憶し，次に後方の位置でスペシャルトークンが出現した際に，その直後の位置にある？が記されたトークンをターゲットトークンだと予測できるかどうかを評価する．[3] では，Mamba は Induction Heads タスクにおいて，訓練時の 4000 倍の系列長に対しても 100%の正答率を達成したと報告されている．

そこで本研究では，Induction Heads タスクの難易度を高めた 2 種類のタスクを追加した．まず，1 つ目の Selective Induction Heads タスクを説明する．このタスクの概要を図 4 に示す．まず，黒色のトークンがスペシャルトークンを表す点は Induction Heads タスクと同じであるが，黄色と紫色で表す 2 種類のターゲットトークンを仮定する．黄色のトークンをターゲットトークン 1，紫色のトークンをターゲットトークン 2 と呼ぶ．2 種類のターゲットトークンはいずれもスペシャルトークンの直後に出現するが，スペシャルトークンの直後にターゲットトークン 1 が出現するか，ターゲットトークン 2 が出現するかは，スペシャルトークンの直前のトークンによって決定される．具体的には，スペシャルトークンの直前に青色のトークンがある場合はターゲットトークン 1 となり，緑色のトークンがある場合はターゲットトークン 2 となる．これにより，黄色および紫色のターゲットトークンを記憶する．次に，後方の位置でスペシャルトークンが出現した際，その直後の位置にある？が記されたトークンを，スペシャルトークンの直前にあるトークンの色に応じたターゲットトークンとして予測する能力を評価する．以上のように，このタスクでは文脈によって同じトークンであっても異なる意味を持つことを想定し，そのような状況に対処できるか評価することができる．

次に，Selective Induction Heads タスクをさらに拡張した，Gap Induction Heads タスクを説明する．このタスクの概要を図 5 に示す．このタスクでは，黄色，紫色の 2 種類のターゲットトークンを仮定することは Selective Induction Heads タスクと同じであるが，文脈に相当する青色，緑色のトークンがスペシャルトークンの直前ではなく，P トークン離れた前方の位置に配置さ

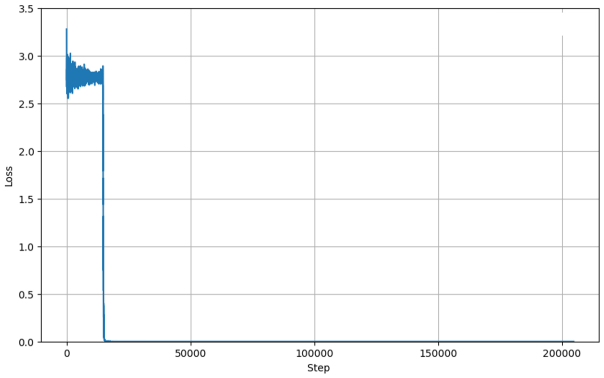

図 6　学習成功時の Mamba の Induction Heads タスクに対しての訓練誤差

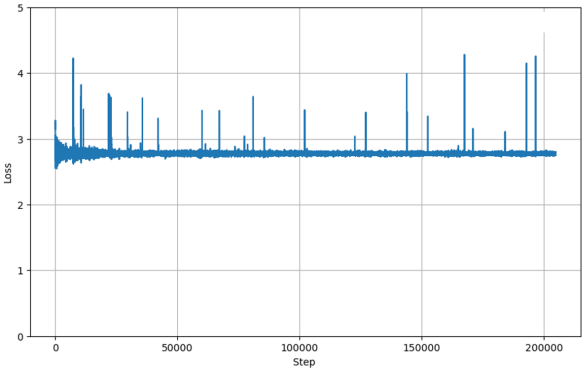

図 7　学習失敗時の Mamba の Induction Heads タスクに対しての訓練誤差

れているとする．そして青色のトークンとスペシャルトークン，緑色のトークンとスペシャルトークンの間には無関係なトークンが挿入されている．以上のように，このタスクでは長距離の関係を捉えた上で，ターゲットトークンを予測する能力を評価することができる．

全てのタスクでモデルの性能を，テストデータの各系列の最後のスペシャルトークンまでを入力系列としてモデルに与え，モデルが次のトークンを予測したときの正答率を評価した．また，事前に [3] に従って通常の Induction Heads タスクの再現実験を行ったところ，学習が完全に成功する場合と全く成功しない場合があることが確認された．そのため，本研究では，学習が成功する割合である学習可能率も評価項目に加えることにした．学習可能率は，各タスクを 100 回実行し，学習途中で訓練誤差がゼロに収束した場合を「学習成功」と定義し，その割合を集計した．データセットに関しては，16 種類の通常トークンと，スペシャルトークンを用意する．ターゲットトークン及び，Selective Induction Heads タスクと Gap Induction Heads タスクにおける文脈に相当する青色と緑色のトークンは 16 種類の中からランダムに選択した．系列データを生成するときはスペシャルトークンとターゲットトークン，文脈に相当するトークンを適切な場所に配置した後，残りのトークンを 16 種類からランダムに決定した．また，系列中のスペシャルトークンの出現位置はランダムに決定した．訓練データとしては，系列長 256 のデータセットを用い，204,800 系列を用いた．テストデータとしては，系列長

