高効率な深層学習を目指すRNNの構造設計と性能評価

藤田, 倫弘

(Degree)
博士（システム情報学）

(Date of Degree)
2021-09-25

(Date of Publication)
2022-09-01

(Resource Type)
doctoral thesis

(Report Number)
甲第8175号

(URL)
https://hdl.handle.net/20.500.14094/D1008175

博士論文

高効率な深層学習を目指す RNN の構造設計と性能評価

2021 年 7 月

神戸大学大学院 システム情報学研究科

藤田倫弘

目次

図目次

表目次

概要

本論文は Recurrent Neural Network (RNN)におけるパラメータの削減ならびに構造解析についての研究に関する論文である．RNN は内部に再帰構造を有し，時系列データなどを処理することが可能であるが，時系列方向へと RNN が展開されるために計算コストやメモリ使用量が問題となることが多い．最も基本的な構造を持つ Simple RNN では，勾配消失や勾配爆発といった，学習時に勾配が不安定になってしまう問題を抱えており，実際に使用される RNN は Long Short-Term Memory (LSTM)や Gated Recurrent Unit (GRU)などのゲート付き RNN となる．これらゲート付き RNN では，複数のゲート構造，及びゲート構造に行列重みパラメータを持つことから，計算コストやメモリ使用量がさらに増加してしまい，実際の使用に際して現実的な時間での学習ができない，メモリが枯渇し学習が行えないなどの制約を受けることも少なくない．また近年では大規模データセットに対し RNN 適用が試みられることもあり，計算コスト・メモリ使用量の問題解決はより重要度を増している．したがって本研究では，この RNN における計算コスト・メモリ使用量の問題点に着目し，RNN のパラメータを削減したより効率的な RNN 構造を構築することが主な目的となる．ゆえに本論文は，RNN のパラメータを削減するべく行われた計算コスト・メモリ使用量の少ない RNN モデルの構築及び構造解析と評価に関しての以下の 3 つの研究内容についてまとめたものとなる．

研究 1: 既存のゲート構造を用いた 4 種類の RNN (Simple Gated RNN: SGR)を構築し，従来のゲート付き RNN から性能を落とすことなくパラメータを削減することを試みた．構築した SGR に対して，複合的な要素を持つ機械翻訳タスクにおける英語から日本語への翻訳により，従来のゲート付き RNN である LSTM と GRU との計算機を用いた性能比較実験を行った．その結果，構築した一部の SGR が従来の LSTM や GRU より優れた機械翻訳スコアを示した．しかしながら，多層化した場合の機械翻訳スコアは期待通りに向上せず，また SGR の機械翻訳スコアが GRU のスコアを上回るためには比較的大きなユニット数が必要であり，大幅にパラメータを削減するには至らなかった．この問題点を解消し，更なるパラメータ削減を行うには，ゲート構造の働きについて考え，ゲート構造そのものについて検討する必要があると結論付けた．

研究 2: 研究 1 における実験の結果及び考察から RNN のパラメータを更に削減するべく，ゲート構造の利点である隠れ状態のスケーリング機能に着目し，解析的な観点からゲート構造の行列重みパラメータをスカラー値に変更後，それに伴って発生するゲート構造のスケーリング機能が働かなくなる可能性を踏まえ，従来のゲート構造での活性化関数である sigmoid 関数を，従来用いられてこなかった sech 関数に置き換えたパラメータの少ない全く新しいゲート構造を構築した．そして勾配消失問題や degradation 問題を念頭に置き，新しく構築した sech 関数とスカラー値により構成されたゲート構造を shortcut connection を

基にした RNN の基本構造に挿入する形で，パラメータの非常に少ない新しい RNN を構築した．この新しい RNN は，行列重みパラメータの有無や出力活性化関数の位置などを考慮して 4 種類構築された．この 4 種の RNN について，WikiText-2 を用いた言語モデリングタスクと IMDB を用いた 2 値分類タスクという性質の異なる 2 つの自然言語処理タスクを用いて評価し，従来型のゲートを持つ LSTM や GRU との性能評価実験を行った．その結果，WikiText-2 を用いた言語モデリングタスクでは，多層化してもなお LSTM や GRU に劣る結果となってしまったが，IMDB を用いた 2 値分類タスクの場合では，パラメータが 1/6 以下と少ないながらも LSTM や GRU に匹敵する精度となった．したがって，実験の結果からゲート構造の改善によってパラメータの削減が可能であることが示された．加えてゲート構造の解析により判明した RNN モデル出力時の勾配消失問題の緩和などにより，言語モデリングタスクの性能も更に精度を向上させることが可能であると考えられ，出力活性化関数の選択や多層化時の構造について更なる検討が必要となった．

研究 3: 研究 2 において構築した sech 関数とスカラー値を用いたゲート構造を持つ RNN において，自然言語処理以外のタスクにおいて性能の更なる検討を行うため，時系列データ処理や画像処理の要素を含む応用的な医療情報処理課題としてパーキンソン病 (Parkinson's Disease: PD)検出タスクを設定する．この PD 検出タスクでは，sech 関数とスカラー値を用いたゲートを持つ RNN の内，研究 2 の 2 値分類タスクにおいて最も性能の良かった RNN を RNN-SH と呼称し，この RNN-SH を用いて更に検証を進めた．PD の検出では，PD 患者の運動障害性構音障害に着目し，PD 患者の音声を特定区間に分割したものを 2 次元リカレンスプロット画像に変換し，CNN と RNN を用いた PD 検出モデルにより検出する．実験では PD 検出精度によって，従来のゲート付き RNN との性能比較や，出力活性化関数を tanh から relu 関数に変更した場合の性能の検討を行った．その結果，tanh を出力活性化関数に用いた場合の RNN-SH と従来の LSTM，GRU を使用した各 PD 検出モデルの精度に統計的優位差は認められなかった．しかしながら性能差が小さいにも関わらず，RNN-SH を用いた PD 検出モデルのパラメータ量は GRU を用いた場合の 1/3 以下であり，RNN-SH が自然言語処理のみならず，時系列データ処理や画像処理などのタスクにも応用可能であることが示唆される．一方で，研究 2 において問題となった RNN-SH の多層化における精度向上は，多層化に重要な役割を果たす活性化関数である relu 関数を用いた場合でも検出精度を向上させることはできず，パラメータの初期化や正規化法を検討し，より詳細な調査を行う必要がある結果となった．PD 検出精度に関しても，実験を通して 70%前後と実用的な精度には到達しなかった．原因として，データセットの大きさや PD 検出に適した単母音の選択，RP 画像を作成する時間間隔， RP 画像のサイズ，PD 検出モデルにおける CNN の構造などが最適ではなかった可能性がある．したがって，これらを最適化することで更に精度を向上させることができると考えられ，今後改善すべき点について詳細な調査・検討を行い，PD 検出精度の改善を行う予定である．

本論文は 6 章で構成され，概要はそれぞれ以下の通りである．

第 1 章では，近年の複雑化するニューラルネットワークの設計思想と計算資源の制約について本研究の背景を述べ，RNN のパラメータ削減における研究の重要性を説明する．また，本研究の目的及び特徴についても述べ，他の関連研究との比較と問題点，並びに本研究の特徴と意義を確認する．

第 2 章では，本研究で扱う従来型 RNN の基本的な構造と特徴などについて述べ，主要な RNN モデルについて具体的に説明する．またそれぞれ RNN モデルにおける問題点やパラメータ量について説明し，本研究の意義について述べる．

第 3 章では，研究 1 の内容について，既存のゲート構造取り入れた新しい RNN モデルの構造や機械翻訳タスクにおける性能評価実験の概要，並びに実験の結果について述べる．

第 4 章では，研究 2 の内容について，新しい sech 関数とスカラー値を用いたパラメータの少ないゲート構造を持つ，shortcut connection を基にした RNN モデルの構造や，性質の異なる 2 つの自然言語処理タスクである WikiText-2 を用いた言語モデリングタスク並びに IMDB を用いた 2 値分類タスクにおける性能評価実験の概要，並びにその実験の結果について論じる．

第 5 章では，研究 3 の内容について，研究 2 で構築した sech 関数とスカラー値を用いたゲートを持つ RNN の内, IMDB を用いた 2 値分類タスクにおいて最も性能の良かった RNN の 1 つ用いて行われた，リカレンスプロットを用いた PD 検出タスクにおける性能評価実験の概要, 並びに実験の結果とリカレンスプロットを用いた PD検出タスクにおける改善すべき点と展望について述べる．

第 6 章では，各章で得られた結果を要約し，本研究における結論及び，今後の課題と展望について述べる．

Abstract

This paper describes the study of parameter reduction and structural analysis in Recurrent Neural Networks (RNNs). RNN has a recursive structure inside and can process time series data, however, the calculation cost and memory usage often become problems due to unrolling input time steps of RNNs in the learning. For Simple RNN which is a basic structure among RNNs, it is difficult to learn long-term time series data due to the vanishing gradient problem, exploding gradients problem, etc. Because of this, the RNN actually used are gated RNNs such as Long Short-Term Memory (LSTM) or Gated Recurrent Unit (GRU). These gated RNNs have multiple gate structures and matrix weight parameters for each gate structure, which further increases the calculation cost and memory usage, and often results in the inability to learn in a realistic amount of time or to train due to out of memory. In recent years, RNN has been tried to be applied to large-scale datasets, and solving the problems of calculation cost and memory usage is becoming more important. Therefore, the purpose of this study is to focus on the problems of calculation cost and memory usage in RNN, and to construct a more efficient RNN structures with reduced parameters. Accordingly, this paper consists of the following three researches on the construction, structural analysis, and evaluation of RNN models with low calculation cost and memory usage, which were carried out to reduce RNN parameters.

Study 1: Four types of RNN (Simple Gated RNN: SGR) using conventional gate structures was constructed, and it was attempted to reduce parameters without degrading performance from conventional gated RNNs such as LSTM and GRU. We compared the performance of SGR with that of LSTM and GRU by translating from English to Japanese in a machine translation task. As a result, some of the constructed SGRs showed better machine translation scores than the conventional LSTM and GRU. However, the machine translation scores of SGRs in the case of multi-layering does not improve as we expected, and the relatively large number of units is required for the scores of SGR to exceed the score of GRU. Therefore, it was not able to reduce the parameters significantly. In order to solve this problem and further reduce the parameters, it is necessary to consider the function of the gate structure and examine the gate structure itself.

Study 2: From the results and considerations of the experiments in study 1, in order to further reduce the RNN parameters, we focused on the scaling feature which is the advantages of the gate structure, changed the matrix weight parameter and conventional sigmoid function of the gate to a scalar value and sech function, which have not been used in the gate in the past, and constructed a novel gate structure with fewer parameters. Then, keeping in mind the

vanishing gradient problem and the degradation problem, a new RNN with fewer parameters was constructed by inserting this newly gate structure composed of sech function and a scalar value into a structure of the RNN based on the shortcut connection. Here, four types of new RNNs were constructed in consideration of the presence or absence of the matrix weight parameters for hidden state and the position of the output activation function. In performance evaluation experiments, these four types of RNNs were evaluated using two natural language processing tasks with different properties, which are a language modeling task using WikiText-2 and a binary classification task using IMDB, and compared with LSTM and GRU. As a result, in the case of the language modeling task using WikiText-2, our RNN model was inferior to LSTM and GRU even if it was multi-layered, however, in the case of the binary classification task using IMDB, the accuracy was about the same as LSTM and GRU with the parameters were 1/6 or less. Therefore, the experimental results showed that the parameters can be reduced by improving the gate structure. In addition, it is assumed that it will be possible to improve the performance of the language modeling task by alleviating the vanishing gradient problem at the time of RNN model output, which was found by analyzing the sech gate structure. Further studies are required on the selection of the output activation function and the structure at the time of multi-layering.

Study 3: In order to further investigate the performance of RNNs constructed in study 2 with a gate structure using sech function and a scalar value in tasks other than natural language processing, we set the Parkinson's Disease (PD) detection task as an applied medical information processing task that includes elements of time series data processing and image processing. Here, in PD detection task, among the RNNs that have gate using the sech function and a scalar value, the RNN with the best performance in the binary classification task in study 2 was called RNN-SH, this RNN-SH was used, and further verification was carried out. In the detection of PD, we focused on the motor disorder dysarthria of PD patients, divided the voice of PD patients into specific sections, and converted they into two-dimensional recurrence plot images. After that, we classified the RP images by the PD detection model using CNN and RNN. In the experiment, regarding the PD detection, we compared the performance of RNN-SH with that of the conventional gated RNN and examined the performance when the output activation function was changed from tanh function to relu function. As a result, statistically significant difference was not found in the accuracy of each PD detection model between RNN-SH with tanh and conventional LSTM and GRU. However, despite the small differences of the accuracy, the parameter amount of the PD detection model using RNN-SH is less than 1/3 of that using the GRU, and it was suggested that RNN-SH can be applied not only to natural language processing but also to tasks such as time series data processing and image processing. On the other hand, the PD

detection accuracy in multi-layering of RNN-SH, which was the problem in study 2, could not be improved even when using the relu function, which is an activation function that plays an important role in multi-layering. From this result of relu, it is necessary to further examine the parameter initialization and normalization methods, and to conduct a more detailed analysis. The PD detection accuracy was around 70% and did not reach a practical accuracy throughout the experiment. This causes may be due to the size of the dataset, the choice of suitable voice for PD detection, the time interval for generating RP images, the size of the RP images, or the structure of the CNN in the PD detection model may not have been optimal. Therefore, we consider that the accuracy can be further improved by optimizing them. We will plan to improve the accuracy of PD detection through detailed analysis and study of above points that need to be improved.

This paper is composed of 6 chapters, and the outline is as follows.

In Chapter 1, the background of this study was described regarding the complicated design concept of neural networks and the constraints of computational resources in recent years, and the importance of study of reducing RNN parameters is explained.

In Chapter 2, I describe the basic structures and features of conventional RNNs used in this study, and explain the main RNN models in detail. I also explain the problems and the number of parameters in each RNN model and describe the significance of this study.

Chapter 3 describes the contents of study 1, which is the structure of the new RNN models incorporating the existing gate structure, the outline of the performance evaluation experiment in the machine translation task, and the result of the experiment.

Chapter 4 describes, as the contents of study 2, the structure of the novel RNN models based on the shortcut connection, which has a gate structure with few parameters using a new sech function and a scalar value. I also describe the outline of the performance evaluation experiments in the language modeling task using WikiText-2 and the binary classification task using IMDB, which are two natural language processing tasks with different properties, and the results of the experiments.

Chapter 5 describes, regarding the contents of study 3, the outline of the PD detection task using the recurrence plot and the performance evaluation experiment in the PD detection task. Here, among the RNNs with gates using the sech function and a scalar value constructed in study 2, one of the RNN with the best performance in the binary classification task using IMDB was used. Then, the results of the experiment and the points and prospects to be improved in the PD detection task using the recurrence plot are explained.

In conclusion, Chapter 6 summarizes the results obtained in each chapter, the conclusions of this study, and future issues and prospects.

第1章 序論

1.1 研究背景

システム制御や画像処理，医療情報処理などの様々な情報処理に対して機械学習を用いて処理することが多くなり，機械学習の中でも特に特徴量を自動的に抽出することができる深層ニューラルネットワークモデルが使用されるケースが急増している．この背景には，計算機の性能向上によって多層化された深層ニューラルネットワークモデルが使用可能となったことや，ニューラルネットワークモデルの研究及び開発に使用されるオープンソースソフトウェアライブラリの発展，深層学習に特化した Field-Programmable Gate Array (FPGA)の登場などにより多種多様なタスクに対して簡易に応用が可能となったことが挙げられる．多層化された深層ニューラルネットワークモデルは，自動運転や医療研究，産業のオートメーション化，文章生成，機械翻訳等の様々なタスクに応用され人々の生活に切り離せない技術として定着しつつある．しかし，未だ計算機の性能は十分ではなく，計算コストやメモリ使用量といった計算資源の面で性能が発揮できないケースも少なくない．それゆえ Recurrent Neural Network (RNN)[1-4]や Convolutional Neural Network (CNN)[5-7]におけるニューラルネットワークモデルの多層化や構造の複雑化，更には扱うデータサイズの大規模化に伴い，計算コスト・メモリ使用量の増加は非常に大きな問題点となっている．加えてニューラルネットワークモデルの構造が複雑化するにつれて性能面での解析が難しく，更には設計者の経験に基づきモデルが設計されるため，精度を高めることに重きが置かれモデルが非常に複雑化するケースが増加している．例えば，上記の内容は RNN における多層化や RNN の種類及びゲート構造の挿入位置，また CNN における shortcut connection[7]による大幅な深層化やカーネルサイズ等のパラメータ調整，CNN と結合する多層パーセプトロンの有無などが該当する．このことは計算コストやメモリ使用量の観点からだけではなく，数理的な観点からも今後のニューラルネットワークモデルの研究において問題視されるべきであり，構造の複雑化が性能においてどのような役割を持つのかを調査することは非常に重要となる．

特に時系列データを扱う RNN では，逆伝搬計算における順伝搬の再計算を避けるため，計算機による計算時には時系列方向へとニューラルネットワークが展開されメモリ使用量が非常に大きくなる．系列長が長ければ，非常に層の深いニューラルネットワークモデルとなることからメモリが枯渇し，正しく学習が行えないケースも想定される．また RNN は，RNN の特徴である再帰構造により，前時刻での計算を処理した後でなければ次時刻での計算を開始できないために並列性も低くなってしまう．可変長入力を処理する場合では，バッチサイズが揃わないためにより並列性が低くなってしまうことから，学習にかかる時間は更に長くなり，現実的な時間での学習が行えなくなるといった問題点が生じる可能性があ

る．そのため，RNN ではモデルの複雑化によるパラメータの増加の影響が非常に大きいと言える．RNN の使用においては，最も基本的な構造を持つ Simple RNN[1]が使用されることは少なく，行列重みパラメータを有するゲート構造を持ち，より性能の良いゲート付き RNN を用いることが一般的となっている．最も代表的なゲート付き RNN である Long Short-Term Memory (LSTM)[2,3]では，入力ゲート，忘却ゲート，出力ゲートの 3 つのゲート構造により必要とされる情報を制御することで，Simple RNN が抱える勾配消失や勾配爆発といった問題点に対してゲート構造が有効に働き，長いタイムステップを有する時系列データ処理において特に優れた性能を発揮する．しかしながら LSTM ではゲート構造に行列重みパラメータを持つことから，RNN の抱える計算コスト・メモリ使用量や計算時における並列性の問題がより大きな課題となり，実際の応用課題や大規模データセットへの適用にはかなりの制限がかかる．具体的には LSTM では 3 つのゲート構造を有することから Simple RNN の 4 倍のパラメータが必要となり，計算コスト・メモリ使用量はさらに増大してしまう．このパラメータ量の問題点に対して，より効率的な構造として提案された RNN が Gated Recurrent Units (GRU)[4]である．GRU ではリセットゲートと更新ゲートという 2 つのゲート構造を用いて必要とされる情報を制御するため，LSTM の 3/4 のパラメータ量であり，LSTM よりも計算コスト・メモリ使用量が改善された構造となっている．しかし Simple RNN と比較した場合のパラメータ量は 3 倍であることから，計算コスト・メモリ使用量は依然として高く，また GRU は構造的に並列性が低いため，実装方法や実行環境によっては LSTM よりも実行速度が遅くなる場合があるなどの問題点がある．

したがって，上記のことから RNN の計算コスト及びメモリ使用量の削減は非常に重要であり，大規模なデータセットへの適用も行われる中での RNN モデルのパラメータ削減を目指す研究意義は大きい．この背景を踏まえ次節において本研究の目的について説明する．

1.2 研究の目的と概要

RNN モデルにおいてゲート構造のパラメータが占める割合は非常に大きいが，不要なパラメータの増加は，モデルの過学習を引き起こしたり解析を難しくしたりするため，本来可能な限り回避されるべきである．パラメータの削減は，計算資源の限られた環境におけるニューラルネットワークモデルの使用において，工学的な面から非常に重要であると考えられる．したがって本研究では RNN の構造そのものに着目し，計算コスト・メモリ使用量の観点からパラメータの削減及び構造解析を行うことを目的とする．そこで従来の RNN を基に，構造をより簡単化した新たな RNN モデルを構築しパラメータを削減する．その上で構築した RNN の性能について計算機を用いた実験により従来型の RNN と比較･検証を行い，妥当性を評価し確認する．以下に本論文のおける研究内容と実験の概要及び得られた結果について簡潔に述べる．

研究 1 として，従来型のゲート構造を用いた 4 種類の RNN を構築し，性能を落とすこと

なくパラメータを削減することを試みた．構築した 4 種類の RNN (Simple Gated RNN: SGR)に対して，複合的な要素を持つ機械翻訳タスクにおける英語から日本語への翻訳により，LSTM と GRU との性能比較実験を行った．その結果として，構築した一部の SGR が従来の LSTM や GRU より優れた機械翻訳スコアを示した．その一方で，SGR を多層化した場合の機械翻訳スコアの向上は期待通りではなく，SGR の機械翻訳スコアが GRU のスコアを上回るためには比較的大きなユニット数が必要となり，大幅にパラメータを削減するには至らなかった．

研究 2 として，RNN におけるパラメータを更に削減するため，ゲート構造の利点である隠れ状態のスケーリング機能に着目し，解析的な観点からゲート構造の行列重みパラメータをスカラー値に変更する．またそれに伴ってゲート構造のスケーリング機能が働かなくなる可能性を踏まえ，従来のゲート構造に用いられる活性化関数である sigmoid 関数を，従来用いられてこなかった sech 関数に置き換えたパラメータの少ない全く新しいゲート構造の構築を行った．そして勾配消失問題や degradation 問題を念頭に置き，新しく構築した sech 関数とスカラー値により構成されたゲート構造を shortcut connection を基にした RNN の基本構造に挿入する形で，パラメータの非常に少ない新しい RNN を 4 種類構築した．この 4 種類の RNN について，WikiText-2 を用いた言語モデリングタスクと IMDB を用いた 2 値分類タスクという性質の異なる 2 つの自然言語処理タスクを用いて評価し，従来型のゲートを持つ LSTM や GRU との性能評価実験を行った．その結果，WikiText-2 を用いた言語モデリングタスクでは，多層化してもなお LSTM や GRU に劣る結果となってしまったが，IMDB を用いた 2 値分類タスクの場合では，パラメータが 1/6 以下と少ないながらも LSTM や GRU に匹敵する精度となった．

研究 3 として，研究 2 で構築した RNN について自然言語処理以外のタスクにおける性能に関して更なる検討を行うため，時系列データ処理や画像処理の要素を含む応用的な医療情報処理課題としてパーキンソン病(Parkinson's Disease: PD)検出タスクを設定した．この PD 検出タスクでは，研究 2 の 2 値分類タスクにおいて最も性能の良かった RNN を RNN-SH と呼称し，この RNN-SH を用いて更に検証を進めた．検出方法としては，PD 患者の運動障害性構音障害に着目し，PD 患者の音声を特定区間に分割したものを 2 次元リカレンスプロット画像に変換後，CNN と RNN を用いた PD 検出モデルにより検出する．実験では PD 検出精度を用いて，LSTM や GRU との性能比較や，出力活性化関数を tanh から relu 関数に変更した場合の性能の検討を行った．その結果，tanh を出力活性化関数に用いた場合の RNN-SH と従来の LSTM，GRU を使用した各 PD 検出モデルの精度に性能差はあまりなく，統計的優位差は認められなかった．しかしながら，RNN-SH を用いた PD 検出モデルのパラメータ量は GRU を用いた場合の 1/3 以下であり，RNN-SH が自然言語処理のみならず，時系列データ処理や画像処理などのタスクにも応用可能であることが示唆される結果が得られた．一方で，研究 2 において問題となった RNN-SH の多層化における精度

向上は，多層化に重要な役割を果たす活性化関数である relu 関数を用いた場合でも検出精度を向上させることはできず，パラメータの初期化や正規化法を検討し，より詳細な調査を行う必要がある結果となった．

一般的に，代表的なゲート付き RNN である LSTM では，長期の依存関係，並びに大規模から中規模程度のデータセットにおける時系列データ処理において性能が良いとされ，GRU では，比較的短い中・長期の依存関係，並びに比較的大規模なものから小規模程度のデータセットまでにおける時系列データ処理において性能が良いとされる．これらを踏まえ，本研究は限られた計算資源のもとで中規模程度までのデータセットを扱う場合のタスクにおけるパラメータ削減に対して取り組むものである．

1.3 関連研究と本研究の意義

本研究の関連研究におけるゲートを持つ RNN の構造には，MUT[8]や Minimal Gated Units (MGU)[9]，Simple Recurrent Units (SRU)[10]などの RNN がある．これらの RNN では，ゲート構造を減らす，あるいはゲート構造の行列重みパラメータをベクトルに変更するなどして LSTM や GRU よりも計算コストを削減し，より効率的な構造を模索している．具体的には MUT や MGU は主に GRU ベースの構造であり，MUT ではゲート構造の重みパラメータを削減し，MGU では GRU における 2 つのゲート構造を共通にしてパラメータを削減している．また SRU は LSTM 及び GRU の両方をベースとしたような構造となっており，ゲート構造の行列重みパラメータをベクトルに変更し，ゲート数が 2 つと LSTM よりゲート数が少なくなっている．しかしながら，MUT や SRU では複数のゲート構造ならびにゲートにおける複数のアダマール積が必要になることから，比較的計算コスト・メモリ使用率が高い．また GRU ではリセットゲートと更新ゲートという 2 つのゲートを持っており，構造的にリセットゲートの計算を先に行ってからでなければ隠れ状態に対する重み行列を計算できない箇所があるため，一括で重み行列を計算できず，LSTM や SRU と比べて並列性が低くなってしまうという問題点がある．MUT や MGU では GRU をベースとしているため，同じように並列性が低いことで実行時間が長くなりやすい問題点が残されている．本研究 1 におけるゲート構造を用いた RNN の簡単化では，隠れ状態に対して重みパラメータを持たない Simple RNN に対し，LSTM もしくは GRU のゲート構造を取り込んだ RNN を構築することでパラメータを削減する．特にゲートからの出力を用いた線形補間によりゲート計算を行う GRU の更新ゲートのみを挿入した RNN モデルでは，ゲートの数が１つしかないことから従来のパラメータ削減手法よりもモデルパラメータが少なく済み，計算コスト・実行時間の点において有利かつ並列性も高いため実装もより単純で高速なものとなる．MUT や SRU などの他のゲート付き RNN と本研究 1 の比較を表したグラフを図 1.1 に示す．

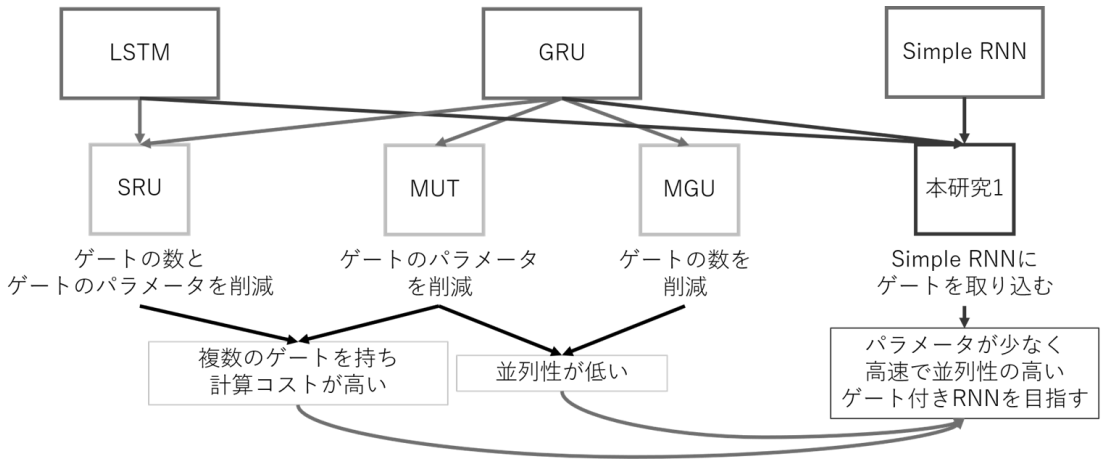

図 1.1 他のゲート付き RNN と本研究 1 の比較

一方で近年，ゲート付き RNN とは別のアプローチ手法として，画像処理において提案された shortcut connection[7]より着想を得た RNN 構造も存在している．shortcut connection は，深層学習において層が増えるにつれ性能が頭打ちになってしまうという degradation 問題を解決すべく導入された構造であり，shortcut connection は特定の層ごとに，中間層による変換を変換前の値に足しこむ形で構成される単純な構造を持つ．この shortcut connection から着想を得た RNN の関連研究には Residual RNN (Res-RNN)[11]や Independently RNN (IndRNN)[12]などがある．これらの RNN では，shortcut connection という単純な構造の上に構成され，計算コストが比較的少ない RNN を構築できる利点がある．しかしながら，単純な構造だけで性能を確保することは難しいため，Res-RNN[11]では shortcut connection 単独の構造ではなく，行列重みパラメータを持つゲート構造を使用することで性能を確保しており，計算コストなどが問題となる．また IndRNN[12]では，再帰重みをベクトルに変更し，batch normalization[13]という正規化手法との組み合わせにより性能を確保している．この IndRNN では，batch normalization がバッチサイズに依存して正規化を行うことから，可変長なデータに対する性能が定かではなく，加えて batch normalization の RNN 内部での挿入位置はタスクごとに変える必要があるため，様々なタスクにおいて画一的に使用することが難しい．本研究 2 における sech 関数を用いたゲート構造によるパラメータ削減では，ゲート構造における隠れ状態のスケーリング機能に着目し，ゲート構造の利点を生かしながらも，従来用いられてこなかった sech 関数とスカラー値を用いたパラメータの非常に少ない新しいゲート構造を構成する．その上で勾配消失問題と degradation 問題を念頭に置き，shortcut connection を基にした RNN の基本構造に sech 関数とスカラー値を用いた新しいゲート構造を挿入することで RNN モデルを作成するという独自の構造的アプローチによりパラメータの少ない RNN を構築する．この RNN は先に述べた関連研究におけるどの RNN モデルよりもモデルパラメータが少なく，ゲート構造が単純かつスカラー値を用いるためゲート構造の解析も容易である．また本研究は正規化手法などに依存せず，柔軟に使

用及び適用可能な構造を目指すものであり，ゲート構造に用いる活性化関数を変更してパラメータ削減を図る研究は本研究特有のものである．ここで shortcut connection を有する Res-RNN 及び IndRNN と本研究 2 の比較を表したグラフを図 1.2 に示す．

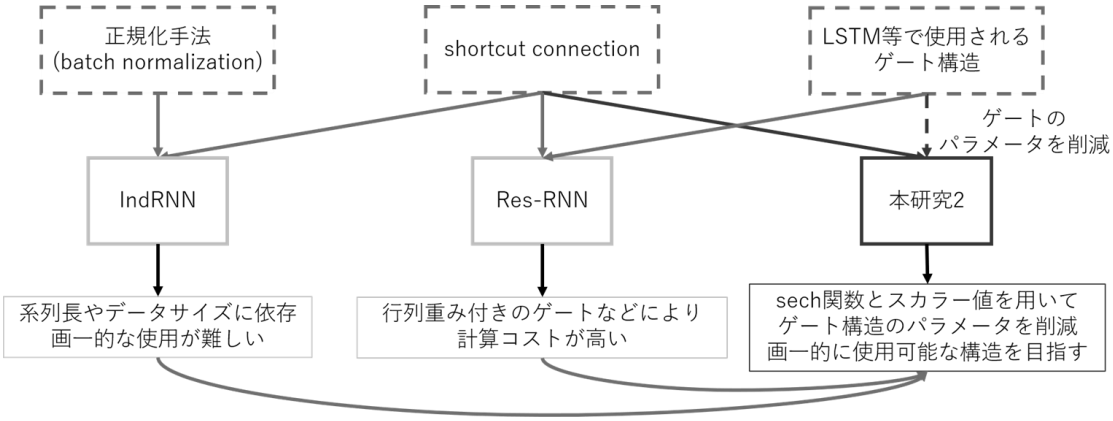

図 1.2 Res-RNN 及び IndRNN と本研究 2 の比較

本研究の意義，特徴を以下にまとめる．

1. 従来の RNN 研究では性能向上のため，構造を複雑にするようなものが多数を占め，計算コストの削減を目指す論文は少ない．
2. 本研究におけるゲート構造を持つ RNN の簡単化では，線形補間によりゲート計算を行う GRU の更新ゲートを隠れ状態に対する行列重みパラメータを持たない Simple RNN に挿入する形で，ゲート数が 1 つしかなく従来研究よりもパラメータの少ないモデル構築を行い，更に活性化関数を RNN の再帰構造の外部で適用することで勾配消失を緩和し，性能を下げることなくパラメータを削減している．
3. RNN における LSTM や GRU などの従来型のゲート構造の利点に着目し，これまでゲート構造に従来用いられてこなかった新たな活性化関数として sech 関数と行列重みパラメータに代わりスカラー値を用いたパラメータの少ないゲート構造を構築する．その上で，勾配消失問題と degradation 問題を念頭に置き，構築したパラメータの少ない sech ゲート構造を shortcut connection を基にした単純な RNN に挿入し，RNN モデルのパラメータを削減するという構造的なアプローチによりパラメータを削減し，様々なタスクにおいて画一的に使用可能な構造を目指すものである．またゲート構造に用いる活性化関数を変更することでパラメータを削減するという先行研究には見られない独自のアプローチを行っている．

以上が，他の関連研究との比較並びに本研究の意義，特徴である．

1.4 論文構成

本論文は次のように構成される．

第 2 章では，まず，本研究で扱う従来型 RNN の構造と特徴などについて説明し，代表的な RNN モデルについて具体的に説明する．またそれぞれの問題点やパラメータ量について述べ，本研究の意義を述べる．

第 3 章では，既存のゲート構造を用いた RNN の簡単化が行われ，性能を保ったままパラメータを削減した RNN を構築するべく，4 種の新しい RNN を構築する．構築した RNN について機械翻訳を用いた計算機実験により，従来のゲート構造を持つ RNN との比較を行いその性能を評価する．

第 4 章では，第 3 章の既存のゲート構造を用いた RNN から更に発展させ，ゲート構造を簡素化した新たな構造の RNN を構築し，ゲート構造が性能に及ぼす影響を考慮し，更なるパラメータ削減の余地について検討を行う．具体的には，スカラー値とこれまで活性化関数として用いられてこなかった sech 関数を用いた，パラメータの少ない新しいゲート構造を持つ RNN を 4 種類構築する．この 4 種の RNN に対して，自然言語処理における言語モデリングタスクと 2 値分類タスクを用いた性能評価を行い，加えて，ゲート構造の解析及び調査を行う．

第 5 章では，第 4 章で構築したスカラー値と sech 関数を用いたゲート構造を持つ RNN において更なる検討を行うため，4 章の実験で性能が良かった 1 つのモデルに対して自然言語処理以外のタスクにおける性能を測る．そのため，時系列データ処理や画像処理を含む応用的な情報処理課題としてパーキンソン病(Parkinson's Disease: PD)の検出タスクを設定し，計算機を用いた実験を行う．より具体的には，PD 患者のもつ運動障害性構音障害に着目し，音声を特定区間に分割したものを 2 次元リカレンスプロット画像に変換，そして RNN を用いた PD 検出モデルにより検出し，その性能を比較及び評価する．

最後に，第 6 章では本研究の成果と結論についてまとめ，今後の課題と展望について述べる．

第2章 Recurrent Neural Network (RNN) の構造と特徴

本章では，主要な RNN の構造とその特徴について述べる．また RNN における研究の重要性について説明する．

2.1 Recurrent Neural Network (RNN) の概要

RNN は内部に再帰構造を持ち，主にフィードバック制御を必要とするタスクにおいて用いられる．再帰構造では，RNN の出力は隠れ状態として再度 RNN に入力され目的の出力を生成することができる．また再帰構造を持つことによって隠れ状態を考慮して出力生成を行うことができるため，予測などのタスクにも用いることができ，RNN の大きな利点の一つである．RNN の使用用途は非常に多岐にわたり，自動運転や医療研究，産業システムの制御，文章生成，機械翻訳や会話文生成などの様々なタスクに使用されており応用性の高いニューラルネットワークである．

RNN の概念図を図 2.1 に示す．図 2.1 において x_t は入力，h_t は隠れ状態を表している．

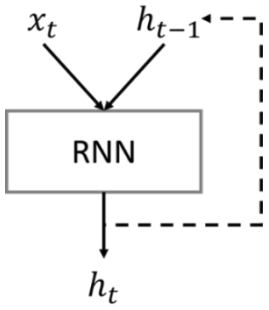

図 2.1: RNN の概念図

RNN の学習には Backpropagation Through Time (BPTT)が用いられ，時系列方向に展開することで通常の多層パーセプトロンと同様に誤差逆伝播法を用いて勾配計算を行うことができる．BPTT の概念図を図 2.2 に示す．

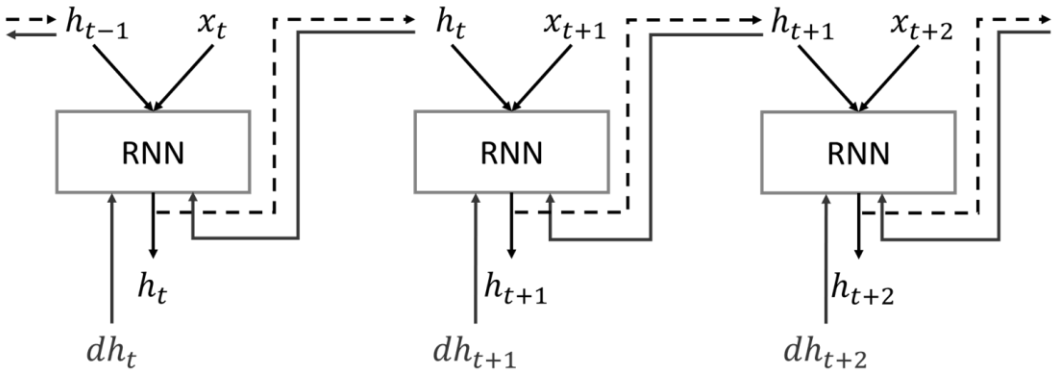

図 2.2: BPTT の概念図

図 2.2 に示すように下層(出力に近い側)からの勾配に，次時刻からの勾配が加算される形で逆伝搬が行われる．

2.2 Simple RNN

RNN の中で最も基本的な構造を持つのが Simple RNN[1]である．Simple RNN は入力に対する重みと隠れ状態に対する重みを持ち，以下の数式で定義される．

$$h_t = \mathrm{act}(Wx_t + Uh_{t-1} + b) \tag{2.1}$$

ここで m を Simple RNN への入力の次元，n を出力の次元とすると，$x_t \in \mathbb{R}^m$ は時刻 t での入力，$h_t \in \mathbb{R}^n$ は時刻 t での隠れ状態及び出力，$W \in \mathbb{R}^{n\times m}, U \in \mathbb{R}^{n\times n}$ は重み行列，$b \in \mathbb{R}^n$ はバイアス項を表している．また，act は活性化関数を表している．

Simple RNN の概念図を図 2.3 に示す．なお，バイアス項は省略している．

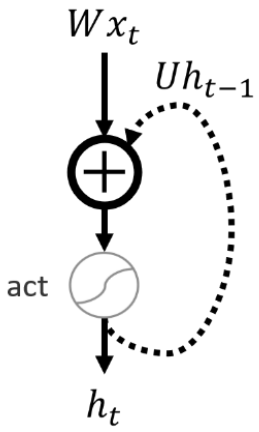

図 2.3: Simple RNN の概念図

Simple RNN に用いられる代表的な活性化関数には sigmoid 関数，tanh 関数，relu 関数などがあり，それぞれ次に示す数式で表される．

sigmoid 関数:

$$\mathrm{sigmoid}(x) = \frac{1}{1+e^{-x}} \tag{2.2}$$

tanh 関数:

$$\tanh(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}} \tag{2.3}$$

relu 関数:

$$\mathrm{relu}(x) = \max(0, x) \tag{2.4}$$

また，それぞれの導関数をそれぞれ次に示す．

sigmoid 関数の導関数:

$$\mathrm{sigmoid}'(x) = \big(1 - \mathrm{sigmoid}(x)\big) \cdot \mathrm{sigmoid}(x) \tag{2.5}$$

tanh 関数の導関数:

$$\tanh'(x) = 1 - \tanh^2(x) \tag{2.6}$$

relu 関数の導関数:

$$\mathrm{relu}'(x) = \begin{cases} 1 & \text{if } x > 0 \\ 0 & \text{otherwise} \end{cases} \tag{2.7}$$

ここで relu 関数は $x = 0$ の時，厳密には微分可能ではないが，便宜上，微分値を 0 とすることで対処する．

各活性化関数のグラフを図 2.4 に示す．

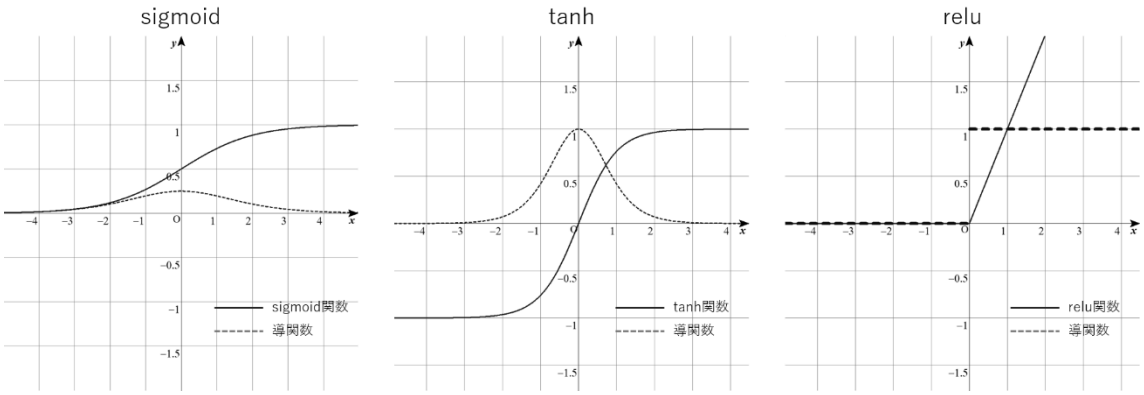

図 2.4: 活性化関数と導関数のグラフ

Simple RNN には長期の依存関係を学習することが難しいという問題点がある．この原因の 1 つには，層を経るにつれて得られる勾配が消失してしまい学習が進行できなくなる，あるいは学習が不安定になってしまうという勾配消失問題が挙げられる．この勾配消失問題では，活性化関数を微分した際に得られる小さな値や特異値が 1 より小さい重み行列 U を誤差逆伝播法において複数回乗算することによって発生する．これとは逆に，勾配が層を経るごとに大きくなってしまい発散してしまう勾配爆発という問題も存在している．他にも，degradation 問題と呼ばれる多層化により精度が悪化してしまうという問題点もある．この degradation 問題の原因としては，学習の収束に必要な時間の増大や非線形性が強くなることにより恒等写像を学習することが難しいといった理由が考えられているが未だ明確な答えがないのが現状である．

したがって上記のことから Simple RNN を学習させることは難しく，実応用では次節以降で述べるゲート付き RNN が用いられるのが一般的となっている．

2.3 Long Short-Term Memory (LSTM)

Long short-term memory (LSTM)[2, 3]は，Simple RNN の勾配消失問題に対処するためのゲート構造を備え，必要とされる情報をゲート構造によって制御する RNN の 1 つである．LSTM は Constant Error Carousel (CEC)という考え方に基づき設計されている．CEC とは重みを掛けることなく誤差情報を逆伝搬させることで勾配消失問題に対処しようとするものである．CEC はメモリセル(memory cell)と呼ばれ，本論文中ではセルの状態を内部セルと呼ぶ．

一般的に使用される典型的な LSTM は以下の数式で定義される．

$$z_t = \tanh(W_z x_t + U_z h_{t-1} + b_z) \tag{2.8}$$

$$i_t = \mathrm{sigmoid}(W_{in} x_t + U_{in} h_{t-1} + b_{in}) \tag{2.9}$$

$$o_t = \mathrm{sigmoid}(W_{out} x_t + U_{out} h_{t-1} + b_{out}) \tag{2.10}$$

$$f_t = \mathrm{sigmoid}\left(W_{for} x_t + U_{for} h_{t-1} + b_{for}\right) \tag{2.11}$$

$$c_t = i_t \circ z_t + f_t \circ c_{t-1} \tag{2.12}$$

$$h_t = \tanh(c_t) \circ o_t \tag{2.13}$$

ここで m を LSTM への入力の次元，n を出力の次元とすると，$x_t \in \mathbb{R}^m$ は時刻 t での入力，$h_t \in \mathbb{R}^n$は時刻 t での隠れ状態及び出力を表す．また下付き文字に関わらず，$W \in \mathbb{R}^{n \times m}, U \in \mathbb{R}^{n \times n}$ は重み行列，$b \in \mathbb{R}^n$ はバイアス項を表している．z_t は時刻 t における内部セルへの入力，i_t, f_t, o_t はそれぞれ時刻 t における入力ゲート，忘却ゲート，出力ゲートからの出力，c_t は時刻 t における内部セルの値を表す．なお，$\circ$はゲートの計算に用いられるアダマール積(要素ごとの積)である．

LSTM は CEC に蓄積される情報を制御するため，入力ゲートと出力ゲートが導入されている．ゲートの出力は，sigmoid 関数を通すことにより，$(0, 1)$ の出力を得る．入力ゲートでは，セルに入力される z_t の値を式(2.9)からの出力である i_t とのアダマール積により制御する．またセルからの出力に関しても，内部セル c_t に活性化関数を適用した出力と式(2.10)からの出力である o_t とのアダマール積により制御し，隠れ状態 h_t を生成する．

初期に考案された LSTM[2]では忘却ゲートが存在せず，内部セルを初期化することができなかったため急激なデータの変化に対応できない問題点があったが，後に Gers らによって式(2.11)に示す忘却ゲート f_t が導入され，内部セルの初期化が可能となった[3]．

LSTM の概念図を図 2.5 に示す．なお，バイアス項は省略している．

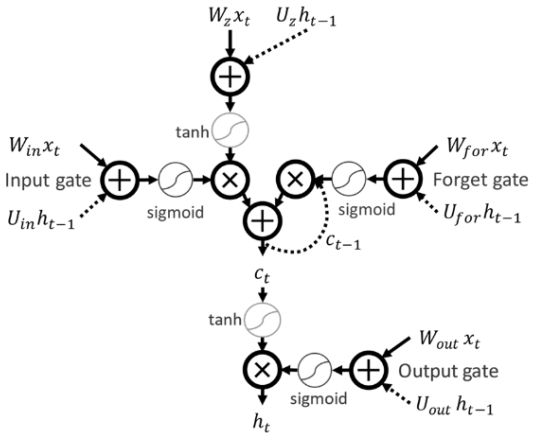

図 2.5: LSTM の概念図

LSTM は 3 つのゲート構造を持つためパラメータが Simple RNN の 4 倍となっており，アダマール積なども考慮すると計算コストはさらに高くなるため，実際の使用に関してメモリ不足や実行時間の増大などが問題となる．また行列重みパラメータを持つゲート構造を複数持つことから，ニューラルネットワーク全体として構造が複雑であり，隠れ状態や内部セルにおける明確な役割の違いを述べることは難しく，並びに各ゲートが果たす役割を明確に分離・解析することも容易ではない．

2.4 Gated Recurrent Units (GRU)

GRU は，LSTM よりも効率的な構造として提案された RNN である．

GRU[4]は以下の数式で表される．

$$r_t = \mathrm{sigmoid}(W_{re} x_t + U_{re} h_{t-1} + b_r) \tag{2.14}$$

$$u_t = \mathrm{sigmoid}\left(W_{up} x_t + U_{up} h_{t-1} + b_u\right) \tag{2.15}$$

$$\widetilde{h_t} = \tanh\left(W_{gr} x_t + U_{gr}(r_t \circ h_{t-1}) + b_{gr}\right) \tag{2.16}$$

$$h_t = u_t \circ h_{t-1} + (1 - u_t) \circ \widetilde{h_t} \tag{2.17}$$

ここで m を GRU への入力の次元，n を出力の次元とすると，$x_t \in \mathbb{R}^m$ は時刻 t での入力，$h_t \in \mathbb{R}^n$ は時刻 t での隠れ状態及び出力を表し，下付き文字に関わらず，$W \in \mathbb{R}^{n \times m}, U \in \mathbb{R}^{n \times n}$ は重み行列，$b \in \mathbb{R}^n$ はバイアス項を表す．また，r_t はリセットゲートを表し，u_t は更新ゲートの出力を表している．式(2.14)で表されるリセットゲートは，前の時刻の隠れ状態である h_{t-1} をどれだけ考慮して新しい隠れ状態である $\widetilde{h_t}$ を生成するかを制御しており，必要のない情報を効果的に削除できるため簡潔な表現が可能であるとされる．また式(2.15)で表される更新ゲートは生成した $\widetilde{h_t}$ を取り込む量を決定し，次時刻の真の隠れ状態となる h_t の生成を制御しており，現在の隠れ状態に引き継ぐべき情報量を制御することで長期依存関係を持つ情報の記憶に役立つとされる．

GRU の概念図を図 2.6 に示す．なお，バイアス項は省略している．

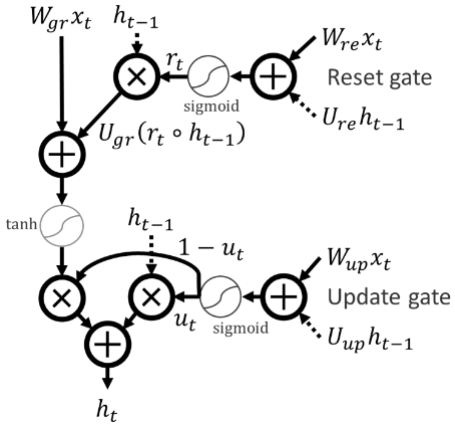

図 2.6: GRU の概念図

GRU はゲートを 2 つしか持たないことから，パラメータ量が LSTM の 3/4 であり，主に計算コスト・メモリ使用量を下げる目的で用いられるが，式(2.16)におけるリセットゲートの計算において並列性が低いため，各ゲートの行列積を一括で計算することのできる LSTM よりも実行時間が長くなってしまうケースがある．また，Simple RNN と比較した場合のパラメータは 3 倍であり計算コストは高い．加えて，LSTM と GRU はタスクによって性能が変わるため単純に優劣をつけることが難しい[14-16]．

2.5 RNN 研究の意義

前節までに述べた通り Simple RNN を使用した場合では性能が低くなってしまうが，LSTM や GRU のようなゲート付き RNN を用いた場合では計算コストやメモリ使用量が問題となるケースが多く，学習が行えない，膨大な計算時間が必要になるなどの問題が生じる可能性がある．この背景には，並列計算が高速に行えることからニューラルネットワークの計算に用いられる Graphics Processing Unit (GPU)において，使用される高速な専用 VRAM メモリが非常に高価であることが一因としてあり，その高速性を享受するために GPU と一体型となっていることから専用メモリだけを増やすことのできない難しさもある．近年多岐にわたる分野で半導体チップが必要となることから半導体チップの世界的な不足の影響もあり，VRAM の容量不足の問題は直ぐには解決しないと考えられる．大規模なデータセットへの適用も行われる中での計算コスト・メモリ使用量の制約は非常に大きく，問題への解決策が必要とされる．またゲート構造がニューラルネットワーク全体に影響を及ぼす影響についても未だ明確ではなく，更なるニューラルネットワークの発展のため，解析が必要である．

本研究では上記の問題点を踏まえ，既存の RNN モデルからのパラメータの削減と簡単化を考える．その上で新しい RNN モデルを構築してパラメータの削減を試み，LSTM 及び GRU との性能比較実験を行うことでそれを評価する．また同時に性能面での解析についても可能な限り行う．

第3章 Simple RNN から出発したゲート構造を有する RNN の構造設計と性能評価

3.1 はじめに

本章では，LSTM や GRU といった RNN モデルが持つ既存のゲート構造を用いて，それらを Simple RNN に挿入する形でゲート付き RNN の簡単化を行い，性能を保ちつつパラメータの少ない，高速で省メモリに扱うことのできる新しい RNN を構築する．新しい RNN モデルは，ゲート構造や入力に対する重みパラメータの有無による性能差を確認するため 4 種類構築される．この新たな RNN モデルの名称は以後，簡潔に表記するため，Simple Gated RNN (SGR)と呼称する．

構築した SGR については，LSTM と GRU との性能比較を行い，その性能を評価する．性能評価には複数の処理要素が含まれる混合タスクとして，最終ステップでの出力が必要(encoder)かつ各タイムステップでの出力が必要(decoder)となる RNN を用いた encoder-decoder モデルによる英語から日本語への機械翻訳を実行し，その性能を評価する[17]．