表 1　Induction Heads タスク学習可能率

モデル名	Original	Selective	Gap
Mamba	68%	83%	47%
exMamba	100%	95%	70%

64 から 2 倍ずつ系列長を増加させ，最大で 104,8576 まで用い，25,600 系列をテストデータとして用いた．モデルのハイパーパラメータ設定については，[3] と同じ値を使用した．モデルの隠れ層の数は 2，隠れ層の次元は 64，訓練時の系列長は 256，学習率は 0.001，エポック数は 25，エポックあたりのステップ数は 8192 で学習を行った．

5.2　実験結果

通常の Induction Heads タスクに加え，本研究で導入した 2 つの追加タスクに対する実験結果を説明する．まず，5.1 節で学習できるときとできないときがあると述べたが，通常の Induction Heads タスクに対して Mamba で学習に成功した場合と失敗した場合の訓練誤差の推移の例を図 6 と図 7 にそれぞれ示す．横軸は学習のステップ数を，縦軸は訓練誤差を示す．図 6 の結果のように，学習が進行する途中で急激に訓練誤差が低下する傾向が確認された．なお，急激に低下するタイミングは実行ごとにばらつき，一定ではなかった．次に図 6 の学習成功時のモデルについて，3 つの Induction Heads タスクに対しての予測精度の結果について述べる．Induction Heads タスク，Selective Induction Heads タスク，Gap Induction Heads の全てのタスクにおいて，Mamaba と exMamaba ともに学習時の系列長の 4000 倍まで正答率 100% で完璧に汎化できていた．

最後に各 Induction Heads タスクの学習可能率を表 1 に示す．ここで Original は通常の Induction Heads タスクの結果を示している．いずれのタスクについても exMamaba のほうが学習可能率が高いことが確認できる．

以上の結果から，Induction Heads タスクにおいては，exMamba が Mamba と同程度の汎化性能を示しており，このことから拡張した場合も Mamba の基本的な能力が保持されていることが確認できる．一方で，exMamba は Mamba を上回る学習可能率を示した．この結果より，文脈情報を活用した時間間隔を導入したことで，学習の安定性につながったことが示唆される．状態空間モデルの仮定に基づき，過去の入力情報を内部状態として活用することで，モデルが文脈を捉えやすくなった可能性がある．

一方で，本研究にはいくつかの課題が残されている．まず，計算効率の低下が挙げられる．Mamba は内部状態を実体化せずに Parallel Scan を行うことで，効率的な学習を可能にしているが，exMamba では内部状態を時間間隔の決定に使用するため，一度内部状態を実体化する必要がある．この影響でメモリ効率が低下し，さらに状態空間モデルを 2 回適用するため学習速度も遅くなった．

これらの問題を解決するためには，以下のような改善が必要である．1 つ目に考えられる改善策として，文脈情報として内部状態以外の情報を利用することが挙げられる．文脈を考慮できる他の扱いやすい情報を活用できれば，内部情報を参照する必要性がなくなり，Mamba の計算効率を維持することができる可能性があ

る．2 つ目に考えられる改善策として Parallel Scan の改良が挙げられる．本研究では既存の Parallel Scan をそのまま活用するために状態空間モデルを 2 回適用したが，Parallel Scan のアルゴリズムを改善し，内部状態を効率的に扱える仕組みを導入することで，計算コストを削減できる可能性がある．なお学習時と異なり，推論時には再帰的に計算するため Parallel Scan が必要ない．そのため推論に対しては，内部状態を逐次参照しつつ，計算を効率化する方法を別途検討する余地がある．

6 おわりに

本研究では，Mamba の時間間隔の決定に文脈情報を取り入れる拡張を提案した．具体的には，時間間隔に内部状態を反映させることで，Mamba が文脈に基づいたより適応的な予測を行えるように状態空間モデルを 2 回適用する設計に拡張した．この拡張により，Induction Heads タスクの学習可能率に改善が見られ，文脈的推論能力が向上する可能性が示唆された．実験では，提案手法が一定の効果をもたらす可能性を示す一方で，計算効率や実用性の面で克服すべき課題が残されていることを明らかにした．

参考文献

[1]Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1724–1734, 2014.

[2]Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4171–4186, 2019.

[3]Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. In *First Conference on Language Modeling*, 2024.

[4]R. E. Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82(1):35–45, 1960.

[5]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. *CoRR*, abs/2001.08361, 2020, 2001.08361.

[6]Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Scott Johnston, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. In-context learning and induction heads. *CoRR*, abs/2209.11895, 2022.

[7]Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. Technical report, OpenAI, 2018.

[8]Prajit Ramachandran, Barret Zoph, and Quoc V. Le. Searching for activation functions. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Workshop Track Proceedings*. OpenReview.net, 2018.

[9]Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems 30*, pages 5998–6008, 2017.