２章で述べた通り， LSTM[2,3]では入力ゲート，忘却ゲート，出力ゲートの３つのゲート構造により必要となる情報を制御するため，長期の依存関係を持つ時系列データに対する性能が良いが，一方で Simple RNN に対してパラメータ量が 4 倍となってしまう．そのため計算コストなどを下げる目的で，LSTM より効率的な構造として提案された，ゲート数が２つで LSTM の 3/4 と少ないパラメータを持つ GRU[4]が使用されることが多くなっている．しかし，GRU であっても Simple RNN の３倍のパラメータ量があり，並列性が低いなどの問題点がある．したがって，LSTM や GRU のようなゲート付き RNN を用いた場合では計算コストやメモリ使用量が問題となるため，ゲート付き RNN の簡単化及び高速化を行った MUT[8]や MGU[9]，SRU[10]などの RNN の簡単化に関する論文が発表されている．これらの論文中でいくつか提案されたモデルの中で代表的なモデルを以下に示す．

[MUT]

$$z_t = \mathrm{sigmoid}(W_{xz}x_t + b_{zz}) \tag{3.1}$$

$$r_t = \mathrm{sigmoid}(W_{xr}x_t + U_{hr}h_{t-1} + b_{rr}) \tag{3.2}$$

$$h_t = \tanh(W_{hh}(r_t \circ h_{t-1}) + \tanh(x_t) + b_{hh}) \circ z_t + h_{t-1} \circ (1 - z_t) \tag{3.3}$$

[MGU]

$$f_t = \mathrm{sigmoid}\left(W_f x_t + U_f h_{t-1} + b_f\right) \tag{3.4}$$

$$\widetilde{h_t} = \tanh(W_x x_t + U_h(f_t \circ h_{t-1}) + b_h) \tag{3.5}$$

$$h_t = (1 - f_t) \circ h_{t-1} + f_t \circ \widetilde{h_t} \tag{3.6}$$

[SRU]

$$f_t = \mathrm{sigmoid}\left(W_{sf}x_t + v_{sf} \circ c_{t-1} + b_{sf}\right) \tag{3.7}$$

$$c_t = f_t \circ c_{t-1} + (1 - f_t) \circ (W_{sx}x_t) \tag{3.8}$$

$$r_t = \mathrm{sigmoid}(W_{sr}x_t + v_{sr} \circ c_{t-1} + b_{sr}) \tag{3.9}$$

$$h_t = r_t \circ c_t + (1 - r_t) \circ x_t \tag{3.10}$$

ここで，m を各 RNN への入力の次元，n を出力の次元とすると，$x_t \in \mathbb{R}^m$ は時刻 t での入力，$h_t \in \mathbb{R}^n$は時刻 t での隠れ状態及び出力を表し，下付き文字に関わらず，$W \in \mathbb{R}^{n\times m}, U \in \mathbb{R}^{n\times n}$ は重み行列，$b \in \mathbb{R}^n$ はバイアス項を表している．また SRU における v_{sf}, v_{sr} はパラメータとなるベクトルを表す．

MUT や MGU は主に GRU ベースの構造であり，SRU は LSTM 及び GRU ベースの構造であるといえる．これらのゲート付き RNN について，MUT や SRU ではゲート構造を複数持ち依然として計算コストが高く，MUT や MGU ではゲート計算の順序の観点から並列性が低く計算時間が長くなってしまうという問題点がある．計算コストを削減するためゲート構造の重みパラメータを減らす，あるいはバイアス項のみでゲートを構成し，計算コストを削減しようとした論文もある[18, 19]が，本章における研究ではゲート構造の数を減らした新しい RNN を構築することで計算コスト削減を図る．

3.2 新しい RNN モデルの構成

まず SGR を構築するにあたり，Simple RNN の隠れ状態に対する重みパラメータが 1.0 に固定された RNN を考える．ここで，”重みを 1.0 に固定する”とは，単位行列を隠れ状態へのパラメータとして設定し，重みの学習は行わない場合と等価である．重みを 1.0 に固定した場合，入力シーケンスが急激に変化する，あるいは複数回にわたり同じ入力が与えられた場合に柔軟な処理ができないといった問題が発生する可能性がある．この問題は初期型の忘却ゲートを持たない LSTM[2]にも存在しており，問題解決のため忘却ゲートが導入された経緯を踏まえると，隠れ状態に対する重みパラメータが 1.0 に固定された Simple RNN に単一のゲート構造を挿入した RNN の性能は複数のゲートを持つ RNN に匹敵する可能性がある．したがって，ここでは隠れ状態に対する重みパラメータが 1.0 に固定された Simple RNN に単一のゲート構造を挿入したモデルを構築する．ただし，GRU のように入力および隠れ状態に対し線形補間を用いた単一のゲートを持つ場合と，LSTM のように入力および隠れ状態の両方に異なるゲートを持つ場合とで性能差が生じる可能性を考慮し，GRU と LSTM のゲート構成の両方を検証する．また，入力に対する重みパラメータの有無による性能差も確認する．

隠れ状態に対する重みパラメータが 1.0 に固定された Simple RNN は，以下の数式で表される．

$$h_t = \mathrm{act}(W_e x_t + h_{t-1} + b) \tag{3.11}$$

ここで，m を各 RNN への入力の次元，n を出力の次元とすると，$x_t \in \mathbb{R}^m$ は時刻 t での入力，$h_t \in \mathbb{R}^n$ は時刻 t での隠れ状態及び出力を表し，$W_e \in \mathbb{R}^{n \times m}$ は重み行列，$b \in \mathbb{R}^n$ はバイアス項を表している．また，図 3.1 に隠れ状態に対する重みパラメータが 1.0 に固定された Simple RNN の概略図を示す．なお，図中でのバイアス項は省略している．

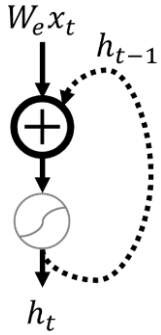

図 3.1: 重みパラメータを 1.0 に固定した Simple RNN

この RNN に対して，1 つのゲート（GRU の更新ゲートに相当）もしくは 2 つのゲート（LSTM の入力ゲートと忘却ゲートに相当）を挿入したものが SGR であり，入力に対する重みパラメータを持つモデルと持たないモデルを構築する．ここで，”重みをもたない”とは，重みパラメータを 1.0 に固定することと等価である．以下にその 4 種類の SGR の数式を示す．

SG1R：1 つのゲート構造及び入力に対して重みパラメータを持たない SGR

$$g_{1_t} = \mathrm{sigmoid}\left(W_{g1} x_t + U_{g_1} h_{t-1} + b_{g_1}\right) \tag{3.12}$$

$$h_t = g_{1_t} \circ x_t + \left(1 - g_{1_t}\right) \circ h_{t-1} \tag{3.13}$$

$$o_t = \tanh(h_t) \tag{3.14}$$

SG1RW：1 つのゲート構造及び入力に対して重みパラメータを持つ SGR

$$g_{1_t} = \mathrm{sigmoid}\left(W_{g1} x_t + U_{g_1} h_{t-1} + b_{g_1}\right) \tag{3.15}$$

$$h_t = g_{1_t} \circ (W_1 x_t + b_1) + \left(1 - g_{1_t}\right) \circ h_{t-1} \tag{3.16}$$

$$o_t = \tanh(h_t) \tag{3.17}$$

SG2R：2 つのゲート構造及び入力に対して重みパラメータを持たない SGR

$$g_{1_t} = \mathrm{sigmoid}\left(W_{g_1} x_t + U_{g_1} h_{t-1} + b_{g_1}\right) \tag{3.18}$$

$$g_{2_t} = \mathrm{sigmoid}\left(W_{g_2} x_t + U_{g_2} h_{t-1} + b_{g_2}\right) \tag{3.19}$$

$$h_t = g_{1_t} \circ x_t + g_{2_t} \circ h_{t-1} \tag{3.20}$$

$$o_t = \tanh(h_t) \tag{3.21}$$

SG2RW：2 つのゲート構造及び入力に対して重みパラメータを持つ SGR

$$g_{1_t} = \mathrm{sigmoid}\left(W_{g_1} x_t + U_{g_1} h_{t-1} + b_{g_1}\right) \tag{3.22}$$

$$g_{2_t} = \mathrm{sigmoid}\left(W_{g_2} x_t + U_{g_2} h_{t-1} + b_{g_2}\right) \tag{3.23}$$

$$h_t = g_{1_t} \circ (W_1 x_t + b_1) + g_{2_t} \circ h_{t-1} \tag{3.24}$$

$$o_t = \tanh(h_t) \tag{3.25}$$

ここで，m を各 RNN への入力の次元，n を出力の次元とすると，$x_t \in \mathbb{R}^m$ は時刻 t での入力，$h_t \in \mathbb{R}^n$は時刻 t での隠れ状態を表し，下付き文字に関わらず，$W \in \mathbb{R}^{n\times m}, U \in \mathbb{R}^{n\times n}$ は重み行列，$b \in \mathbb{R}^n$ はバイアス項を表している．また，g_{1_t}, g_{2_t} はゲートからの出力を表しており，o_t は隠れ状態に対して活性化関数を適用した後の下層への出力である．なお，SGR に関しては同じ機能を持つものに限り同じ添え字を用いている．図 3.2 に構築した 4 種の SGR についての概略図をそれぞれ示す．なおここでもバイアス項は省略している．

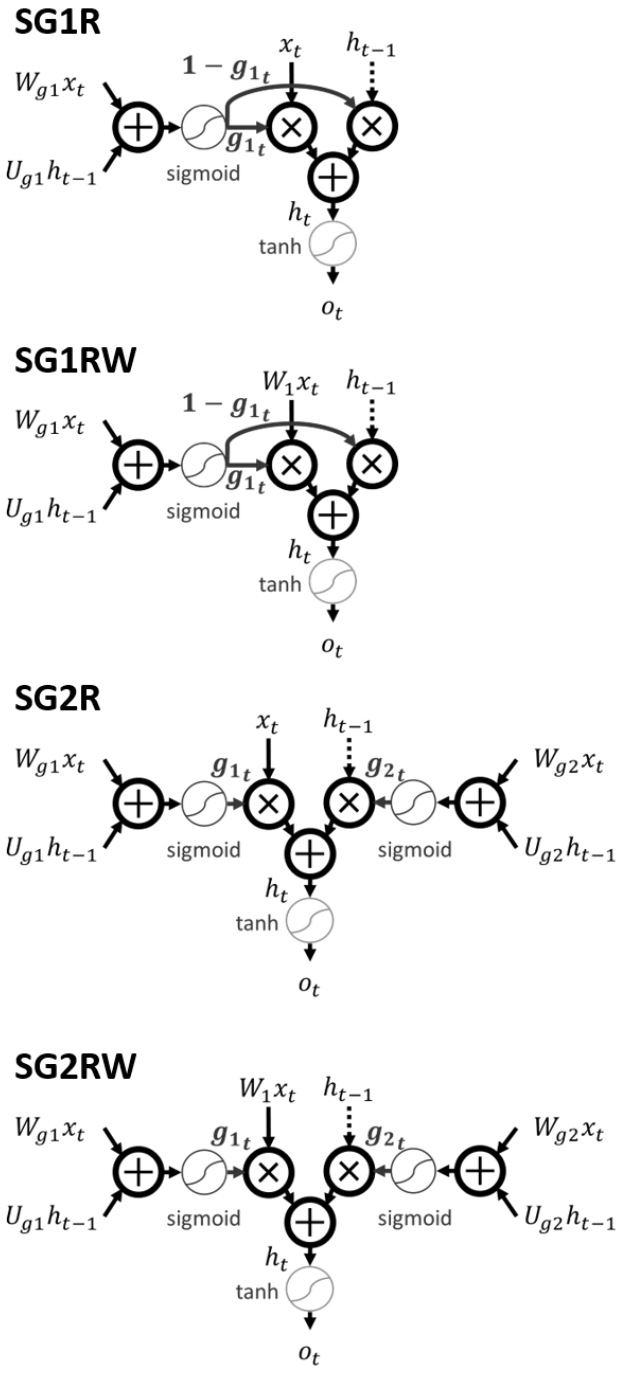

図 3.2: 構築した 4 種類の SGR

SG1R と SG1RW は，隠れ状態に対する重みパラメータが 1.0 に固定された Simple RNN に GRU の更新ゲートを挿入して構成されたものであり，SG2R と SGR2W は LSTM の入力ゲートと忘却ゲートを挿入する形で構成されたものである．また，RNN への入力とゲートをアダマール積によって計算することは重み付きの非線形な計算であるため，入力に対する重みパラメータが不要である可能性がある．そのため，入力に対する重みを 1.0 に固定した構造を構築した．SG1R と SG2R は入力に対し重みパラメータをもたないモデルであるが，SG1RW と SG2RW は入力に対して重みパラメータ W_1 をもつモデルとなっている．ここで単純に考えた場合，重みパラメータが増加すると性能が向上するものと想定される．活性化関数を RNN の再帰構造の外部に挿入し，下層への出力時に隠れ状態に対して適用するように構築した理由は LSTM の CEC と同様に活性化関数による勾配消失問題などの学習の不安定さに対処するためである．

3.3 性能評価実験

3.3.1 性能評価の概要

4 種類の SGR の性能評価を行うために，機械翻訳による性能評価を行う．機械翻訳に用いる encoder-decoder モデルでは，2 値分類に代表される最終時刻の出力を用いて学習が行われる encoder 部と，文章生成に代表される各時刻での出力のすべてを用いて学習が行われる decoder 部の 2 つの RNN モデルにより構成される．ゆえに encoder-decoder モデルを用いた機械翻訳タスクは，複数の異なる性質のタスクが合わさる優れた性能評価タスクであると考えられる．ここでの機械翻訳における使用言語は英語と日本語である．なお，計算機を用いた機械翻訳では，Python 3.7 と機械学習ライブラリである chainer v6.4.0 と Cuda 10.1 を用いて単精度浮動小数点演算により実行される．

3.3.2 機械翻訳モデル

機械学習には，先ほども述べた通り RNN を用いた encoder-decoder モデル[20]を使用する．encoder-decoder モデルでは，入力された翻訳元の文章を特徴量ベクトルとして RNN の隠れ状態にマッピングする encoder 部と，encoder 部がマッピングした特徴量ベクトルを RNN の隠れ状態として用いて出力を生成する decoder 部を持つ．ここで，encoder-decoder モデルは sequence-to-sequence（seq2seq）モデルと呼ばれることもある．モデルへの入力は，分かち書きされた単語単位で行われ，出力は単語の確率分布として生成されたものから 1 つずつ単語を選び出力していく．単語の選択は確率分布の中からランダムに選ぶ，もしくはビームサーチや最大値を用いる方法など複数あるが，本研究に用いるモデルでは，より分かりやすくモデルを評価するため確率分布の中で最大のものを選択する．

encoder 部は単語埋め込み層（embedding layer）と複数のスタックされた RNN 層で構成される．入力時の埋め込み層では，それぞれの入力単語の one-hot ベクトルを分散表現ベクト

ルとしてマッピングを行う．ここで one-hot ベクトルとは，1 つの要素が 1，それ以外の要素が 0 となるベクトルであり，各単語の ID に相当するベクトルの次元の要素が 1 となる．例えば，すべての文章が 5 つの単語で構成され，2 番目の単語 ID を表す場合であれば，$[0,1,0,0,0]$ のように表される．one-hot ベクトルでは単語すべてを平等に扱うことができ，スパースかつ簡潔な表現が可能であるが，すべての文章で出現する単語数がベクトルの次元となるため，次元が非常に大きくなってしまう欠点がある．ゆえに単語埋め込み層を用いて，密なベクトルとして分散表現ベクトルに単語をマッピングすることで次元削減を行う．その後，分散表現ベクトルは単語ごとに RNN へと入力され，隠れ状態に文章表現ベクトルとしてマッピングされる．一方 decoder 部は，単語埋め込み層と複数のスタックされた RNN 層，及び出力層で構成される．単語埋め込み層は，翻訳先の言語が異なるため encoder とは異なる重みを持つものを用いる．スタックされた RNN 層では，encoder がマッピングした隠れ状態を初期状態とする．ここで encoder においてスタックされた RNN の隠れ状態は，対応する同じ位置の decoder の各 RNN に初期状態として設定される．出力層では RNN からの出力を受け取り，単語 ID の数と同じ次元のベクトルを出力する．この出力を softmax 関数に通すことで単語の出現確率へと変換し，最も出現確率の高い単語に相当する要素の単語 ID を選択する．N 次元の出力ベクトル o の i 番目の要素に対応する softmax 関数適用後の要素は以下に示す数式で表される．

$$\mathrm{softmax}(o_i) = \frac{\exp(o_i)}{\sum_{j=1}^{N} \exp(o_j)} \tag{3.26}$$

decoder には，encoder からの隠れ状態を初期状態として引く継いだ後，文の終わりを意味する end of sentence (EOS)というトークンが入力されることで出力単語 ID の生成を開始し，decoder から EOS が出力された時点で単語 ID の生成を終了する．encoder-decoder モデルの概念図を図 3.3 に示す．

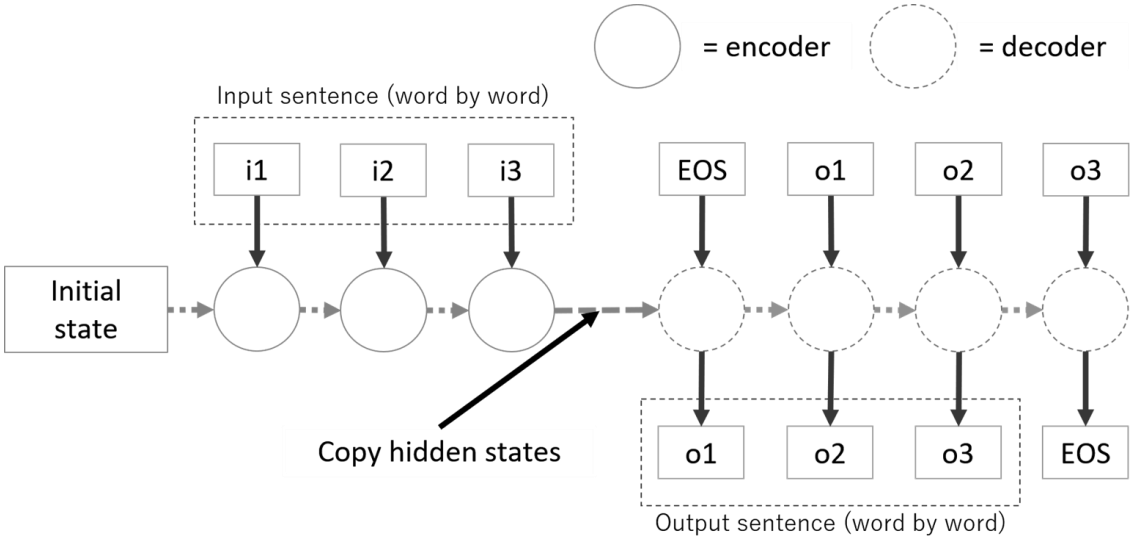

図 3.3: encoder-decoder モデルの概念図

3.3.3　機械翻訳に用いるコーパスと前処理

機械翻訳におけるコーパスには”Tanaka corpus”[21]を用いて英語から日本語への翻訳タスクを行う．コーパスにおける翻訳ペア数は 149786 対であり，前処理として日本語文章は Mecab[22]，英語文章は NLTK[23]を用いて単語単位に分割する．これにより句読点やピリオドを含め英単語数は 25722 語，日本語の単語数は 31809 語となる．これに，文の終了を意味する EOS を加えたものが総単語数となる．

翻訳ペア数のうち 90%を訓練データセット，残りの 5 %ずつを検証データデータセットとテストデータセットに分割する．訓練にはデータセットに含まれるすべての単語を含め語数の制限などは特に行わない．

3.3.4　機械翻訳による性能評価実験の設定

本章の実験に使用される encoder-decoder モデルの各 RNN において，ユニット数による性能差を確認するため，入出力の次元は 250，500，1000 の 3 パターンを設定する．またそれに伴い，埋め込み層の出力次元や出力層の入力次元も 250，500，1000 に設定される．スタックする RNN の層数は，RNN ごとに 1 層から 3 層までとし，encoder と decoder は同じ層数となる．学習回数は 25 回，ミニバッチサイズは 256 に設定する．

学習に使用する確率的勾配降下法のアルゴリズムは AdamW 法[24]を用いる．設定されるハイパーパラメータは，過学習を防ぐため荷重減衰(weight decay)を 1e-4 とし，その他は推奨パラメータである $\alpha = 0.001,\ \beta_1 = 0.9,\ \beta_2 = 0.999$ とした．各層には過学習を防ぐための dropout[25]を設定し，dropout 率は 0.2 とする．ここで，RNN においては dropout を入力に対してのみ行い，隠れ状態に対しては行わない．勾配爆発を防ぐための Gradient Clipping のパラメータは 25 とする．各層の初期化に関して，重みは平均が 0，分散が $\sqrt{1/n}$ (n:ユニット数)となるガウス分布で初期化[26]して，バイアス項はすべて 0 に初期化する．

学習に用いる損失関数には，softmax cross entropy を用い，softmax cross entropy (L)は以下に示す数式で表される．

$$L = -\frac{1}{m}\sum_{i,j} t_{i,j} \log\left(\frac{\exp(y_{i,j})}{\sum_{k=1}^{n}\exp(y_{i,k})}\right) \tag{3.27}$$

ここで，$m,\ n$ はそれぞれバッチサイズと出力サイズ，i はデータの番号，j は次元であり，$y_{i,j},\ t_{i,j}$ はそれぞれ出力層からの出力と教師ラベルを表している．例えば，$t_{i,j}$ は i 番目のデータの j 次元目の教師ラベルとなる．

実験において VRAM メモリが足りずメモリエラーが出ることを防ぐため，chainer の softmax cross entropy 関数では中間キャッシュを保持しないように設定した．また同様の理由により，LSTM が 3 層かつユニット数 1000 の場合のバッチサイズは 128 に設定した．実験は重みや確率的勾配降下法のランダム性を考慮して 2 回行われ，その平均で評価する．

3.3.5　性能評価指標

性能評価指標として，Bilingual Evaluation Understudy (BLEU)[27]を用いる．BLEU は，n-gram の一致率を基に，機械翻訳による翻訳文と参照訳との類似度を算出する．

BLEU は以下に示す数式で表される．

$$\mathrm{BLEU} = \mathrm{BP} \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right) \tag{3.28}$$

$$\mathrm{BP} = \begin{cases} 1 & , \text{ if } c > r \\ e^{1-r/c} & , \text{ if } c \leq r \end{cases} \tag{3.29}$$

$$p_n = \frac{\sum_{C\in\text{全訳数}} \sum_{n-gram\in C} Count_{clip}(n-gram)}{\sum_{C'\in\text{全訳数}} \sum_{n-gram'\in C'} Count(n-gram')} \tag{3.30}$$

ここで，c は機械翻訳結果となる文の長さ，r は参照訳の長さであり，BP (brevity penalty) は参照訳の長さよりも機械翻訳結果となる文の長さが短くなった場合のペナルティ項となる．p_n は長さ N の n-gram の精度を表し，w_n は合計が 1 となるように設定される重みである．p_n の分子は，翻訳文と参照訳で一致した n-gram 数を計算し，分母は翻訳文の全 n-gram 数を計算したものである．ただし分子の計算では，翻訳文において同じ単語が何度も繰り返されることにより精度が高くなってしまう問題点を解消するため，修正された n-gram 精度が使用される．修正された n-gram 精度とは，翻訳文における同一単語の出現回数が参照訳に含まれる同一単語の出現回数の最大値を超える場合には，参照訳の最大値に揃える(clip)といった修正が行われるというものである．1-gram の精度は単語訳の正しさを，高次の N-gram の精度は翻訳の流暢さを示す指標となる．本章における実験では，推奨パラメータである $N = 4$ と均一な重み $w_n = 1/N$ を使用した．

3.3.6　プログラムのフローチャート

図 3.4 にプログラムの大まかなフローチャート図を示す.

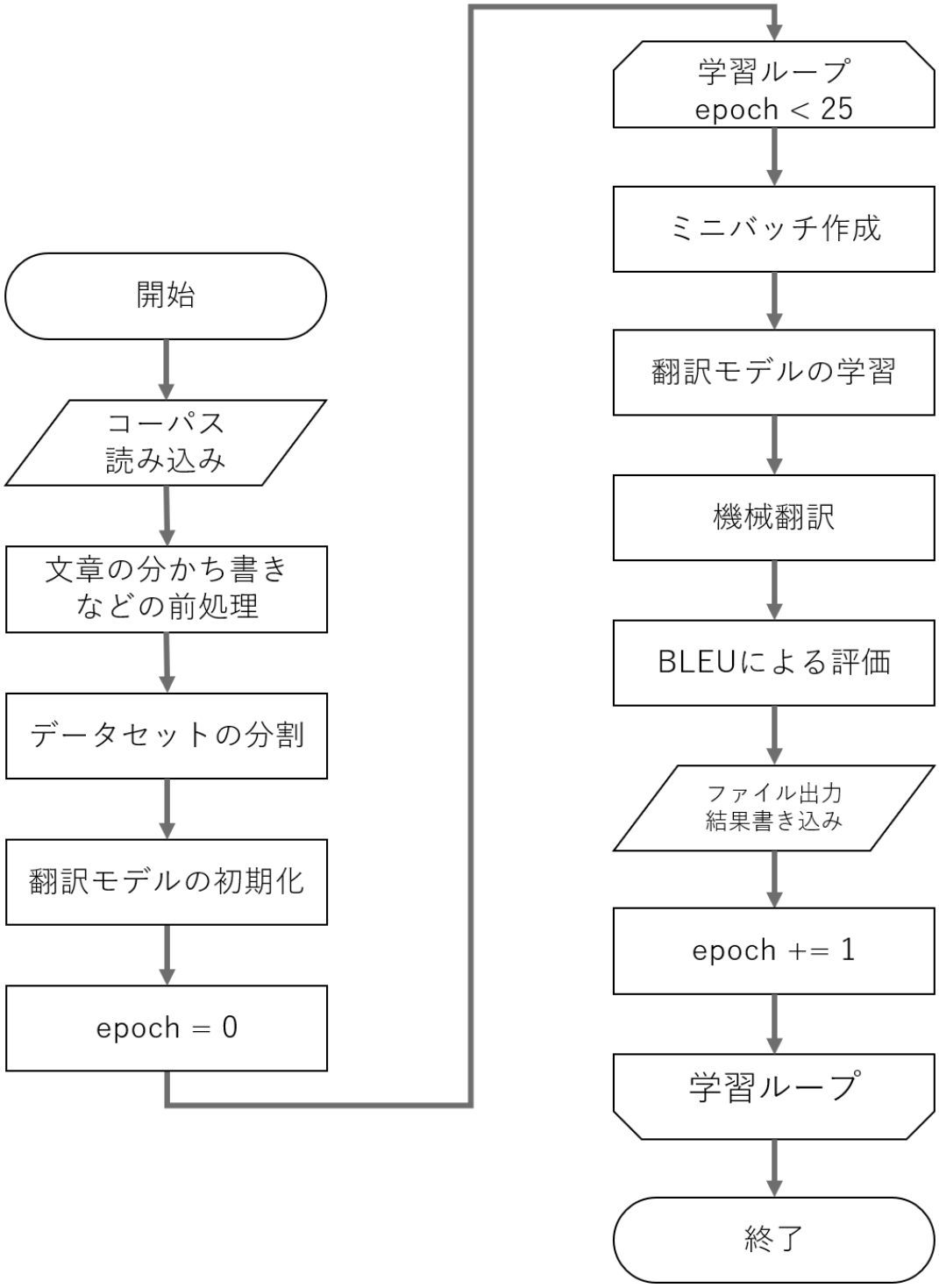

図 3.4: プログラムの大まかなフローチャート図

3.4 機械翻訳における性能評価の結果

表 3.1 に実験結果となる BLEU スコア(%)を示す．表 3.1 では，25 回の学習回数の中で検証データセットに対して最も loss が低くなった場合のテストデータのスコアを示している．なお，スコアは 2 回の実験の平均と標準偏差で表され，小数点第 3 位を四捨五入している．またユニット数を 1000 とした場合，各 RNN を用いた場合の実行完了までに必要な時間のグラフを図 3.5 に，層数の増加とともに必要とされる RNN の行列重みパラメータについてのグラフを図 3.6 に示す．なお，encoder-decoder モデルにおいて同じ層数の RNN が使用されるため, 実際に使用されている行列重みパラメータは図 3.6 に示される値の 2 倍である．また LSTM は単一カーネルでの実装であり，SGR と GRU は chainer function の組み合わせで実装されていることから速度に関する単純比較ができない. そのため図 3.5 においては各 SGR と GRU の実行時間のみを示している．

表 3.1 より，最も BLEU スコアが高いモデルは 3 層: 1000 units の SG1R であり，次いで 2 層: 1000 units の SG1R，3 番目に 2 層: 1000 units の SG2R となっている．しかし，2 層: 1000 units の SG1R と 2 層: 1000 units の SG2R におけるスコアの差は小さい．LSTM や GRU との比較では，LSTM や GRU よりも 1000 units の SGR におけるスコアは総じて高い．図 3.6 より，SG2R の重みパラメータは，LSTM の 1/2，GRU の 2/3 であり，SG1R の重みパラメータは，LSTM の 1/4，GRU の 1/3 である．よって，隠れ状態に設定されるユニット数が同じである場合，SGR では LSTMや GRU と比較して重みパラメータが少なくなる．また図 3.5 より，SGR のパラメータが少なくなると実行時間も少なくなっており，GRU よりも高速に実行できることがわかる．ここでパラメータ量と実行速度が完全に比例しないのは入力シーケンスの転置やバッチ学習，CPU-GPU 間の転送オーバーヘッドなどによるものである．

表 3.1: 実験結果: BLEU スコア(%)

BLEU score (%)		Hidden units		
		250 units	500 units	1000 units
Simple RNN with a fixed weight of 1.0	1 layer	14.34±0.12	16.76±0.09	16.06±0.09
	2 layers	15.45±0.14	17.46±0.18	16.42±0.18
	3 layers	13.56±0.00	15.97±0.29	14.85±0.14
Simple RNN	1 layer	9.04±0.27	9.91±0.44	9.31±0.06
	2 layers	9.82±0.05	10.33±0.23	8.58±0.07
	3 layers	8.87±0.31	9.46±0.16	7.16±0.14
SG1R	1 layer	16.73±0.15	20.68±0.20	23.04±0.05
	2 layers	18.00±0.09	22.16±0.06	24.62±0.02
	3 layers	17.11±0.09	21.66±0.08	**24.83±0.04**
SG1RW	1 layer	19.03±0.11	21.42±0.08	22.50±0.24
	2 layers	18.00±0.09	22.55±0.02	22.90±0.15
	3 layers	19.77±0.26	22.71±0.21	22.29±0.30
SG2R	1 layer	20.33±0.04	23.40±0.10	24.60±0.18
	2 layers	20.15±0.03	23.01±0.02	24.62±0.07
	3 layers	19.65±0.12	22.55±0.10	24.34±0.04
SG2RW	1 layer	20.35±0.02	22.56±0.38	23.06±0.02
	2 layers	20.44±0.22	22.60±0.05	22.66±0.06
	3 layers	20.53±0.04	23.05±0.11	20.65±0.04
LSTM	1 layer	16.35±0.36	19.47±0.05	20.28±0.34
	2 layers	18.00±0.09	20.42±0.56	20.99±0.58
	3 layers	16.56±0.21	18.92±0.45	19.97±0.52
GRU	1 layer	19.55±0.04	22.04±0.07	22.92±0.10
	2 layers	20.69±0.10	22.77±0.06	23.31±0.31
	3 layers	21.79±0.09	24.07±0.07	23.44±0.21

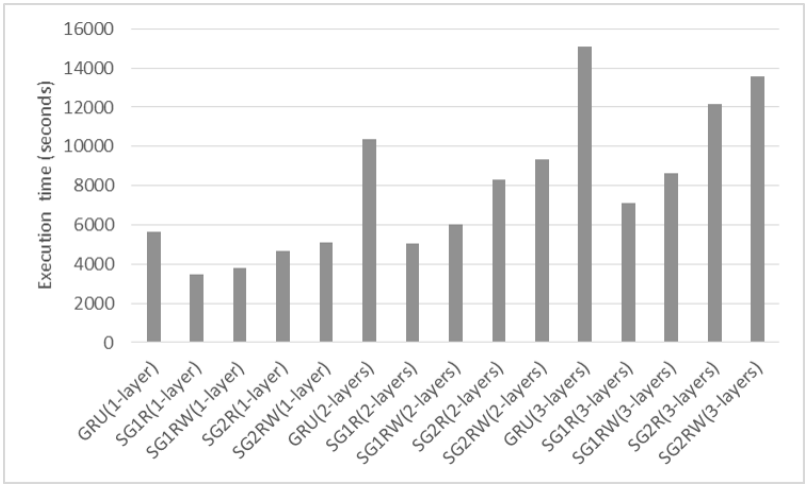

図 3.5: 各 RNN を用いた場合の実行完了までに必要な時間

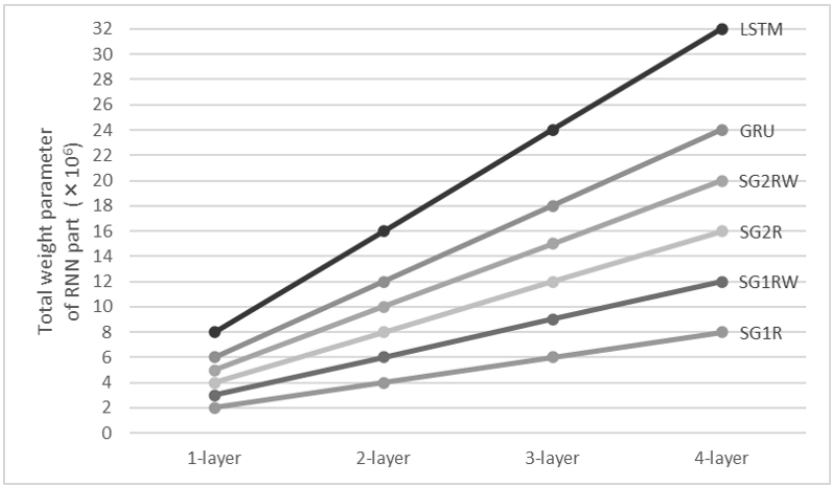

図 3.6: 層数の増加とともに必要とされる RNN の行列重みパラメータ

3.5 考察

表 3.1 の実験の結果から，ユニット数の少ない 250 units の場合では GRU が SGR を上回る BLEU スコアとなっており，SGR 同士の比較においても入力に対して重みパラメータを持つ SG1RW と SG2RW は，重みを持たない SG1R と SG2R に比べてスコアが高いことが確認できる．一般的には，重みパラメータの数が多いほど性能は高くなると仮定されるため妥当な結果である．また 500 units の場合においても 1 層と 2 層の SG2R を除き，概ね同様の傾向が見られる．しかし 1000 units の場合では，入力に対して重みパラメータの無い SG1R と SG2R が他のモデルと比較してスコアがより高くなっていることが確認できる．ここで，図 3.7 にユニット数を 1000 units とした時の LSTM と SG2RW，SG2R を用いた場合の学習曲線を，図 3.8 にユニット数を 1000 units とした時の GRU と SG1RW，SG2R を用いた場合の学習曲線を示す．

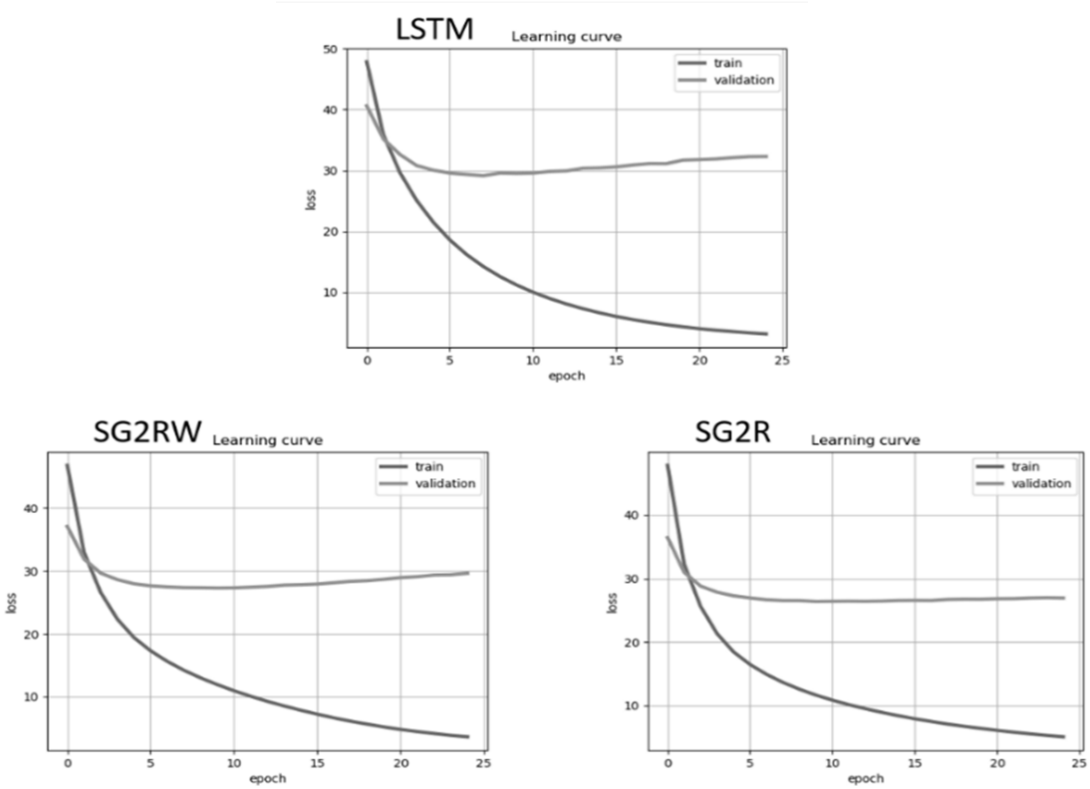

図 3.7: LSTM と SG2RW，SG2R を用いた場合の学習曲線

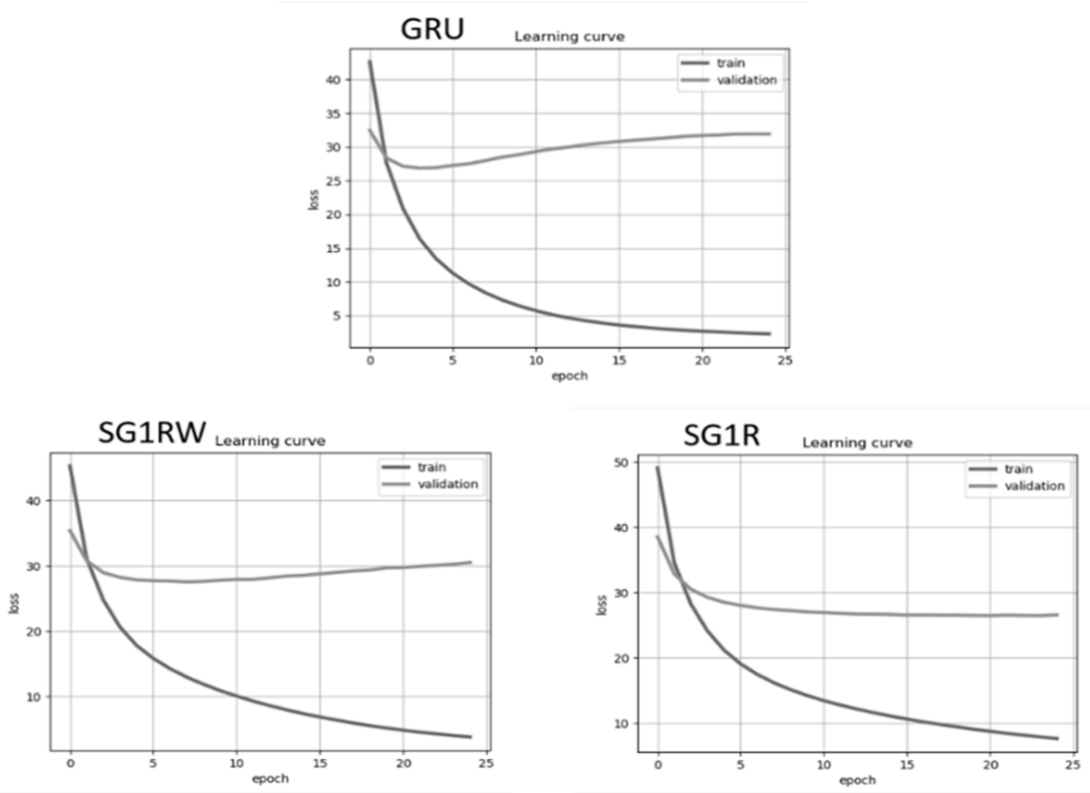

図 3.8: GRU と SG1RW，SG1R を用いた場合の学習曲線

図 3.7 及び図 3.8 を見れば SG1R と SG2R は学習曲線において過学習の兆候が見られず，非常に緩やかに学習が進んでいく様子が見て取れる．したがって，重みパラメータのない SG1R と SG2R では過学習を起こしにくく，ユニット数が増えるにつれて精度が向上したものと考えられる．この理由としては，重みパラメータがゲート構造にしか存在しないことから，sigmoid 関数を微分して得られる緩やかな勾配により過学習が起こりにくい状態であったからだと推測できる．したがって，SG1RW と SG2RW は比較的過学習を起こしにくいと考えられるユニット数の少ない場合で使用でき，ユニット数を十分に確保できる場合には SG1R や SG2R を使用することで過学習を抑えながら精度を向上させることができると想定される．ゆえに，より大きなモデルが必要とされる場面においてパラメータ量を削減することができ，計算コスト・メモリ使用量の削減が期待できる．

250 units の GRU と 1000 units の SG1R のパラメータを比較した場合では，250 units の GRU におけるパラメータ量は，1000 units の SG1R よりも小さくなるが，SG1R は並列性が高く，GPU を用いることで GRU より高速に計算することが可能である．実行速度の例として，25 回の学習が終了するまでに 250 units の GRU は 5200 秒かかったのに対して，1000 units の SG1R は 3468 秒となる．これは，SGR を用いる明確な利点になる．

1 つのゲート(GRU の更新ゲートに相当)もしくは 2 つのゲート(LSTM の入力ゲートと忘却ゲートに相当)を用いた場合の優劣について，比較的小さい重みパラメータの場合では 2 ゲートのスコアがより高く，大きい重みパラメータの場合では 1 ゲートのスコアが高くなった．2 ゲートの SGR では 1 ゲートの SGR に対してゲート計算に必要な計算コストやメモリ使用量が 2 倍となってしまうため結果から単純に優劣をつけることは難しく，更なる実験及び検証が必要であると考えられる．

1 つのゲート構造を用いた場合に関して，時刻 t でのゲートへの入力を in_t とおくと，ゲート計算における式は以下のように変形できる．

$$\begin{aligned} h_t &= g_{1_t} \circ in_t + \left(1 - g_{1_t}\right) \circ h_{t-1} \\ &= h_{t-1} + g_{1_t} \circ (in_t - h_{t-1}) \end{aligned} \tag{3.30}$$

近年注目されている shortcut connection[7]は，ニューラルネットワークに変換の残差を学習させることで性能を向上させるというものであり，上記の式(3.30)は，この shortcut connection に非常によく似た構造となっている．したがって，単一ゲート構造のみで性能が発揮できた理由の 1 つとして shortcut connection のように学習を行うことができたからではないかということが考えられる．

3.6 おわりに

本章では，LSTM や GRU といった RNN モデルが持つ既存のゲート構造を用いて，それらを隠れ状態に対する重みパラメータが 1.0 に固定された Simple RNN に挿入する形でゲート付き RNN の簡単化を行い，性能を維持しつつパラメータを削減し，高速かつメモリ消費の少ない新しい RNN の構築を図った．その結果として，比較的小さなコーパスにおいてではあるが複合タスクである機械翻訳タスクにおいて，構築した SGR が LSTM や GRU よりも優れた精度となった．

一方で多層化した場合の更なる精度向上や，1 ゲートもしくは 2 ゲートの場合の優位性に関してはさらに検討が必要である．また，SGR の精度が GRU の精度を上回るためには比較的大きな重みパラメータが必要となり，大幅にパラメータが削減されたとは言い難いのも確かである．これらの問題点を解消するためには，ゲート構造そのものが性能に及ぼす影響について考慮し，パラメータを削減する必要がある．次章ではゲート構造が性能に及ぼす影響について考察し，ゲート構造に改善を加えることでより大きなパラメータの削減を図る．

第4章 sech 関数を用いたゲート構造及び shortcut connection を持つ RNN におけるパラメータ削減と機能解析

4.1 はじめに

第 3 章より，LSTM や GRU で用いられるような従来型のゲート構造を用いた RNN の簡単化では性能確保を確保するために比較的大きな重みパラメータが必要であり，多層化やゲート構造の差による性能解析が難しいため，RNN モデルのパラメータをより大幅に削減するにはゲート構造が性能に及ぼす影響を考慮した上で，パラメータの削減について更に検討する必要があることがわかった．

従来のゲート構造は行列重みパラメータ持つため解析が難しく，パラメータを単純に削減することも難しい．そこで本章ではゲート構造の利点である隠れ状態のスケーリング機能に着目し，解析な観点からゲート構造を簡素化した新たな RNN を構築した上で，ゲート構造が性能に及ぼす影響と更なるパラメータ削減の余地について検討する．そのために，ゲート構造から行列重みパラメータを取り除き，従来用いられてこなかった活性化関数をゲート構造に用いることでパラメータ量の少ない RNN を構築する．そして新しく構築した RNN に対して計算機を用いた実験を行う．それによって従来型のゲートを持つ LSTM や GRU との性能比較や，ゲート構造を解析及び調査する[28]．

第 3 章の 3.5 節で述べた通り，単一ゲート構造の SGR が性能を発揮できた理由としてゲート構造そのものが shortcut connection と非常によく似た構造であるためとの仮説を立てた．したがって本章で構築する新たな RNN は，shortcut connection を持つ単純な構造の RNN にゲート構造を挿入する形で構築し，実験を行う．

shortcut connection は，152 層と非常に深い層を持つ ResNet[7]の中で用いられ注目を集めている．層が深いモデルにおいてはパラメータの増加に関わらず精度が頭打ちとなり，層の浅いモデルよりも精度が悪化してしまうという degradation 問題がある．この問題に対して，ニューラルネットワークの変換における残差を学習させることで性能の向上を助けるのが shortcut connection である．求めたい関数を$H(x)$，入力をx，ニューラルネットワークによる変換を$F(x)$ とすると shortcut connection は以下の数式で表される．

$$H(x) = F(x) + x \tag{4.1}$$

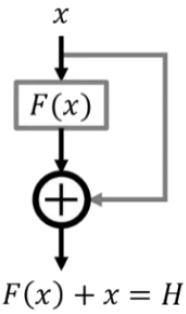

図 4.1: shortcut connection

shortcut connection の図を図 4.1 に示す．近年では，shortcut connection が勾配消失を防ぐことで性能の向上を助けることを示唆する論文も発表されている[29,30]．主に Convolutional Neural Network (CNN)などの画像処理において使用される構造ではあるが，この shortcut connection から着想を得た Res-RNN[11]や IndRNN[12]などの RNN が提案されている．これらのモデルでは，従来型のゲート構造と組み合わせたり，batch normalization[13]などの正規化手法と組み合わせたりといった方法で性能確保を行っている．しかし，これらの手法では従来のゲート構造による計算コストや，batch normalization におけるバッチサイズへの依存性並びに挿入位置をタスクによって変えなければならないなどの問題点がある．本章における RNN モデルの構築では，更にパラメータの少ない，より簡潔かつ画一的に使用可能なモデルの構築を目指す．

次節でモデル構築の詳細を述べる．

4.2 sech 関数を用いたゲート構造及び shortcut connection を有する RNN モデルとその概要

モデルの構築に関して，RNN は時刻方向に対して層が深いニューラルネットワークであるため，時刻方向への shortcut connection の適用により性能を向上させることができると考えられる．この時，単純に shortcut connection を挿入した RNN ではタイムステップ数が非常に長い場合，隠れ状態が発散してしまう可能性がある．従来型のゲート付き RNN であれば，$(0,1)$ の範囲の出力を持つゲート構造が隠れ状態を適切にスケーリングすることで発散を防いでいるため，このことは問題とはならない．したがって本章におけるモデル構築では，ゲート構造におけるスケーリング機能に着目し，shortcut connection にゲート構造を挿入して RNN モデルを構築する．

SGR が持つような行列重みパラメータを用いた従来型のゲート構造では計算コストが高く，ゲートにおける解析も難しいため，パラメータを削減することが困難である．加えて，行列重みパラメータを持たせない場合には，ゲート構造に用いられる sigmoid 関数が偶関数でないことから，負の値と正の値に対して等しくスケーリングさせることができない．これは，隠れ状態の要素が負であれば非常に小さな隠れ状態となってしまう場合や，正であれば非常に大きな隠れ状態となり発散してしまう場合を想定している．この問題点に対し，本章における RNN モデルでは sigmoid 関数に変わり，偶関数かつ出力が $(0,1]$ の範囲となる sech 関数を用いてゲート構造を構築することで解決を図る．

sech 関数は以下の数式で表される．

$$\mathrm{sech}(x) = \frac{2}{e^{x} + e^{-x}} \tag{4.2}$$

また，sech 関数を微分すると以下の数式のようになる．

$$\frac{d}{dx}(\mathrm{sech}(x)) = -\mathrm{sech}(x)\cdot\tanh(x) \tag{4.3}$$

sech 関数は偶関数であるため，正負どちらの値に対しても等しく隠れ状態をスケーリングすることが可能であると考えられる．図 4.2 に sech 関数と微分値のグラフを示す．

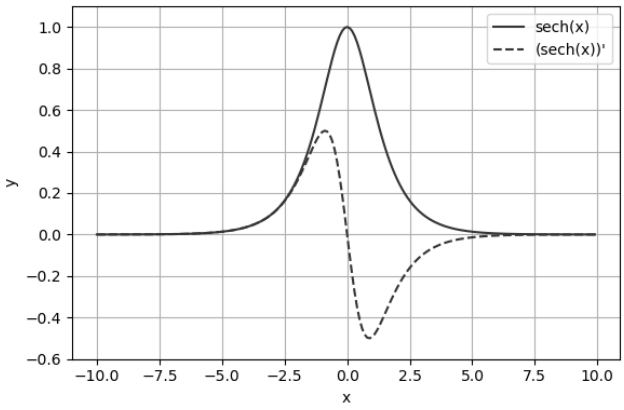

図 4.2: sech 関数とその微分値

RNN モデルは，隠れ状態に対する重みパラメータの有無や，活性化関数の位置による性能差を確認するため 4 種類のモデルを構築する．構築したモデルの数式を以下に示す．

model 1 pre-activation

$$\tilde{h}_{t-1} = \mathrm{sech}(ah_{t-1}) \circ h_{t-1} \tag{4.4}$$
$$h_t = \mathrm{act}\left(W_x x_t + U_h \tilde{h}_{t-1} + b\right) + \tilde{h}_{t-1} \tag{4.5}$$
$$o_t = h_t \tag{4.6}$$

model 1 post-activation

$$\tilde{h}_{t-1} = \mathrm{sech}(ah_{t-1}) \circ h_{t-1} \tag{4.7}$$
$$h_t = \left(W_x x_t + U_h \tilde{h}_{t-1} + b\right) + \tilde{h}_{t-1} \tag{4.8}$$
$$o_t = \mathrm{act}(h_t) \tag{4.9}$$

model 2 pre-activation

$$\tilde{h}_{t-1} = \mathrm{sech}(ah_{t-1}) \circ h_{t-1} \tag{4.10}$$
$$h_t = \mathrm{act}(W_x x_t + b) + \tilde{h}_{t-1} \tag{4.11}$$
$$o_t = h_t \tag{4.12}$$

model 2 post-activation

$$\tilde{h}_{t-1} = \mathrm{sech}(ah_{t-1}) \circ h_{t-1} \tag{4.13}$$
$$h_t = (W_x x_t + b) + \tilde{h}_{t-1} \tag{4.14}$$
$$o_t = \mathrm{act}(h_t) \tag{4.15}$$

ここで，m を各 RNN への入力の次元，n を出力の次元とすると，$x_t \in \mathbb{R}^m$ は時刻 t での入力，$h_t \in \mathbb{R}^n$は時刻 t での隠れ状態を表し，$W_x \in \mathbb{R}^{n\times m}, U_h \in \mathbb{R}^{n\times n}$ は重み行列，$b \in \mathbb{R}^n$ はバ

イアス項を表している．また a は学習によって値が変化するスカラー値であり，勾配が消失する際の度合いを制御するためのパラメータとして導入した．model 1 は隠れ状態に対して行列重みパラメータ U_h を持つモデルであり，model 2 は隠れ状態に対して行列重みパラメータを持たないモデルである．活性化関数の位置について，shortcut connection の加算前に活性化関数を適用したモデルが pre-activation モデル，shortcut connection の加算後に下層への出力にのみ活性化関数を適用するモデルを post-activation モデルとする．post-activation モデルでは，再帰構造の外部にのみ活性化関数が適用される形となる．なおここでは，model 1 と model 2 ともに同じ機能を持つものに限り，同じ添え字を用いており，o_t は下層への出力を示している．図 4.3 に各モデルの図を示す．図中でのバイアス項は省略している．

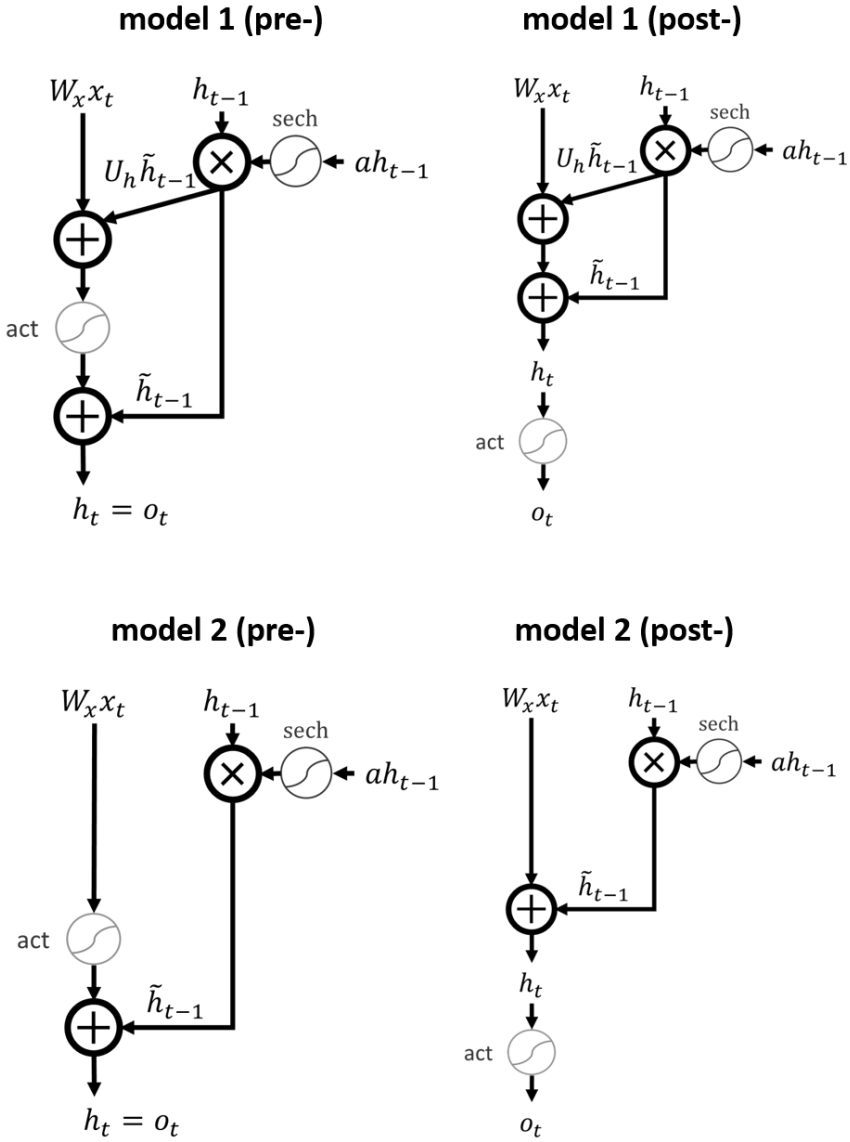

図 4.3: 構築した 4 種類の RNN モデル

4.3 実験の概要と性能評価

4.3.1 実験の概要

本章で構築したモデルの性能評価では，2 つのコーパスを用いた性質の異なるタスクによる計算機実験を行う．具体的にはコーパスに WikiText-2[31]を用いた言語モデリングタスクと，コーパスに IMDB[32]を用いた 2 値分類タスクを通じてモデルの性能を検討する．WikiText-2 を用いた言語モデリングタスクでは，各々のタイムステップごとに単語出力が必要となるため，文章における長期の依存関係に加えて単語間の短い依存関係もとらえる必要がある．一方で，IMDB を用いた 2 値分類タスクでは判定結果となる最終出力が重要であるため，文章における長期の依存関係のみを捉えるタスクである．

実験は，Python 3.8 と機械学習ライブラリである Pytorch 1.4.1 と Cuda 10.1 を用いて単精度浮動小数点演算により実行される．なお本章における実験では，cudnn は使用せず，すべての乱数 seed は 123 に初期化した．また 4.2 節で構築した 4 種類の RNN モデルにおけるスカラー値 a はすべて 1.0 で初期化され，出力に用いられる活性化関数には tanh を使用する．

4.3.2 WikiText-2 を用いた言語モデリングタスクにおける性能評価

WikiText-2[31]は，Wikipedia から抽出された言語モデリングデータセットであり，訓練データが 600 記事，検証データとテストデータがそれぞれ 60 記事ずつ用意されたものである．データセット全体における語彙は 33278 語であり，訓練データと検証データ，テストデータにおける総単語数はそれぞれ 2088628 トークン，217646 トークン，245569 トークンとなっている．

WikiText-2 を用いた言語モデリングタスクに使用する RNN を用いたニューラルネットワークモデルは，入力層，RNN 層，出力層を持ち，入出力のサイズは語彙サイズと同じ 33278 語である．また，中間ユニット数は，64，128，256 の 3 種類に設定する．実験ではデータセットにおける文章サイズが大きいため，BPTT 法における逆伝搬のタイムステップ数を特定の長さ τ に限定し BPTT の近似計算を行う truncated BPTT を用いる．ここでは $\tau = 35$ として設定する．確率的勾配降下法には，3 章でも用いた AdamW 法[24]を用いる．AdamW 法のパラメータは，荷重減衰(weight decay)を 1e-2 とし，その他は推奨パラメータである $\alpha = 0.001,\ \beta_1 = 0.9,\ \beta_2 = 0.999$ とした．過学習を防ぐため，dropout 率を 0.2，勾配爆発を防ぐために Gradient Clipping の係数は 5 とした．学習回数は 50 回，ミニバッチサイズは 128 に設定する．なお実験では，末尾の単語をいくつか除去することでバッチサイズがすべて均等になるように調整する．初期化に関して，重みは平均が 0，分散が $\sqrt{1/n}$ (n:ユニット数)となるガウス分布で初期化[26]し，バイアス項はすべて 0 に初期化する．損失関数には，3 章でも用いた softmax cross entropy を用いた．実験では各タイムステップでの単語が 1 単語ずつ生成されるため，性能評価には Perplexity を用いて評価する．Perplexity は softmax cross entropy を L として，次のように計算できる．

$$\text{Perplexity} = \exp(L) \tag{4.16}$$

Perplexity はモデルの予測性能を評価することができ，より良いモデルは正解となる単語を高い確率で選択することができるため，Perplexity がより小さくなる．Perplexity の最小値は 1.0 である．

先に結論から述べると，4.2 節で構築した 4 種類の RNN モデルにおける WikiText-2 を用いた言語モデリングタスクの Perplexity は LSTM や GRU に対して劣る結果となった．RNN が 1 層のみである場合，4.2 節で構築した 4 種類の RNN モデルのパラメータは LSTM や GRU と比較して少なくなる．したがって，パラメータの増加による性能の向上について調査するため，構築した 4 種類の RNN モデルを多層化した場合の実験を追加して行う．各種設定やパラメータは先述の実験条件と同じであり，RNN だけを多層化する．層数は多層化による性能やパラメータの増加量を考慮して model 1 では 2~3 層までを，model 2 では 2~5 層までを追加実験の対象とする．

4.3.3 IMDB を用いた 2 値分類における性能評価

IMDB[32]は映画におけるレビューデータセットであり，訓練データが 25000，テストデータが 25000 ずつ含まれる．また positive 及び negative の割合は 50%ずつである．本章における実験では更に，元々のテストデータを positive 及び negative の含まれる割合が同じになるように検証データセットとテストデータセットに分割して使用する．したがって，検証データは positive が 6250，negative が 6250 とし，テストデータも同様である．実験で扱う語彙に関しては計算コスト削減のため出現頻度が上位 20000 語以上のものを語彙として扱い，その他は unk で置き換えた．IMDB を用いた 2 値分類タスクに使用する RNN を用いたニューラルネットワークモデルは，入力層，RNN 層，出力層を持ち，入力サイズは語彙サイズと同じであり，出力サイズは positive もしくは negative の 2 値となる．学習回数は 30 回とし，通常の BPTT を用いて学習させる．dropout 率は 0.5，ミニバッチサイズは 256，Gradient Clipping の係数は 25 に設定する．その他実行における各種設定は WikiText-2 を用いた言語モデリングタスクと同様である．IMDB を用いた 2 値分類タスクでは通常の BPTT を用いて学習させるため，末尾の単語を除去してバッチサイズを揃えることはせず，ミニバッチサイズが文長により変化する．

IMDB を用いた 2 値分類では，判定結果となる最終最終タイムステップでの出力のみが分類に用いられるため，性能評価は Accuracy を用いて評価する．Accuracy は，以下に示すように計算できる．

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.17}$$

ここで，TP, TN, FP, FN はそれぞれ，true positive(真陽性)，true negative(真陰性)，false positive(偽陽性)，false negative(偽陰性)を表している．Accuracy は正確さを表す指標であり，IMDB は positive および negative の割合が同じである均衡のとれたデータセットであるため Accuracy のみを性能評価に用いる．

4.3.4　プログラムのフローチャート

図 4.4 に 2 つのタスクにおけるプログラムの大まかなフローチャート図を示す.

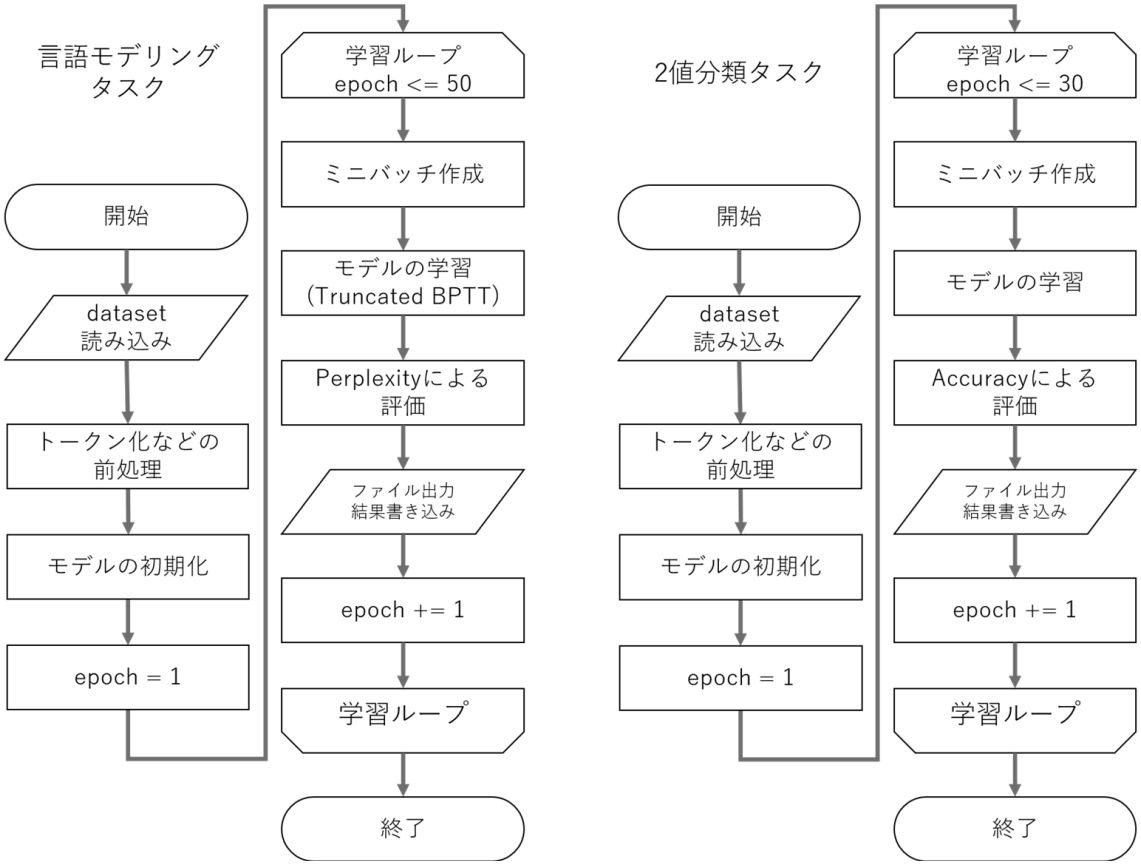

図 4.4: 2 つのタスクにおけるプログラムの大まかなフローチャート図

4.4 実験の結果

WikiText-2 を用いた言語モデリングタスクの結果を表 4.1，IMDB を用いた 2 値分類タスクの結果を表 4.2 に示す．それぞれ，最も検証データセットに対して最も loss が低くなった時点でのテストデータに対する結果を示している．なお，表中では小数点第 3 位を四捨五入し，小数点第 2 位までの値を示している．また，Accuracy はパーセント(%)表記で示している．

表 4.1 から WikiText-2 を用いた言語モデリングタスクでは，Perplexity は値が小さいほどモデルの性能が良いため，model 1，model 2 ともに LSTM や GRU よりも性能が劣ってしまったことが見て取れる．model 1 と model 2 を多層化した場合の性能は 64units の場合を除き，多層化しない場合と比べて Perplexity の値が小さくなり性能が良くなっている．しかし，多層化した場合であっても LSTM や GRU を用いたモデルよりも性能が低く，パラメータ量から考えても優位性は低い．また多層化による性能の向上について，model 1 では 3 層まで，model 2 では 5 層まで多層化した場合には，逆に Perplexity の値が大きくなり，

表 4.1: WikiText-2 を用いた言語モデリングタスクの結果

model	units	Test Perplexity				
		layers				
		1	2	3	4	5
LSTM	64	147.85	-	-	-	-
	128	135.10	-	-	-	-
	256	129.93	-	-	-	-
GRU	64	140.68	-	-	-	-
	128	128.12	-	-	-	-
	256	128.14	-	-	-	-
Simple RNN	64	165.55	-	-	-	-
	128	153.35	-	-	-	-
	256	158.39	-	-	-	-
model 1 (pre-)	64	165.09	175.17	187.14	-	-
	128	157.19	156.57	163.09	-	-
	256	166.18	165.68	164.32	-	-
model 1 (post-)	64	155.64	164.35	173.43	-	-
	128	157.88	156.70	160.14	-	-
	256	173.75	169.74	173.52	-	-
model 2 (pre-)	64	193.17	183.73	190.41	201.15	214.59
	128	185.29	163.05	164.22	170.61	177.61
	256	206.79	170.85	163.38	164.69	171.00
model 2 (post-)	64	173.48	175.65	184.29	193.46	208.20
	128	173.84	163.46	167.76	168.81	171.06
	256	198.41	176.79	166.80	158.61	160.55

表 4.2: IMDB を用いた 2 値分類タスクの結果

model	Test Accuracy (%)		
	units		
	64	128	256
LSTM	76.61	86.94	82.04
GRU	84.53	87.39	86.30
Simple RNN	73.18	72.39	72.70
model 1 (pre-)	72.62	76.91	78.50
model 1 (post-)	68.77	70.39	71.20
model 2 (pre-)	84.68	84.28	83.43
model 2 (post-)	85.35	85.65	84.50

性能が悪化したことが確認できる．構築した model 1 と model 2 における比較では，1 層の場合では model 1 の方が model 2 よりも Perplexity が良い．しかし，多層化した場合の

model 1 では性能が悪化しているのに対し，model 2 では多層化によって性能が向上しており，1 層，128 units の model 1 のスコアと，4 層，256 units の model 2 における性能差は非常に小さい．pre-activation モデルと post-activation モデルでは，post-activation モデルの方が，スコアが良く性能が高くなりやすい傾向が見られる．

一方で IMDB を用いた 2 値分類タスクでは, 表 4.2 から model 2 において LSTM や GRU に匹敵する Accuracy となっていることがわかる．同じユニット数である場合，model 2 のパラメータ量は LSTM の約 1/8，GRU の約 1/6 であることから，パラメータ量が少なく非常に有利である．

model 1 と model 2 との比較では，model 2 の方が，model 1 よりも明らかに Accuracy が良い．また pre-activation モデルと post-activation モデルの比較では，model 1 においては pre-activation モデルの方が post-activation モデルよりスコアが良い．しかし model 2 においては post-activation モデルの方が pre-activation モデルよりスコアが良く, WikiText-2 を用いた言語モデリングタスクと同様の傾向になっていることが確認できる．

4.5 実験結果に対する解析及び考察

構築した sech 関数を用いたゲート構造に関して，隠れ状態の 1 つの要素の x とするとゲートを通過した後の要素は $x \cdot \mathrm{sech}(ax)$ となる．これを微分した場合は以下のようになる．

$$\frac{d}{dx}(x \cdot \mathrm{sech}(ax)) = \mathrm{sech}(ax) \cdot (1 - ax \cdot \tanh(ax)) \tag{4.18}$$

この微分値が各時刻におけるゲートの勾配計算に重要となるため，実験においてユニット数を 128 とした場合に最も性能が高くなった時点での $x \cdot \mathrm{sech}(ax)$ のグラフと微分値のグラフをそれぞれ図 4.5 と 4.6 に示す．グラフ中では，スカラー値 a を見やすくするため大文字の A に置き換えている．

sech 関数を用いたゲート構造においては，a の値が大きいほど勾配が小さくなりやすく，a の値が小さいほど勾配は大きく保たれる．図 4.5，4.6 から model 2 においては 2 つの実験タスクによる性質の違いがパラメータ a にも表れている．まず WkiText-2 を用いた言語モデリングタスクでは, 文章における長期の依存関係だけではなく, 単語間の短い依存関係もとらえる必要がある. 情報を取捨選択し, 適切に勾配を消失させるため a が比較的大きくなるように学習している．一方 IMDB を用いた 2 値分類タスクでは，長期の依存関係をとらえるため, a が非常に小さく, 勾配をより離れた時刻まで伝えられるように学習している．model 1 においては 2 つの実験タスクにおけるパラメータ a がいずれのタスクの場合であっても比較的大きくなるよう学習している．このことは, 隠れ状態に対して行列重みパラメータを有することから，過学習を起こさないように勾配を小さくするような学習の仕方をしたものと考えられる．表 4.2 の結果から IMDB タスクでは隠れ状態への行列重みパラメータは不要であったと考えられ，model 2 の方が model 1 より性能が高い．

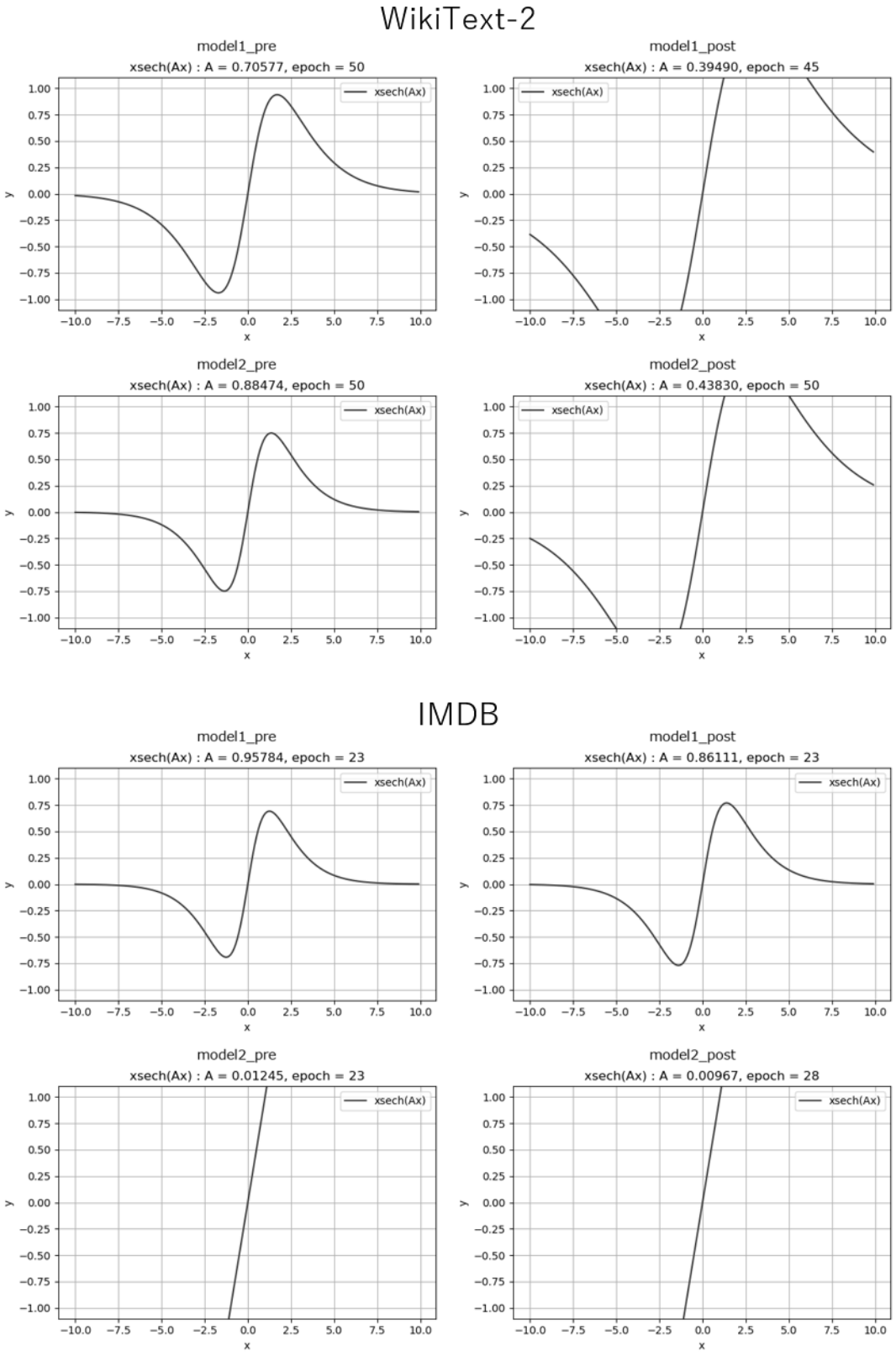

図 4.5: 各モデルにおける $x \cdot \mathrm{sech}(Ax)$

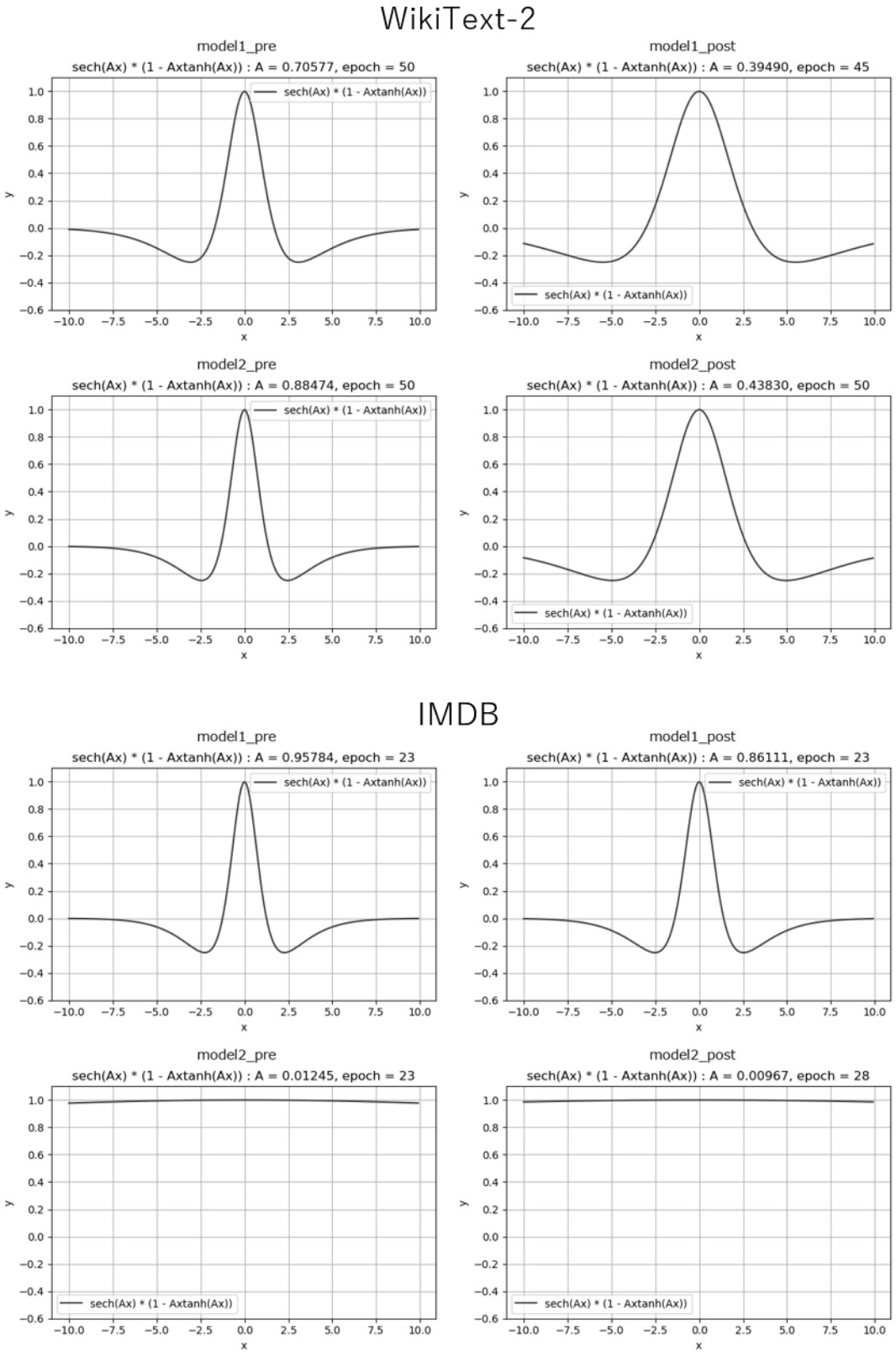

図 4.6: 各モデルにおける$(x \cdot \mathrm{sech}(Ax))'$

pre-activation モデルと post-activation モデルに関しては，図 4.5 と図 4.6 から post-activation モデルの方が比較的 a が小さくなる傾向が見られる．pre-activation モデルでは活性化関数を先に適用するため，RNN モデルの出力 o_t は比較的大きな値となる可能性がある．したがって a の値が大きくなったのは，ゲートを通過した $x \cdot \mathrm{sech}(ax)$ の値が小さくなるように学習することで，出力 o_t が過度に大きな値となり過学習を起こすことを避けるためと推測される．しかし，a が大きい場合には $(x \cdot \mathrm{sech}(Ax))'$ の値も小さくなってしまう．ゆえに時系列方向への勾配消失も発生しやすく，性能面で悪影響を及ぼしている可能性が考えられる．post-activation モデルでは，活性化関数を再帰構造の外部で適用するため，活性化関数に tanh を用いた場合の下層への出力は，$-1 < o_t < 1$ に正規化される．そのため，再帰構造内部における隠れ状態の値が比較的大きな値をとっても下層では問題とはならない．よって a が小さくなるように学習し，$(x \cdot \mathrm{sech}(Ax))'$ の値を大きく保つことで勾配をより遠い時刻まで伝えることができ，pre-activation モデルよりも性能が高くなったと推測できる．

実験では従来の RNN との比較を行いやすくするために tanh を活性化関数として用いたが，再帰構造内部での隠れ状態が大きくなった場合には tanh を通過する際に勾配が小さくなってしまい，勾配消失が起きてしまう．このことが多層化した場合の性能に悪影響を及ぼしている可能性が考えられる．例えば，post-activation の model 2 において，隠れ状態の 1 つの要素 x へ入力される上層からの出力が 0 であった場合を考える．この要素 x に対応する出力 o は，以下のように計算できる．

$$o = \tanh(x \cdot \mathrm{sech}(ax)) \tag{4.19}$$

この o を微分すると，式(4.3)を用いて以下のように計算できる．

$$\begin{aligned} \frac{do}{dx} &= (1 - \tanh^2(x \cdot \mathrm{sech}(ax))) \cdot (x \cdot \mathrm{sech}(ax))' \\ &= (1 - \tanh^2(x \cdot \mathrm{sech}(ax))) \cdot \mathrm{sech}(ax) \cdot (1 - ax \cdot \tanh(ax)) \end{aligned} \tag{4.20}$$

post-activation の model 2 において，ユニット数が 128 units，RNN の層数が 5 層の場合について最も性能が高くなった時点での各層の式(4.19)と式(4.20)のグラフをそれぞれ図 4.7 と図 4.8 に示す．ここでも図中では，スカラー a を大文字の A に置き換えて示している．図 4.7 から，出力層に近い 5 層目において要素 x の値が大きくなれば，出力 o も小さくなることが見て取れる．また図 4.8 から x が大きな値になれば，勾配がより小さくなることも確認できる．実際の入力においては，様々な値が入力されるため，常に 0 が入力されるわけではないが，多層化した場合には比較的勾配消失を起こしやすい状態であることが確認できる．

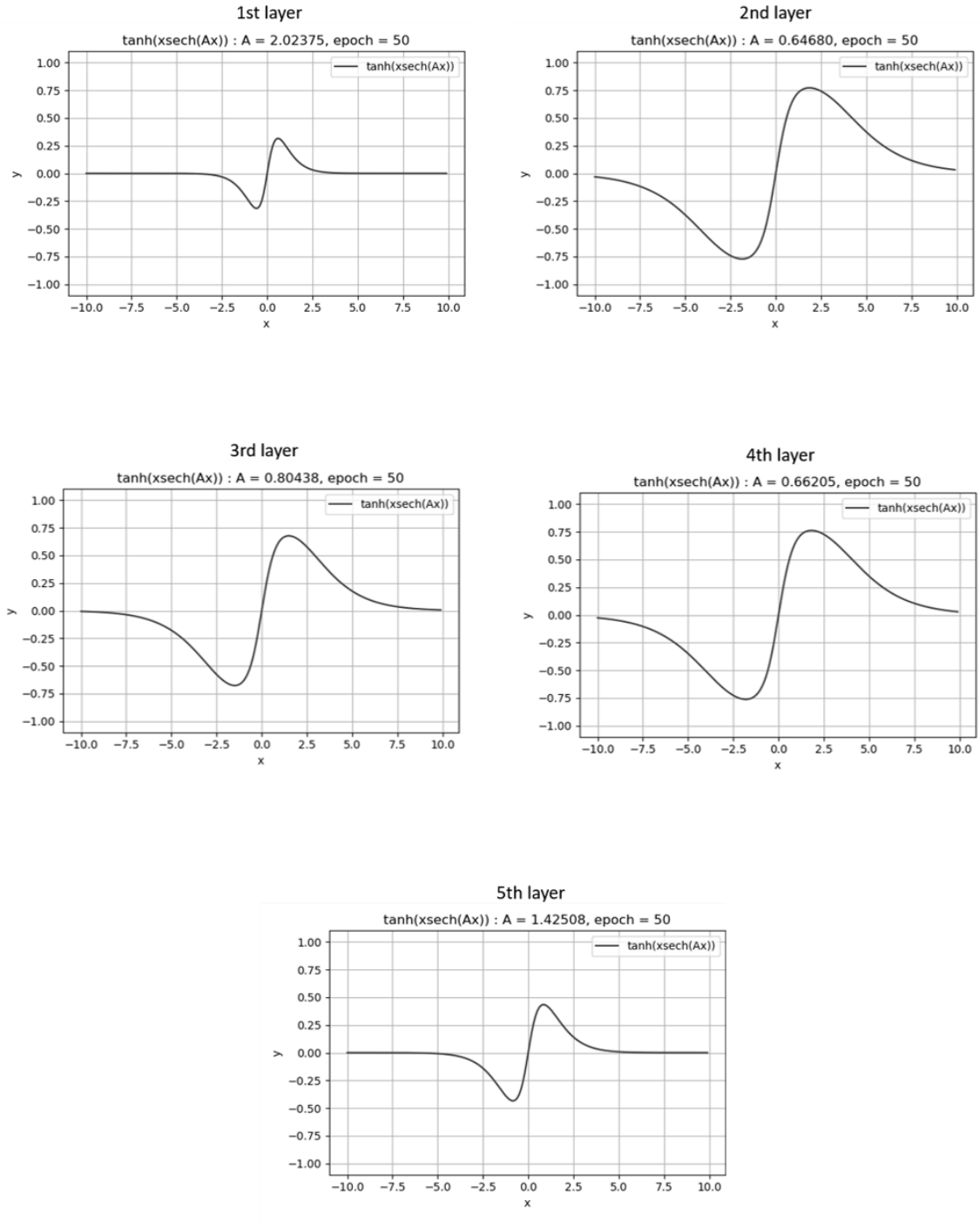

図 4.7: model 2 post-activation における式(4.19)のグラフ

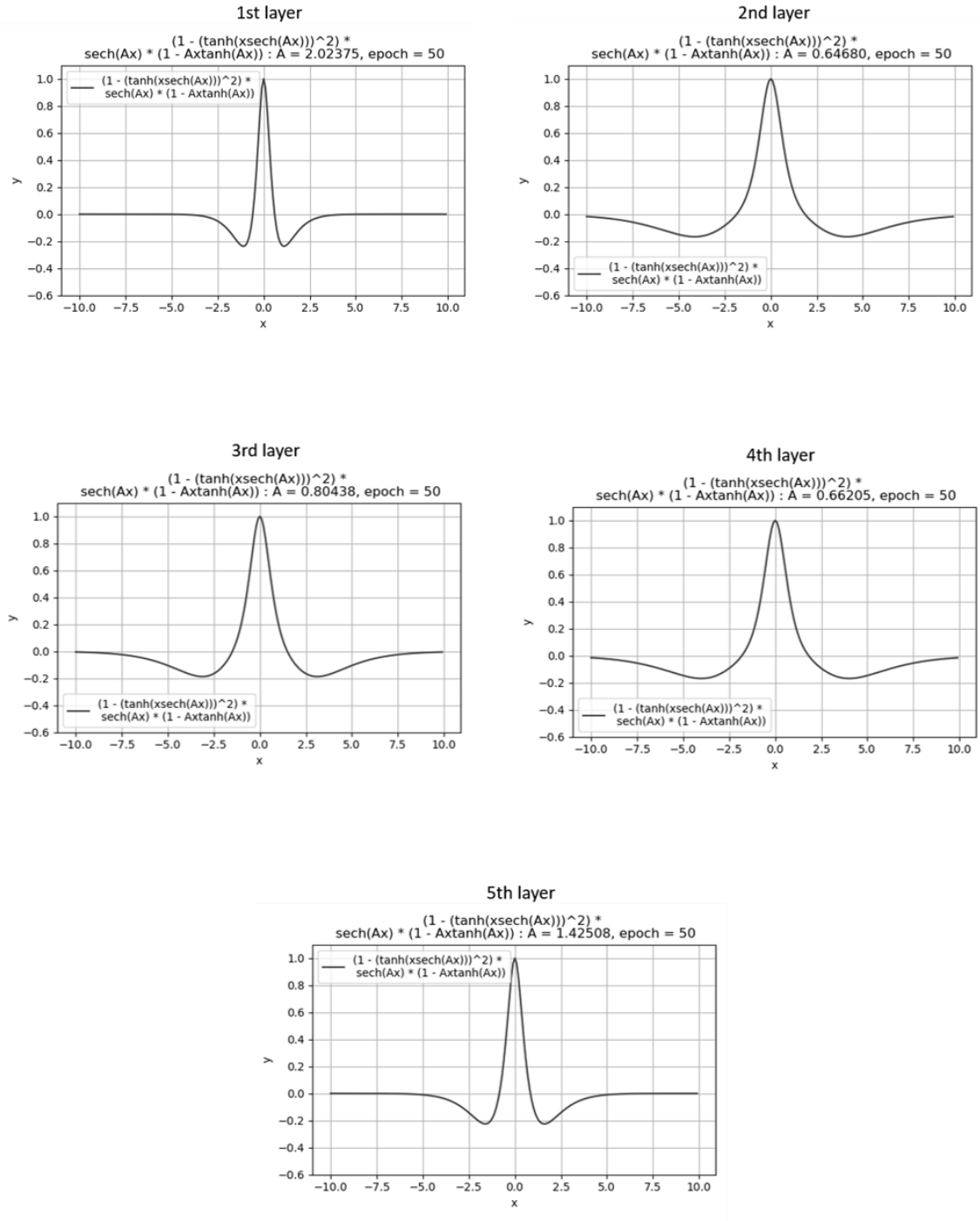

図 4.8: model 2 post-activation における式(4.20)のグラフ

4.6 おわりに

本章では RNN のパラメータを更に削減するため，行列重みパラメータをもつ従来のゲート構造に変わり，sech 関数とスカラー値 a を用いたゲート構造を用いてパラメータ量が少ない RNN を構築し，それを元に WikiText-2 を用いた言語モデリングタスクと IMDB を用いた 2 値分類タスクという性質の異なる 2 つのタスクについて性能評価及び解析を行った．結果として，最終時刻の出力が重要となる 2 値分類タスクにおいては従来のゲートをもつ RNN よりも 1/6 以下と非常に少ないパラメータでも同程度の性能が達成できることがわかった．これにより，ゲート構造を改善することによって更なるパラメータの削減を行うことが可能であるということが示された．

一方で，各時刻の出力を必要とする言語モデリングタスクでは LSTM や GRU に劣る性能となり，多層化した場合でも同様であった．これは，4.5 節で述べた通り出力に用いる活性化関数との間で勾配が消失してしまったことが原因ではないかと考えられるが，詳細については今後さらに検討する必要がある．

本章では自然言語処理に関するタスクによって性能を評価したが，時系列データ処理や画像処理などにおけるタスクにも sech 関数とスカラー値 a を用いたゲート構造が有効に機能するのかを確認する必要がある．次章では，自然言語処理以外の時系列データ処理や画像処理を含む応用的な医療情報処理として，2 次元リカレンスプロットを用いたパーキンソン病の検出タスクを用いて更なる実験を行い，モデルの性能を検討及び調査する．

第5章 リカレンスプロットを用いたパーキンソン病検出

5.1 はじめに

第 4 章より，sech 関数とスカラー値 a を用いたゲート構造を持つパラメータ量の少ない RNN において，自然言語処理における 2 値分類タスクでは LSTM 及び GRU よりも 1/6 以下と非常に少ないパラメータ量で性能が発揮できることが示された．しかし，自然言語処理以外の時系列データ処理や画像処理などにおけるタスクにおいても性能が発揮できるかは未知数であり，多層化における性能を改善するためにも更なる実験を行う必要がある．

本章では時系列データ処理や画像処理を含む応用的な医療情報処理としてパーキンソン病(Parkinson's Disease: PD)の検出タスクを新たに設定し，更に検証を進める．具体的には，PD 患者のもつ運動障害性構音障害に着目し，単母音の音声を特定区間に区切って分割したものを 2 次元リカレンスプロット画像に変換した上で RNN を用いた PD 検出モデルにより検出する．その検出精度を従来型のゲートを持つ LSTM 及び GRU と比較することで性能を検討し，評価を行う[33]．

本章における主な目的は，時系列データ処理や画像処理を含む PD 検出タスクを応用課題として用い，4 章で構築した sech 関数とスカラー値 a を用いたゲート構造を持つ RNN モデルを評価することである．

実験では，音声データの前処理としてリカレンスプロット（Recurrence Plot: RP）への画像化を用いて PD 検出を行う．音声データのような時系列データを解析する時には，時間周波数解析として短時間フーリエ変換（Short-Time Fourier Transform: STFT）を用いるのが一般的である．STFT は窓関数を用いて時間信号を同じ長さの短いセグメントに分割し，各セグメントに対して個別にフーリエ変換を行うことで計算される．しかしフーリエ変換は非定常信号の分解能が低いという欠点がある．一方，RP はカオスな時系列を表現し周期性を視覚化する方法として Eckmann らによって導入された 2 次元画像である[34]．リカレンスプロット画像の生成では，時系列データ間の任意の 2 点間で距離計算を行い，任意のしきい値より小さい距離にある点をプロットする方法や，任意のパーセンタイル値よりも小さい距離の点をプロットする方法などがある．フーリエ変換は独立な基底関数を適切に選択できない系の記述には適さないが，RP は対象が非線形および非定常でも扱うことができる[35]．PD 患者の音声は運動障害性構音障害により，声の震えやかすれ，声の大きさにばらつきが生じるなど非定常性を持っているため，非線形かつ非定常信号でも処理可能な RP を用いることでより正確に特徴をとらえられる可能性がある．また RP は，従来の時間周波数解析では捉えられなかった動物の音声信号の微弱な変調なども検出することができ，非常に強力なツールである[36]．本章における RP の生成では，絶対距離を計算し，任意のパーセンタイル値によって 2 次元 RP 画像を生成する．

近年の音声を用いた PD 検出における関連研究では，機械学習を用いたものが一般的で

ある．関連研究における機械学習におけるアプローチには，x-vector と呼ばれる，ディープニューラルネットワークから抽出された埋め込みベクトルを用いて Latent Dirichlet Allocation (LDA)や Polylingual Latent Dirichlet Allocation (PLDA)といった分類モデルのコサイン距離などによって PD を分類する研究[37]や, 4 つの機械学習法(k-nearest neighbor, multi-layer perceptron，optimum-path forest，support vector machine）と 18 の特徴抽出法を用いて PD 検出を行う研究[38]がある．一方，深層学習におけるアプローチでは，複数のニューラルネットワーク及び主成分分析と自己組織化マップによって選択された 26 の音声特徴を用いて PD 検出を行った研究がある[39]．またディープニューラルネットワークと 16 の生物医学的な音声特徴を使用し PD の重症度を検出しようとする研究もある[40]．これらの研究では，音声特徴量として複数の特徴を必要とし，その抽出にかかる計算コストも高くなる傾向にある．一方，本章における RP を用いた PD 検出における音声データの前処理では，単母音の音声データから振幅の絶対距離とパーセンタイル値のみを計算して RP 画像に変換するため，従来の PD 検出手法と比べて前処理にかかる計算コストは低く，実装も容易である．RP を用いた研究では，画像認識に用いられる Convolutional Neural Network (CNN)[5, 6]を用いて RP 画像を分類する研究なども増えており[41, 42]，CNN によって PD 患者の手書きダイナミクスデータの RP 画像を識別する研究[43]などもある．

したがって本章では，PD 患者の単母音の音声データから生成された RP 画像を用いて PD 検出を行うため，CNN と RNN を組み合わせた複合ニューラルネットワークモデルを新たに構築し，4 章で使用した sech 関数とスカラー値 a を用いたゲート構造を持つ RNN の性能評価に用いる．筆者の知る限り，PD 患者の単母音の音声データから生成される RP 画像を用いてニューラルネットワークモデルにより PD を検出する研究は，この研究の他に類を見ない．

次節では，音声データの前処理の手順及び検出モデルの概要について述べる．

5.2 音声データの前処理の手順及び PD 検出モデルの概要

5.2.1 PD 患者の音声データ

本章で扱う PD 患者の音声データについて説明する．音声データセットは被験者の音声を約 4 秒から 10 秒程度録音したものとなっており，対象となる単母音は /a/ である．このデータセットには，22 人の健常者(Healthy Person: HP)と 30 人の PD 患者のデータが含まれ，1 人につき 3 回分のデータが記録されている．よってデータの総数は 156 ((22＋30)×3)となる．性別の内訳は，女性 27 名，男性 25 名である．また PD 患者は Hoehn and Yahr（ホーン・ヤール）の重症度分類によってステージ 1 から 5 までに分類されており，ステージの内訳はステージ 3-5 の PD 患者が 13 名，ステージ 3 よりも低い PD 患者が 17 名である．

5.2.2　音声データの前処理と RP 画像の作成手順

一部の音声データはそれぞれ異なった環境で録音されたものであるため，すべての音声データは録音環境やその方法に関わらず，モノラルかつサンプリング周波数が 16000 [Hz] となるように変換される．なお，すべての音声データは量子化ビット数が 16 ビットで量子化されたものである．

RP 画像を作成する前の音声データにおける前処理の手順を以下に示す．

1. 声の大きさによる影響を避けるため，すべての音声データそれぞれについて，最大振幅が量子化ビット数: 16 ビットで表現可能な最大振幅となるように正規化を行う．

2. 上記の正規化を行った音源に対し，-40dB 以下の区間を無音区間として音声波形の両端からのみ削除する．

3. 無音区間を削除した後の音声波形の両端 0.1 秒間のデータを更に削除する．これは無音区間削除後の音声において，声の出始めと終わりが非常に不安定になりやすいことを考慮して行われる．

上記手順により前処理された音声データに対し，以下の手順で RP 画像を生成する．

1. 前処理された音声データを 0.01s 間のセクションに分割する．

2. RP 画像を作成する直前に，0.01s 間の音声データそれぞれについて振幅が[-1, 1]になるように正規化を行う．

3. 0.01s 間の音声データにおいて，それぞれの点における振幅の絶対距離を計算する．この絶対距離において 35 パーセンタイルに相当する距離を選択後，35 パーセンタイルの値よりも小さい距離の点を 1 (黒)でプロットし，その他の点は 0 (白)とする．ここで 35 パーセンタイルを使用する理由は，実験において 35 パーセンタイルを使用した RP 画像で最も高い精度が得られたためである．

4. 生成された 160×160 の画像サイズを持つ白黒 RP 画像をバイリニア補間により，20×20 の画像サイズに圧縮する．このことで検出精度が悪化してしまう可能性があるが，実験に使用する GPU の VRAM メモリ(8GB)の容量を考慮して画像の圧縮を行う．

5. すべての分割された音声データが RP 画像になるまで上記の 2-4 を繰り返す．ただし，音声データの末尾が 0.01s に満たない場合は，切り捨てられる．

RP 画像の作成において，時系列を $\{x_i\}_{i=1}^{N} = \{x_1, x_2, \cdots, x_N\}$ とし，この時系列データの絶対距離の 35 パーセンタイルに相当する距離を $d_{35\%}$ とおくと，RP 画像を表す行列 R は次のようになる．

$$R(i,j) = \begin{cases} 1 & \text{if } |x_i - x_j| < d_{35\%} \\ 0 & \text{otherwise} \end{cases} \tag{5.1}$$

ここで，$i,j \in \{1,2,\cdots,N\}$ は時系列 $\{x_i\}_{i=1}^{N}$ 及び行列 R の要素番号を表す．また，RP 画像において $R(i,j)=1$ は黒，$R(i,j)=0$ は白となる．図 5.1 に音声データと生成された RP 画像の例を示す．

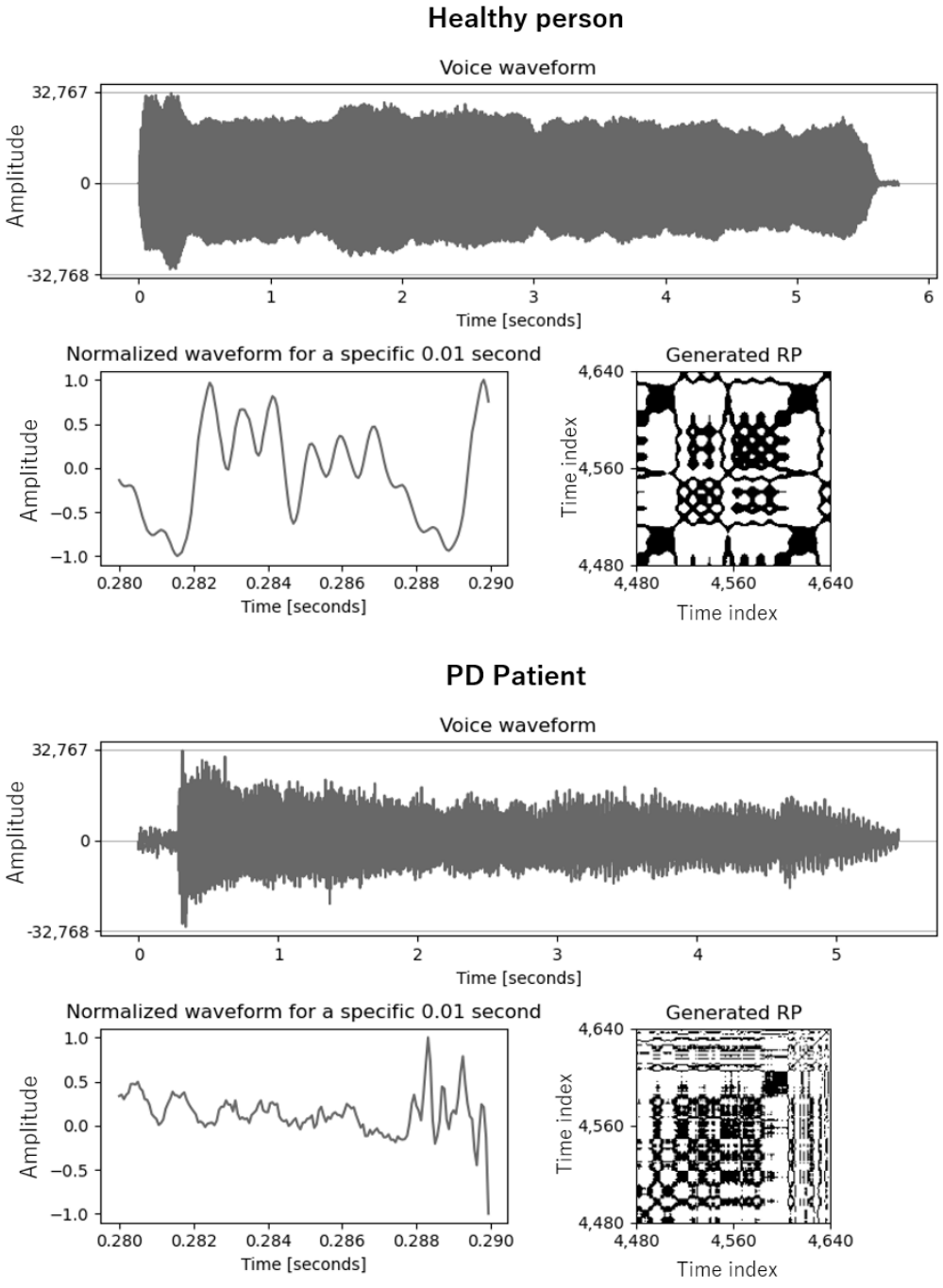

図 5.1 音声データと生成された RP 画像の例（上: 健常者，下: PD 患者）

図 5.1 における RP 画像は，0.28 秒から 0.29 秒までの区間において生成された画像を時計回りに 90 度回転して示している．この図 5.1 の RP 画像の例では，健常者と PD 患者の RP 画像にはっきりとした違いが表れている．

5.2.3　PD 検出モデルの概要

PD 検出モデルは，RP 画像を処理する CNN 部と，CNN 部が出力した特徴ベクトルを時系列データとして処理する RNN 部及び，最終的な検出結果を出力する出力層で構成される．図 5.2 に PD 検出モデルの構成についての全体図を示す．また，表 5.1 に CNN の各層における設定パラメータを示す. CNN に入力される RP 画像サイズは 20×20 であり, CNN からの出力サイズは 64×5×5(=1600)である．したがって RNN には，1600 次元の特徴ベクトルが入力される．RNN は後述の実験において，ユニット数が 64，128，256 のいずれかに，層数が 1 層もしくは 2 層に設定され，それに応じて出力層のユニット数も RNN と同じユニット数が設定される. 出力層からの出力結果は 2 値(HP あるいは PD)となる. PD 検出モデルは，すべての RP 画像が入力された後に最終的な結果を出力する．

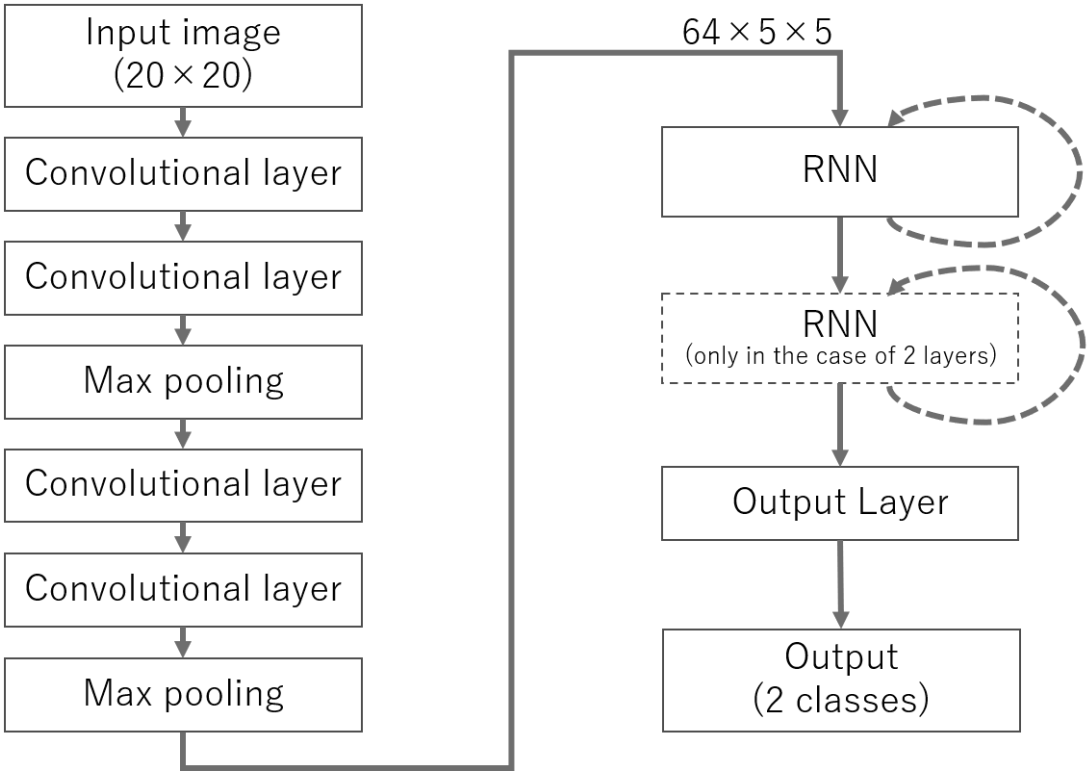

図 5.2: PD 検出モデルの全体図

表 5.1: CNN の各層における設定パラメータ

layer	In-channels	Out-channels	Kernel size	Stride	Padding (for each dimension)
Convolutional layer	1	32	3×3	1×1	1
Convolutional layer	32	32	3×3	1×1	1
Max pooling	-	-	3×3	2×2	1
Convolutional layer	32	64	3×3	1×1	1
Convolutional layer	64	64	3×3	1×1	1
Max pooling		-	3×3	2×2	1

CNN の各パラメータは表 5.1 に示すとおりであるが，convolutional layer の後に適用される活性化関数には relu を用いた．また，学習時の過学習を防ぐ dropout は CNN 部の max pooling の直後と出力層の直前に挿入される．ただし RNN が 2 層の場合には，2 層目の RNN の直前にも dropout が挿入される．

5.3 PD 検出における実験の概要と性能評価

5.3.1 実験の概要

本章における実験では 5.2 節で述べた PD 検出モデルと RP 画像を用いた計算機実験により各 RNN の性能比較を行うともに，PD 検出モデルの性能を検討する．実験は Python 3.8 と機械学習ライブラリである Pytorch 1.7.1，並びに Cuda 11 を用いて単精度浮動小数点演算により実行される．音声の前処理に使用するライブラリには Pydub[44]を用いた．なお，Pydub の一部機能は FFmpeg[45]に依存している．また実験における再現性を確保するため，環境変数として CUBLAS_WORKSPACE_CONFIG＝:16:8 が設定され，乱数シードはすべて 10 に設定される．

5.3.2 性能評価実験における設定

5.2.1 項で述べた通り，データセットにおけるデータの総数は，156 (HP: 22×3，PD: 30×3)である．このデータセットを，訓練データセット: 28×3 (HP: 10×3, PD: 18×3, 約 54%)，検証データセット: 12×3 (HP: 6×3, PD: 6×3, 約 23%)，テストデータセット: 12×3(HP: 6×3, PD: 6×3, 約 23%)に分割して PD 検出モデルを学習させる．ここで，同じ被験者の音声は同じデータセットに属するように分割する．これは同じ被験者の音声が別のデータセットに含まれることによって，評価時の精度が極端に高くなってしまうなどの問題を避けるために行われる．

訓練データセットでは，健常者と PD 患者の間でデータの総数に不均衡が生じないよう，健常者の音声を 2 秒だけシフトした音声をオーバーサンプリングデータとして加える．またデータの分け方による精度のばらつきの影響を抑えるため，実験は各 RNN に対して 5 回ずつ行われ，5 回の平均精度に基づいて性能評価を行う．各実行の前にデータセットをシャ

ッフルして分割するが，乱数シードを固定しているため，異なる RNN を用いた場合でもシャッフル後に分割されたデータセットの結果は同じ結果となる．

PD 検出モデルに使用される RNN について，本章の実験には，sech 関数とスカラー値 a をゲートに持つ RNN の内，第 4 章の自然言語処理における 2 値分類タスクで最も精度が高かった model 2 の post-activation モデルを用いて実験を行う．以後，本章では model 2 の post-activation モデルを表記簡略化のため”RNN-SH”と呼称する．また RNN-SH との比較対象として，従来型のゲートを持つ RNN である LSTM 及び GRU を用いて同様に実験を行う．

RNN 及び出力層のユニット数は，64，128，256 の 3 通りに設定される．また多層化した場合の性能を確認するため，1 層または 2 層の RNN を用い，RNN-SH では出力に用いる活性化関数として tanh または relu を用いて実験を行う．学習に用いる確率的勾配降下法には RAdam 法[46]を用いた．RAdam のパラメータは，過学習を防ぐための荷重減衰(weight decay)を 1e-2 とし，その他は推奨パラメータである，$\alpha = 0.001,\ \beta_1 = 0.9,\ \beta_2 = 0.999$ とした．その他，dropout 率は 0.5，学習回数は 50 回，バッチサイズは 27 に設定した．各層の重みパラメータの初期化に関しては，重みは平均が 0，分散が $\sqrt{1/n}$ (n:ユニット数)となるガウス分布で初期化[26]して，バイアス項はすべて 0 に初期化する．CNN のカーネルにおける重みの初期化は，$v = \sqrt{3/n}$ となる v を用いて，一様分布 $[-v, v]$ で初期化[26]し，バイアス項はすべて 0 とした．RNN-SH のゲートにおけるスカラーパラメータ a は 1.0 に初期化される．損失関数には，第 3 章，第 4 章でも用いた softmax cross entropy を使用する．

性能評価には，Accuracy，F-score，Matthews Correlation Coefficient（MCC）の 3 つの指標を用いる．ここで，各指標における計算式は以下のようになる．

Accuracy:

$$Accuracy = \frac{TP\ +\ TN}{TP\ +\ TN\ +\ FP\ +\ FN} \tag{5.2}$$

F-score:

$$F - score = \frac{2TP}{2TP\ +\ FP\ +\ FN} \tag{5.3}$$

Matthews Correlation Coefficient（MCC）:

$$MCC = \frac{(TP \times TN) - (FP \times FN)}{\sqrt{(TP\ +\ FP) \times (TP\ +\ FN) \times (TN\ +\ FP) \times (TN\ +\ FN)}} \tag{5.4}$$

ここで $TP,\ TN,\ FP,\ FN$ は，第 4 章に示したものと同様に，それぞれ true positive(真陽性)，true negative(真陰性)，false positive(偽陽性)，false negative(偽陰性)の数を表している．Accuracy と F-score は 0 から 1 の実数値を取り，値が 1 に近いほど性能が良いと判断される．また MCC は，－1 から＋1 の実数値を取り，＋1 が完全な予測，0 がランダムな予測

と同等，－1 が完全な逆予測となっていることを示す. 不均衡なデータにおいては, F-score と MCC の方が Accuracy よりも正確に評価をすることが可能となる．F-score では，2 値分類における正と負のクラスにどのクラスを割り当てるかによってスコアが変動してしまうが, MCC ではクラスの割り当てに依存しない評価が可能である. なお本章における F-score の性能評価時には, 正負のクラスを入れ替えた場合を計算して 2 つ値の平均を取る, macro-average の値を示す.

5.4 PD 検出における実験の結果

各 RNN モデルにおける PD 検出の結果を表 5.2 に示す．表 5.2 ではそれぞれの実行において，検証データセットに対して loss が最も低くなった時点での，検証データセットとテストデータの 5 回の実行結果の平均と標準偏差を示している．また 5 回の検証データセットとテストデータセットに対するすべての結果の平均と標準偏差も示す. なお, Accuracy と F-score は小数点第 3 位を四捨五入した値をパーセント(%)表記で示しており，MCC は小数点第 4 位を四捨五入しした値を示している. 加えて, 性能比較をわかりやすくするため, 図 5.3 に各 RNN モデルにおける MCC の結果をグラフ化したものを示す．グラフを見やすくするため標準偏差は省略して示している．図 5.3 ではそれぞれ，左が検証データセット，中央がテストデータセット，右が検証データセットとテストデータセットの全体平均を表している.

図 5.4 は，PD 検出モデルのパラメータと RNN のユニット数及び層数の関係を表したグラフである．また図 5.5 には，各モデルの 1 度目の学習においてプログラムが終了するまでに実行にかかった時間のグラフを示す. ここで, 公正な速度比較を行うため, LSTM と GRU では並列計算可能な行列重みパラメータを一括計算した後に分割するように，高速に計算可能な実装を行っている.

表 5.2: 各 RNN モデルにおける PD 検出の結果

			Acc. (%)			F-score (%)			MCC		
layers	model	units	Average (validation set)	Average (test set)	Total Average	Average (validation set)	Average (test set)	Total Average	Average (validation set)	Average (test set)	Total Average
1	LSTM	64	68.33±7.58	66.11±6.89	67.22±7.33	68.11±7.67	65.54±6.95	66.83±7.43	0.371±0.154	0.334±0.141	0.353±0.149
		128	63.33±10.45	58.89±16.61	61.11±14.05	62.98±10.23	58.46±16.48	60.72±13.90	0.278±0.219	0.187±0.340	0.233±0.290
		256	68.89±9.20	70.00±7.54	69.44±8.43	68.40±9.13	69.54±8.04	68.97±8.62	0.395±0.194	0.407±0.145	0.401±0.171
	GRU	64	67.22±7.54	66.11±8.85	66.67±8.24	66.94±7.54	65.33±9.19	66.13±8.44	0.352±0.158	0.337±0.176	0.345±0.167
		128	65.56±9.56	66.67±8.43	66.11±9.03	64.98±9.64	64.92±9.64	64.95±9.64	0.327±0.209	0.363±0.170	0.345±0.191
		256	70.56±10.03	66.67±6.09	68.61±8.52	70.29±10.29	65.84±6.70	68.07±8.97	0.417±0.205	0.347±0.117	0.382±0.170
	RNN-SH (tanh)	64	71.11±7.97	63.33±5.67	67.22±7.93	70.16±8.87	61.75±7.27	65.96±9.13	0.440±0.151	0.282±0.107	0.361±0.153
		128	68.89±7.74	63.33±5.39	66.11±7.22	66.98±10.48	62.86±5.45	64.92±8.60	0.402±0.140	0.274±0.111	0.338±0.142
		256	70.56±7.78	70.00±6.43	70.28±7.14	70.04±8.26	69.32±6.80	69.68±7.58	0.421±0.155	0.417±0.124	0.419±0.140
	RNN-SH (relu)	64	57.22±7.16	67.78±9.72	62.50±10.04	54.42±8.92	65.24±14.33	59.83±13.10	0.164±0.185	0.358±0.195	0.261±0.213
		128	65.56±6.24	63.89±5.83	64.72±6.09	62.10±9.78	60.16±10.88	61.13±10.39	0.370±0.099	0.321±0.078	0.346±0.092
		256	70.00±6.89	63.89±11.25	66.94±9.82	69.56±6.94	63.48±11.76	66.52±10.13	0.413±0.143	0.279±0.229	0.346±0.202
2	LSTM	64	74.44±8.13	66.67±6.80	70.56±8.44	74.29±8.12	65.98±7.43	70.14±8.82	0.496±0.165	0.343±0.130	0.420±0.167
		128	70.00±7.54	68.89±6.89	69.44±7.24	69.57±7.34	67.85±7.40	68.71±7.42	0.418±0.163	0.405±0.136	0.411±0.150
		256	65.00±7.37	69.44±9.94	67.22±9.02	63.60±8.08	68.95±10.30	66.27±9.64	0.322±0.160	0.397±0.197	0.360±0.183
	GRU	64	68.89±6.43	65.56±7.58	67.22±7.22	68.26±6.69	64.99±8.18	66.62±7.65	0.394±0.133	0.316±0.150	0.355±0.147
		128	70.56±9.40	66.11±7.54	68.33±8.80	70.42±9.36	64.90±7.93	67.66±9.10	0.417±0.193	0.350±0.158	0.383±0.180
		256	70.56±9.06	70.00±8.31	70.28±8.70	70.44±9.04	69.01±8.94	69.72±9.02	0.416±0.186	0.425±0.163	0.421±0.175
	RNN-SH (tanh)	64	68.89±8.31	56.11±9.69	62.50±11.06	66.18±10.50	52.25±11.18	59.21±12.89	0.433±0.142	0.163±0.210	0.298±0.225
		128	68.89±5.39	62.78±5.15	65.83±6.09	66.81±7.19	59.49±8.78	63.15±8.82	0.429±0.097	0.297±0.071	0.363±0.108
		256	68.33±8.89	66.67±7.03	67.50±8.06	67.04±10.03	65.12±8.23	66.08±9.23	0.385±0.170	0.358±0.132	0.372±0.153
	RNN-SH (relu)	64	63.33±7.11	66.67±7.66	65.00±7.58	61.86±6.99	65.31±8.69	63.58±8.08	0.298±0.163	0.354±0.146	0.326±0.158
		128	67.78±6.94	65.56±5.44	66.67±6.34	66.61±7.40	63.92±6.08	65.27±6.90	0.380±0.136	0.346±0.120	0.363±0.129
		256	63.33±8.50	61.11±7.66	62.22±8.16	60.44±11.46	57.78±11.65	59.11±11.63	0.288±0.158	0.233±0.157	0.261±0.160

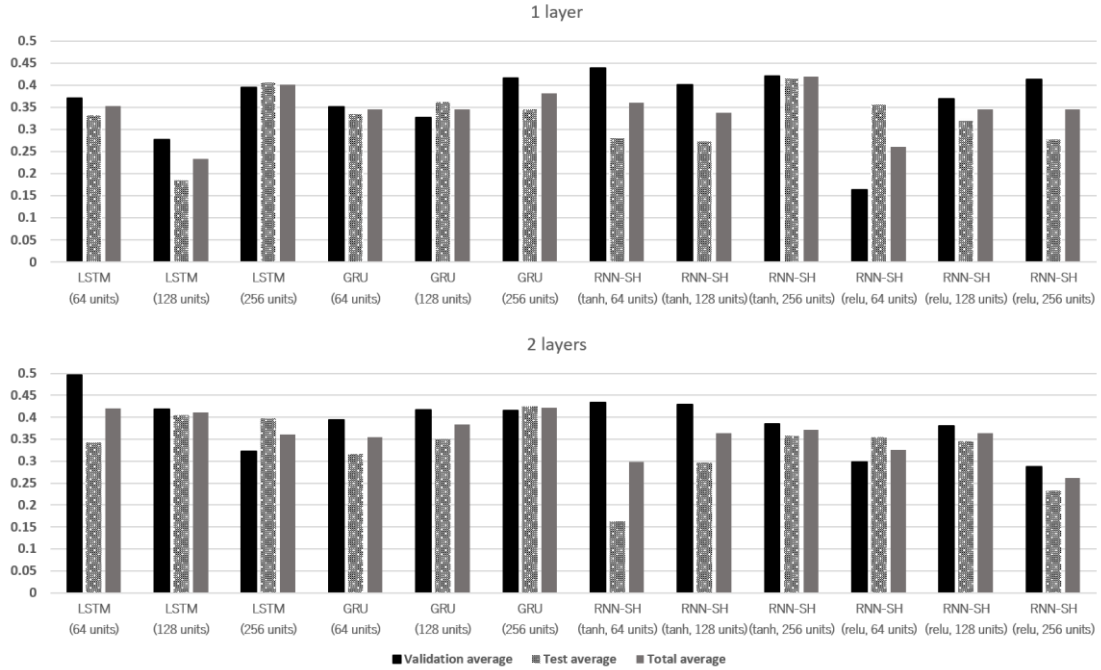

図 5.3: 各 RNN モデルにおける MCC の結果比較

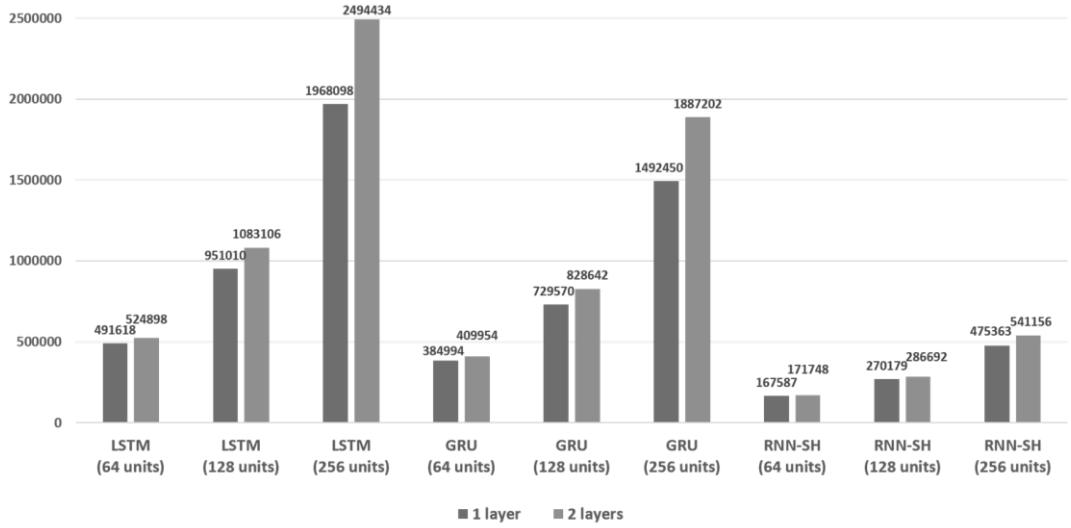

図 5.4: PD 検出モデルのパラメータと RNN のユニット数及び層数の関係

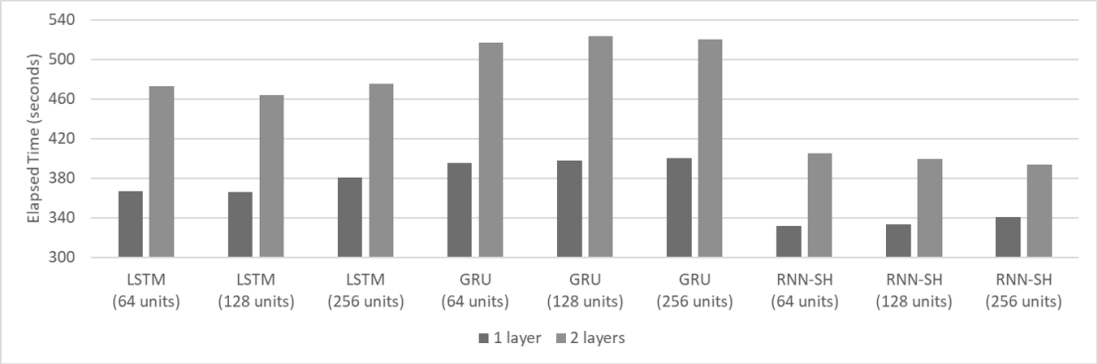

図 5.5: 1 度目の試行でプログラム終了までにかかった時間

表 5.2 の結果から，MCC の全体平均のスコアは上から順に，GRU(2 層, 256 units)，LSTM(2 層, 64 units)，RNN-SH (1 層, 256 units, tanh)の順番にスコアが高くなっている．しかし 5 回の実行において，この上位 3 モデルに対してフリードマン検定を実行したところ，統計的優位差は見つからなかった．2 層かつ 256 units の GRU と 1 層かつ 256 units の RNN-SH の比較において，実行 5 回の平均 ROC 曲線と AUC 値を図 5.6 に示す．

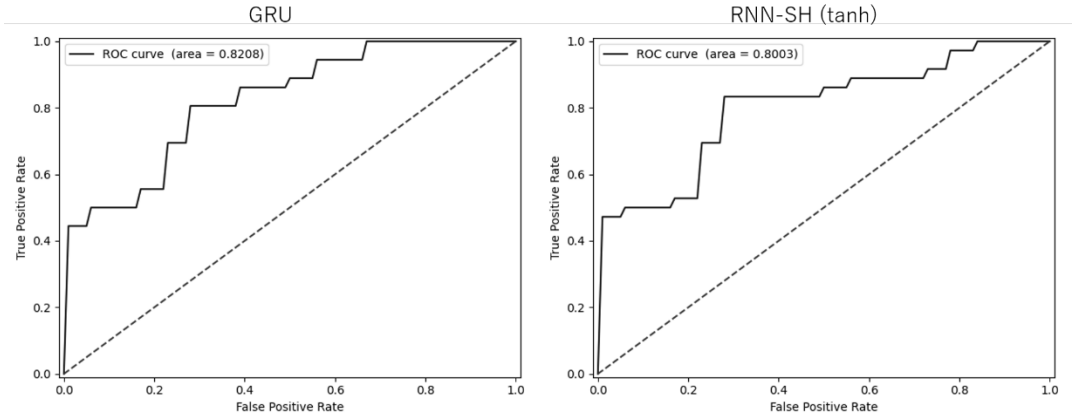

図 5.6: GRU(2 層, 256 units)と RNN-SH(1 層, 256 units, tanh)の ROC 曲線と AUC 値

ここで ROC 曲線における AUC 値が高いほど性能が良いとされるが，図 5.6 では GRU(2 層, 256 units)の平均 AUC 値が RNN-SH (1 層, 256 units, tanh)の平均 AUC 値を上回っていることが見て取れる．このことから，RNN-SH(tanh)を使用した場合の PD 検出能力が GRU を使用した場合の検出能力より少し低くなったことがわかる．しかし図 5.4 から，GRU(2 層, 256 units)のパラメータ量は RNN-SH (1 層, 256 units, tanh)のパラメータ量の 3 倍以上となっており，図 5.5 を見れば，RNN-SH (1 層, 256 units, tanh)は GRU(2 層, 256 units)より約 1.5 倍高速に学習が可能である．

LSTM(2 層, 64 units)と RNN-SH (1 層, 256 units, tanh)との比較では，図 5.5 より RNN-SH (1 層, 256 units, tanh)の方が LSTM(2 層, 64 units)よりもパラメータ量が少なく，表 5.2 より全体平均の MCC は非常に近い値となっている．しかし図 5.3 より，LSTM(2 層, 64 units)では，検証データセットに対しては性能が良いがテストセットに対して性能が低くなり過学習を起こしていることがわかる．一方 RNN-SH (1 層, 256 units, tanh)では，検証データセットとテストデータセットとの間でスコアの差は小さく，過学習を起こすことなく学習できている．ここで，LSTM(2 層, 64 units)と RNN-SH (1 層, 256 units, tanh)の 1 回目の試行における正規化された confusion matrix の例を図 5.7 に示す. 図 5.7 からも LSTM(2 層, 64 units)は検証データセットに対しては性能が良いが，テストデータセットに対して性能が低くなっており，同様の傾向が見て取れる．速度比較では，RNN-SH (1 層, 256 units, tanh)は，LSTM(2 層, 64 units)よりも約 1.4 倍高速に学習が可能である．

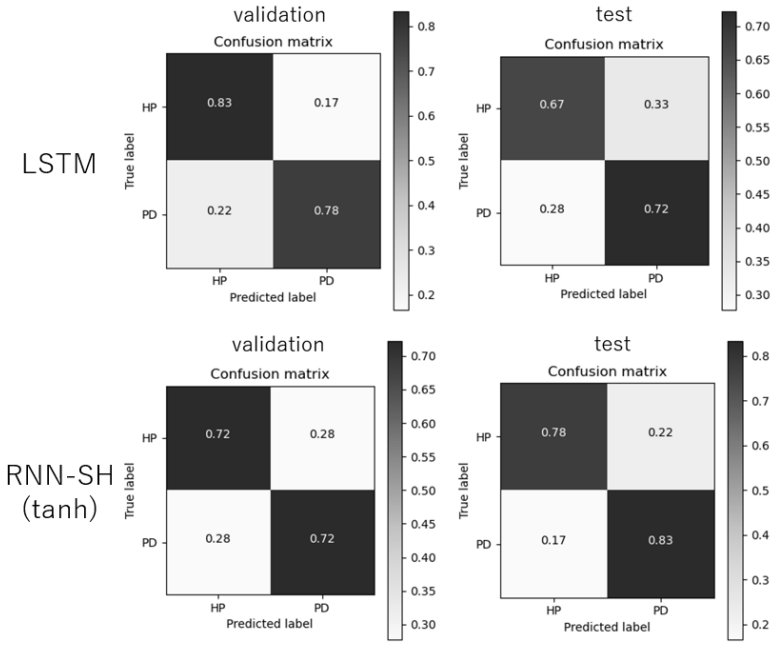

図 5.7: LSTM(2 層, 64 units)と RNN-SH(1 層, 256 units, tanh)の confusion matrix

RNN-SH の出力活性化関数に relu を用いた場合の性能は，tanh を用いた場合よりも低くなっている．このことに関する考察は，次の 5.5 節の考察にて述べる．

5.5 考察

図 5.8 に 50 回の学習回数における各 RNN モデルを使用した場合の学習曲線を示す．なお，図 5.8 に示す学習曲線はいずれの RNN モデルにおいても，1 回目の試行における結果となっている．図 5.8 から，RNN-SH(tanh)と RNN-SH(relu)を使用した場合では，LSTM や GRU と比較的してゆっくりと学習が進んでいる様子が見て取れる．RNN-SH ではパラメータが少なく，行列重みパラメータを入力に対してのみ持つことから，比較的緩やかに学習を進行させることができ，過学習に対して対処が容易であると考えられる．ただし，2 層の場合の RNN-SH(tanh)では学習曲線が大きく振動していることが確認できる．これは出力活性化関数に tanh を用いたことによって PD 検出モデル全体で勾配消失を起こし，悪影響を及ぼしたためと考えられる．一方で出力活性化関数に relu を用いた RNN-SH の場合では，図 5.8 では比較的なめらかな学習曲線であり，うまく学習できているように見受けられる．しかし，relu を使用した場合の PD 検出モデルの精度は，表 5.2 から他のモデルと比べて低くなっている．ここで図 5.9 に，1 回目の試行における RNN-SH の出力活性化関数に tanh と relu を用いた場合の正規化された confusion matrix の比較のグラフを示す．図 5.9 から検証データセットにおける confusion matrix において，relu を用いた場合の方が tanh を用いた場合より FP(偽陽性)の割合が高く，TN(真陰性)の割合が低いことから，relu を使用した RNN-SH では PD 患者を正しく識別できず，学習がうまく行えなかったことが確認できる．この原因として，relu を用いた場合では，relu からの出力値が 0 になることで活性化しなくなり，逆伝搬時の勾配が消失してしまう dying relu[47]が発生してしまったものと考えている．本章の実験では，relu に最適なパラメータを探すことは行っていないため，RNN-SH の重みパラメータの初期値が最適ではなかった可能性がある．relu は多層化に貢献する活性化関数として注目されているが，原点に対して対称でないことから正規化手法を relu と同時に用いることも多く，今後どのような手法や初期値と組み合わせることで relu を用いた場合でも RNN-SH の性能が発揮できるかを詳しく調査する必要があると考えている．

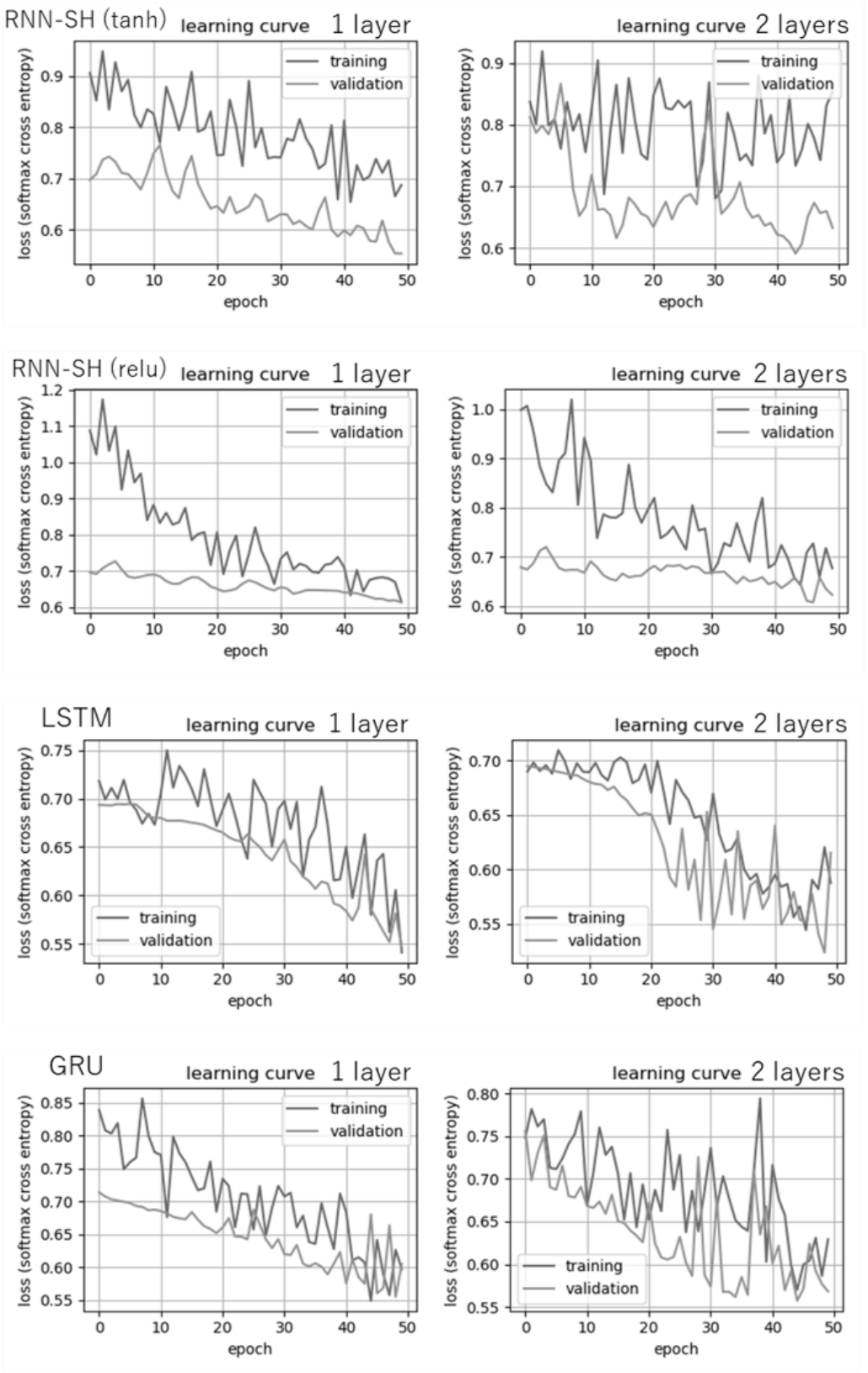

図 5.8: 1 回目の試行における各モデルを使用した場合の学習曲線

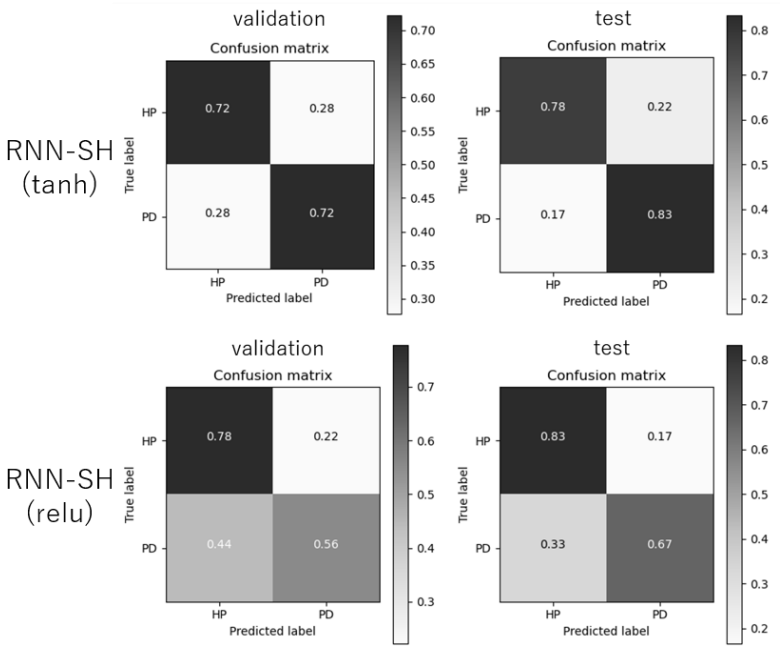

図 5.9: 1 層, 256 units の RNN-SH(tanh)と RNN-SH(relu)の confusion matrix

PD 検出モデルの精度に関しては，実験全体を通した精度は約 70%程度と実用レベルには到達しなかった．この原因として，本章の実験に用いた単母音である /a/ において長時間の発声が健常者であっても比較的難しく，かすれ声や声量が不安定になってしまうケースがあったことや，実験に用いたデータセットのサイズが小さかったことが原因ではないかと考えている．したがって今後は，どのような音声が PD 検出に適するのかを調査し，より音声データを増やしたデータセットを用いることで検出モデルの性能をさらに検討する必要がある．また実験では，VRAM のメモリ容量を考慮してバイリニア補間を用いた RP 画像のリサイズを行ったが，リサイズされた RP 画像では元の画像にあった特徴が失われたり，全く別の特徴が現れたりといった問題が生じることがあり，精度が悪化した可能性も考えられる．加えて RP 画像を作成する時間間隔として，0.01s という時間間隔が適切であったかどうかも今後検討しなければならないと考える．ゆえに，これらの画像圧縮方法や圧縮後の画像サイズや，音声分割における時間間隔の最適化，同時に CNN アーキテクチャによる画像のプーリングなどの PD 検出モデルにおける改善について更に検討することで，RP を用いた PD 検出により適した方法を模索する必要があると考えている．

5.6 おわりに

本章では 4 章で構築した sech 関数とスカラー値 a を用いたゲート構造を持つパラメータ量の少ない RNN の 1 つにおいて，自然言語処理以外のタスクにおける性能を測るため，時系列データ処理や画像処理の要素を含むタスクとして，音声波形データから生成された RP 画像を用いる PD 検出タスクを設定し，RNN の性能評価を行った．実験の結果から，従来型のゲート構造をもつ LSTM や GRU との性能比較では，出力活性化関数に tanh を用いる RNN-SH を使用した PD 検出モデルにおいて，LSTM や GRU を用いた PD 検出モデルとの性能差は非常に小さく，統計的優位差は確認できなかった．しかしながら，RNN-SH を使用した PD 検出モデルのパラメータの総数は他のゲート付き RNN よりも非常に少なく，計算速度及びメモリ効率ともに非常に有利な結果が得られた．したがって，以上のことから RNN-SH は時系列データ処理や画像処理などのタスクにも応用可能であることが示唆される．

一方で RNN-SH の活性化関数として relu を用いた場合の PD 検出性能は tanh を用いた場合の性能よりも低く，単純に活性化関数を tanh から relu に置き換えるだけでは多層化による性能向上を実現することは難しいとわかった．この原因は relu における dying relu 問題によるものと考えられるが，パラメータの初期化や正規化法，より優れた RNN 構造の開発なども含め，さらに詳細な実験と検討を行う必要がある．

PD 検出精度は，PD 検出に適した単母音の選択，RP 画像を作成する時間間隔，最適な RP 画像のサイズ，CNN の構造の最適化などにより，更に精度を向上させることができると考えられる．また実験ではデータセットのサイズが比較的小さかったため，データセットを増やすことでも PD 検出能力の改善が見込まれる．その他，計算機リソースが十分にある場合には，時間周波数解析における短時間フーリエ変換などの他手法による特徴抽出と RP における特徴抽出を組み合わせることで，更に検出精度を向上させることも可能であり，今後はこれらの改善すべき点についてより詳細に調査を行い，検出精度の改善を行う予定である．

第6章 結論

本論文は，複雑化するニューラルネットワークモデルの中でも特に RNN モデルに着目し，そのパラメータを削減するべく行われた計算コスト・メモリ使用量の少ない RNN モデルの構築及び構造解析と評価に関しての 3 つの研究をまとめたものである．RNN はその汎用性の高さから様々なタスクに応用が可能であるが，計算コスト及び計算機におけるメモリ容量の観点から制約を受けることも少なくない．

第 1 章では，近年のデータセットの大規模化やニューラルネットワークの設計傾向と RNN 構造における性能解析及びパラメータの削減の重要性について述べた．また，それに伴う研究の目的，意義や特徴について論じた．

第 2 章では，主要な RNN の構造とその特徴について述べ，それぞれの RNN の問題点や必要とされるパラメータについて説明した．加えて，本研究におけるパラメータ削減の意義と方向性について論じた．

第 3 章では，隠れ状態に対する重みパラメータを 1.0 に固定した Simple RNN に LSTM や GRU などの RNN モデルが持つ既存のゲート構造を挿入し，入力に対して重みを持つもしくは持たない，並びに 1 ゲート(GRU 相当)もしくは 2 ゲート(LSTM 相当)のゲート構造を有する新しい構造の RNN を 4 種類構築することで，従来型の RNN から性能を落とすことなくパラメータを削減することを試みた．性能評価では複合的な要素を持つ機械翻訳タスクとして，Tanaka corpus を使用した英語から日本語への翻訳により各 RNN 間での性能比較実験を行った．その結果として，比較的小さなコーパスではあるものの，複合タスクである機械翻訳タスクにおいて，構築した一部の SGR が従来型のゲート構造を持つ LSTM や GRU よりも優れた精度を示す結果となった．しかしながら多層化した場合の精度は期待通りには向上せず，更なる検討が必要であり，また SGR の精度の確保には比較的大きなユニット数が必要であることから，従来型の RNN から大幅にパラメータを削減するには至らなかった．この問題を解消するためにはゲート構造が性能に及ぼす影響について考慮し，ゲート構造そのものの改善について検討する必要があると結論付けた．

第 4 章では，第 3 章の結果及び考察から，ゲート構造の利点に着目しつつ，解析な観点からゲート構造を簡単化した新しい RNN を構築した上で，ゲート構造が性能に及ぼす影響，並びに更なるパラメータ削減の余地について検討を行った．具体的には，shortcut connection を持つ単純な構造の RNN に対して，従来の行列重みパラメータと sigmoid 関数を用いたゲート構造ではなく，代わりにスカラー値 a とこれまで活性化関数として用いられてこなかった sech 関数により構成された，全く新しいゲート構造を挿入することでパラメータの少ない RNN を構築した．従来のゲート構造における行列重みをスカラー値 a に単純に置き換えた場合では，sigmoid 関数の特性により正と負のスケーリングが正しく機能しない可能性があったが，偶関数である sech 関数を導入することで，スカラー値 a を用いた場合でもゲートとして正常に機能するものと考えたからである．この sech 関数とスカラー

値 a を用いたゲート構造を有する RNN は，隠れ状態に対して重みを持つか持たないか，並びに出力活性化関数を shortcut connection の前後どちらに適用するかという観点から 4 種類構築された．そしてその性能を，WikiText-2 を用いた言語モデリングタスクと IMDB を用いた 2 値分類タスクという性質の異なる 2 つの自然言語処理タスクを用いて評価し，従来型のゲートを持つ LSTM や GRU と比較した．その結果，文章間の長期の依存関係だけではなく単語間の比較的短い依存関係もとらえる必要がある WikiText-2 を用いた言語モデリングタスクでは，LSTM や GRU に劣る結果となってしまった．しかしながら，文章における長期の依存関係をとらえ最終時刻の出力が重要となる IMDB を用いた 2 値分類タスクの場合では，パラメータが 1/6 以下と少ないにも関わらず LSTM や GRU に匹敵する精度となった．この結果から，ゲート構造の改善によって更なるパラメータの削減が可能であることが示された．ゲート構造が sech 関数とスカラー値 a に置き換えられたことにより解析も容易となっており，ゲート構造の解析により判明した RNN モデル出力時の勾配消失問題の緩和などにより，言語モデリングタスクの性能も更に精度を向上させることが可能であると考えられ，出力活性化関数の選択や多層化時の構造について更なる検討を要することを述べた．加えて自然言語処理以外のタスクにおける性能を確認する必要があることも論じた．

第 4 章における自然言語処理タスクおける 2 値分類タスクにおいて sech 関数とスカラー値 a を持つ RNN が有効に機能することが確認できたが，時系列データ処理や画像処理などのタスクにも有効であるかを確認する必要がある．そのため第 5 章では，時系列データ処理や画像処理を含む応用的な医療情報処理タスクとしてパーキンソン病(PD)検出を設定し，sech 関数とスカラー値 a を持つ RNN の内で，第 4 章の 2 値分類タスクにおいて最も性能の良かった RNN を RNN-SH と呼称し，この RNN-SH を用いて更に検証を進めた．ここでは PD の運動障害性構音障害に着目し，PD 患者の音声データを特定区間に分割したものを 2 次元 RP 画像に変換した上で RNN を用いた PD 検出モデルにより検出し，その検出精度を従来型の RNN と比較することで性能を検討し評価した．その結果，LSTM，GRU，RNN-SH における各モデル間での統計的優位差は認められず，性能差が極めて小さいという結果が得られた．ゆえに，RNN-SH のパラメータは GRU を用いた場合の 1/3 以下であることから，パラメータが少ないにも関わらず従来型のゲートを持つ RNN に匹敵する精度となった．このことから，RNN-SH が自然言語処理のみならず，時系列データ処理や画像処理などのタスクにも応用可能であると示唆される．しかしながら，第 4 章でも問題となっていた RNN-SH の多層化において，多層化に重要とされる活性化関数である relu 関数を用いた場合でも検出精度を向上させることはできず，パラメータの初期化や正規化法を検討し，より詳細な調査を行う必要があるとわかった．PD 検出の精度に関しては 70%前後であったが，精度が伸び悩んだ原因として PD 検出に適した単母音の選択，RP 画像を作成する時間間隔，RP 画像のサイズ，PD 検出モデルにおける CNN の構造が最適ではなかった可能性があり，これらを最適化すれば更に精度を向上させることが可能であると考えられる．また

その他にも，計算機リソースが潤沢にあれば，時間周波数解析などの他手法との組み合わせにより精度向上が可能であると考えられ，実験に用いたデータセットを今後更に増やした上で，改善すべき点について詳細な調査・検討を行い，PD 検出精度の改善を行う予定となっていることを述べた．

以上の第 3 - 5 章の研究により，RNN 構造を最適化し，パラメータの削減が可能であることが示されたと考えられる．また特定のタスクにおいてではあるが，本研究は RNN のパラメータ削減並びにゲート構造の解析について貢献できたものと考えている．しかしながら，未だすべてのタスクにおいて従来型の RNN に匹敵あるいは超える性能を持つ，普遍的に使用可能な RNN 構造には至っておらず，様々な観点から更に構造の改善を行う必要がある．本研究では SRU や MGU などの LSTM や GRU 以外のゲート構造を有する RNN との性能比較は行っていないため，今後は他のゲートを有する RNN についても様々な観点から実験及び性能比較を行い，性能向上への更なる改善点を見つけたい．特に多層化した場合の精度向上は，パラメータを増やすことで性能を向上させることができることと概ね等しいため，近年のニューラルネットワークモデルにおける設計思想からも非常に重要だと考えている．これは本研究の結果から RNN 出力時の勾配消失を抑えることができれば解決可能であると期待されるが，他の要因と関連している可能性もあり，今後より詳細な実験及び調査を行いたいと考えている．

謝辞

本研究を進めるにあたり，お忙しい中，研究の始まりから論文の取りまとめに至るまで，熱心にご指導ご鞭撻を賜りました神戸大学システム情報学研究科 羅志偉 教授に心より厚く御礼申し上げます．また，本研究，本論文の執筆において数々のご助言をいただきご指導いただきました神戸大学システム情報学研究科 全昌勤 准教授に謹んで感謝申し上げます．本論文の作成に関して，丁寧にご指導・ご教示いただきました神戸大学システム情報学研究科 滝口哲也 教授に厚く御礼申し上げます．丁寧にご指導・ご教示いただきました神戸大学システム情報学研究科 的場修 教授に厚く感謝申し上げます．

システム制御研究室において，様々なご助言，ご指導をいただきました，森耕平 助教授，曹晟 助教授に深い感謝の意を申し上げます．また，同研究室の皆様にも数多くの協力をしていただきましたことを深く感謝いたします．

最後になりますが，大学生活においてお世話になった先生方や職員の方々，データの収集にご協力いただきました協力者の皆様に深く感謝いたします．

参考文献

[1] J. L. Elman: Finding Structure in Time; *Cognitive Science*, 14 (2), pp. 179-211 (1990)

[2] S. Hochreiter and J. Schmidhuber: LONG SHORT-TERM MEMORY; *Neural Computation*, 9 (8), pp. 1735-1780 (1997)

[3] F. A. Gers, J. Schmidhuber, and F. Cumminsm: Learning to Forget; Continual Prediction with LSTM; *Neural Computation*, 12 (10), pp. 2451-2471 (2000)

[4] K. Cho, B. V. Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio: Learning phrase representations using rnn encoder-decoder for statistical machine translation; *Conference on Empirical Methods in Natural Language Processing*, pp. 1724–1734 (2014)

[5] A. Krizhevsky, I. Sutskever, and G. E. Hinton: Imagenet classification with deep convolutional neural networks; *Advances in neural information processing systems*, 25, pp. 1097-1105 (2012)

[6] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, and A. Rabinovich: Going deeper with convolutions; *In Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1-9 (2015)

[7] K. He, X. Zhang, S. Ren, and J. Sun: Deep Residual Learning for Image Recognition; *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770-778 (2016)

[8] R. Jozefowicz, W. Zaremba, and I. Sutskever: An empirical exploration of recurrent network architectures; International Conference on Machine Learning, pp. 2342–2350 (2015)

[9] G. B. Zhou, J. X. Wu, C. L. Zhang, and Z. H. Zhou: Minimal gated unit for recurrent neural networks; International Journal of Automation and Computing, Vol. 13, No. 3, pp. 226–234 (2016)

[10] T. Lei, Y. Zhang, S. I. Wang, H. Dai, and Y. Artzi: Simple recurrent units for highly parallelizable recurrence; Conference on Empirical Methods in Natural Language Processing, pp.4470-4481 (2018)

[11] B. Yue, J. Fu, and J. Liang: Residual recurrent neural networks for learning sequential representations; Information, 9(3):56 (2018)

[12] S. Li, W. Li, C. Cook, C. Zhu, and Y. Gao: Independently recurrent neural network (indrnn): Building a longer and deeper rnn; Proc.IEEE conference on computer vision and pattern recognition, pp. 5457-5466 (2018)

[13] S. Ioffe, and S. Christian: Batch normalization: Accelerating deep network training by reducing internal covariate shift; International conference on machine learning. PMLR

(2015)

[14] J. Chung, C. Gulcehre, K., Cho, and Y. Bengio: Empirical evaluation of gated recurrent neural networks on sequence modeling; arXiv preprint arXiv:1412.3555 (2014)

[15] R. Jozefowicz, W. Zarembam, and I. Sutskever: An empirical exploration of recurrent network architectures; Proceedings of the 32nd International Conference on Machine Learning, pp. 2342-2350 (2015)

[16] G. Weiss, Y. Goldberg, and E. Yahav: On the practical computational power of finite precision RNNs for language recognition; arXiv preprint arXiv:1805.04908 (2018)

[17] T. Fujita, Z. Luo, C. Quan, and K. Mori: Simplification of RNN and Its Performance Evaluation in Machine Translation; Transactions of the Institute of Systems, Control and Information Engineers, 33(10), pp.267-274 (2020)

[18] J. C. Heck and F. M. Salem: Simplified minimal gated unit variations for recurrent neural networks; 2017 IEEE 60th International Midwest Symposium on Circuits Syst., pp. 1593-1596 (2017)

[19] R. Dey and F. M. Salem: Gate-variants of gated recurrent unit (GRU) neural networks; 2017 IEEE 60th International Midwest Symposium on Circuits Syst., pp. 1597-1600 (2017)

[20] I. Sutskever, O. Vinyals, and Q. V. Le: Sequence to sequence learning with neural networks; Advances in neural information processing systems, pp. 3104-3112 (2014)

[21] Y. Tanaka: Compilation of a multilingual parallel corpus; Proceedings of PACLING 2001, pp. 265–268 (2001)

[22] MeCab: Yet Another Part-of-Speech and Morphological Analyzer. http://taku910.github.io/mecab/, (2021-5-25)

[23] Natural Language Toolkit. https://www.nltk.org/, (2021-5-25)

[24] I. Loshchilov and F. Hutter: Decoupled Weight Decay Regularization; The International Conference on Learning Representations (2019)

[25] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov: Dropout: a simple way to prevent neural networks from overfitting; The journal of machine learning research, 15(1), pp. 1929-1958 (2014)

[26] Y. LeCun, L. Bottou, G. B. Orr, and K. Muller: Efficient BackProp; Neural Networks; Tricks of the Trade, pp. 9-48 (1998)

[27] K. Papineni, S. Roukos, T. Ward, and W. Zhu: Bleu: a Method for Automatic Evaluation of Machine Translation; Proceedings of the 40th annual meeting of the Association for Computational Linguistics, pp. 311-318 (2002)

[28] T. Fujita, Z. Luo, C. Quan, K. Mori: Structure Construction and Performance Analysis of RNN Aiming for Reduction of Calculation Costs; Transactions of the Institute of

Systems, Control and Information Engineers, 34(4), pp.89-97 (2021)

[29] A. Veit, M. J. Wilber, S. Belongie: Residual networks behave like ensembles of relatively shallow networks; In Advances in neural information processing systems, pp. 550-558 (2016)

[30] D. Balduzzi, M. Frean, L. Leary, J. Lewis, K. W.-D. Ma, B. McWilliams: The Shattered Gradients Problem: If resnets are the answer, then what is the question?; arXiv preprint arXiv:1702.08591 (2017)

[31] S. Merity, C. Xiong, J. Bradbury, and R. Socher: Pointer sentinel mixture models; arXiv preprint arXiv:1609.07843, (2016)

[32] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts: Learning Word Vectors for Sentiment Analysis; Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, pp. 142-150 (2011)

[33] T. Fujita, Z. Luo, C. Quan, K. Mori, and S. Cao: Performance Evaluation of RNN with Hyperbolic Secant in Gate Structure through Application of Parkinson's Disease Detection; Applied Sciences, 11(10), 4361 (2021)

[34] J. P. Eckmann, S. O. Kamphorst, and D. Ruelle: Recurrence plots of dynamical systems; World Scientific Series on Nonlinear Science Series A, 16, pp. 441-446 (1995)

[35] 城真範, 平田祥人, 合原一幸; リカレンスプロットとフーリエ変換の類似性と相違性: 生産研究, 72(2), pp. 137-138 (2020)

[36] A. Facchini, H. Kantz, and E. Tiezzi: Recurrence plot analysis of nonstationary data: The understanding of curved patterns; Physical Review E, 72(2), 021915 (2005)

[37] L. Jeancolas, D. Petrovska-Delacrétaz, G. Mangone, B. E. Benkelfat, J. C. Corvol, M. Vidailhet, S. Lehéricy, and H. Benali: X-Vectors: new quantitative biomarkers for early Parkinson's disease detection from speech; Frontiers in Neuroinformatics, 15, 4 (2021)

[38] J. S. Almeida, P. P. Rebouças Filho, T. Carneiro, W. Wei, R. Damaševičius, R. Maskeliūnas, and V. H. C. de Albuquerque: Detecting Parkinson's disease with sustained phonation and speech signals using machine learning techniques; Pattern Recognition Letters, 125, pp. 55-62 (2019)

[39] L. Berus, S. Klancnik, M. Brezocnik, and M. Ficko: Classifying Parkinson's disease based on acoustic measures using artificial neural networks; Sensors, 19(1), 16 (2019)

[40] S. Grover, S. Bhartia, A. Yadav, and K. R. Seeja: Predicting severity of Parkinson's disease using deep learning; Procedia computer science, 132, pp. 1788-1794 (2018)

[41] E. Garcia-Ceja, M. Z. Uddin, and J. Torresen: Classification of recurrence plots' distance matrices with a convolutional neural network for activity recognition; Procedia computer science, 130, pp. 157-163 (2018)

[42] Y. Chen, S. Su, H. Yang: Convolutional neural network analysis of recurrence plots for anomaly detection; International Journal of Bifurcation and Chaos, 30(01), 2050002 (2020)
[43] L. C. Afonso, G. H. Rosa, C. R. Pereira, S.A. Weber, C. Hook, V. H. C. Albuquerque, and J. P. Papa: A recurrence plot-based approach for Parkinson's disease identification; Futur. Gener. Comput. Syst. 94, pp. 282–292 (2019)
[44] Pydub. https://github.com/jiaaro/pydub, (2021-5-8)
[45] FFmpeg. https://github.com/FFmpeg/FFmpeg, (2021-5-8)
[46] L. Liu, H. Jiang, P. He, W. Chen, X. Liu, J. Gao, J. Han: On the variance of the adaptive learning rate and beyond; arXiv preprint arXiv:1908.03265 (2019)
[47] CS231n Convolutional Neural Networks for Visual Recognition Course Website https://cs231n.github.io/neural-networks-1/#actfun (2021-5-12)

神戸大学博士論文「高効率な深層学習を目指す RNN の構造設計と性能評価」全 82 頁
提 出 日 ２０２１年7月１３日