# RNNLMを用いた日本語テキストの誤字・脱字検出および再変換を用いた誤変換検出

14173020　白井 稔久　　指導教員　小町 守 准教授

平成30年9月2日

**概 要**

近年パーソナルコンピュータやスマートフォンを用いて文章を記述する機会が多くなった．しかし，日本語母語話者であっても，日本語テキストを記述する際誤って記述してしまう場合がある．これを人手で検出することは非常にコストがかかる．そこで我々は日本語母語話者が記述したテキストの誤字・脱字の自動検出を行う．日本語母語話者が記述したテキストの誤字・脱字がアノテーションされているコーパスが少なく，教師あり学習の手法を行うことが困難である．そこで我々はRNN言語モデルを用いた日本語母語話者が記述したテキストの誤字・脱字検出と，再変換を用いた日本語母語話者が記述したテキストの誤変換検出を提案する．

## 1　はじめに

文章には誤りが含まれていることがある．例えば,「ペ**ルー**は1853年に浦賀に来航した.」のように「ペ**リー**」が「ペ**ルー**」と誤って記述されているものや,「今日はホテルに**泊**まります」が「今日はホテルに**止**まります」のように「**と**まる」が誤変換されてしまっているものである．公的に情報を発信する際に前述のような誤りを含んでいると，その情報の受信者が混乱してしまう恐れがある．そのため，これらの誤りは情報を発信する前に校正する必要がある．しかし，このような誤りを人手で検出しようとすると大きなコストがかかる．日本語学習者の作文の誤り検出や誤り訂正の研究は進められているが[7][10]，日本語母語話者におけるそれらの研究はまだ発展途上である．また，誤りがアノテーションされたコーパスも少ないため，教師あり学習による誤り検出を行うことは困難である．そこで我々は日本語母語話者が記述したテキストの誤字・脱字と誤変換に着目してそれらを検出する手法を提案する．

日本語テキストの誤り検出の先行研究として文字$n$-gramを用いた教師なし手法がある[9]．しかし，文字$n$-gramを用いた手法の問題点として近くの文字しか依存関係を考慮できないことが挙げられる．そこで我々は，長期依存関係を考慮できるリカレントニューラルネットワーク（RNN）を用いることを提案する．長期依存関係を考慮することによって,「浦賀」や「来航」と「ペ**リー**」や「ペ**ルー**」との共起を考慮することができ，上述のよ

うな例文でも誤りを検出できるようになると考える．また，上述の例は誤字であるが，脱字についても文字のつながりが不自然になるため，脱字についても検出できると考える．そこで，我々は文字ベースの RNN 言語モデルを用いた誤字・脱字検出を提案する．

また，我々はデバイスを用いて日本語テキストを入力する際に起こりうる，誤変換を対象とした誤り検出の手法を提案する．我々が提案する手法は Kytea を用いて平仮名列に直した入力文を，Google 日本語入力を用いて漢字に再変換し，その候補の中で予測確率が最大の文節と入力文中の文節を比較し，誤変換の検出を行う．

# 2 関連研究

## 2.1 日本語における誤字・脱字

日本語母語話者が記述したテキストの誤り訂正の研究として，新納らは平仮名 $n$-gram を用いて誤りを検出し，訂正する手法を提案した [9]．この研究は我々の提案手法と同じく，日本語母語話者が記述したテキストを対象に誤りを検出している．一方で，我々の提案手法と $n$-gram を用いた点，文字全体ではなく平仮名だけを用いた点が異なる．

一方，日本語学習者が記述したテキストに対する誤り自動訂正の研究は広く進められている．日本語学習者の作文における誤りの自動訂正においては，今村らは間違えやすい助詞のフレーズテーブルを用いることによって訂正する手法を提案した [7]．また，南保らは文節内の特徴からルールを自動作成し，ルールベースで日本語の助詞誤りを検出し，訂正する手法を提案した [11]．今村ら [7]，南保ら [11] の研究と違い，我々の研究は日本語ネイティブの書いた文に存在するすべての誤りを検出の対象としている．水本ら [10] は，日本語学習者コーパスを用い統計的機械翻訳を適用した，文字単位での訂正と，文字-単語間での訂正の 2 つの手法を提案した．水本ら [10] の手法とは言語モデルを用いてすべての誤りを検出する点で共通しているが，我々の提案手法は教師なし手法であり，日本語ネイティブの書いた大規模な生テキストを用いる点が異なっている．

また，鈴木らは日本語の統計的機械翻訳の出力結果中の格助詞を，分類器を用いて訂正する手法と，原文を用いた統計的機械翻訳によって訂正する手法を提案した [2]．我々の研究と日本語の誤りを対象とした点は共通しているが，格助詞のみを訂正の対象とした点，手法が統計的機械翻訳と分類器を用いている点．統計的機械翻訳の出力文を対象として訂正を行った点で異なる．

## 2.2 日本語における誤変換

日本語母語話者が記述したテキストの誤変換検出・訂正の研究もまた進められている．

林らは入力文を単語ごとに分割し辞書からその単語の読みを予測し，その読みから辞書を用いて漢字の候補を出力，その候補における周辺の単語との単語 2-gram 辞書を用いて誤変換を検出する手法を提案した [8]．本研究と入力文を平仮名に戻してから漢字への変換候補を出力した点は共通するが，我々は単語 2-gram 辞書ではなく Mozc [6] で用いられている言語クラス 2-gram を用いており，またデータセットを自作せず既存のデータセットで実験を行っている．

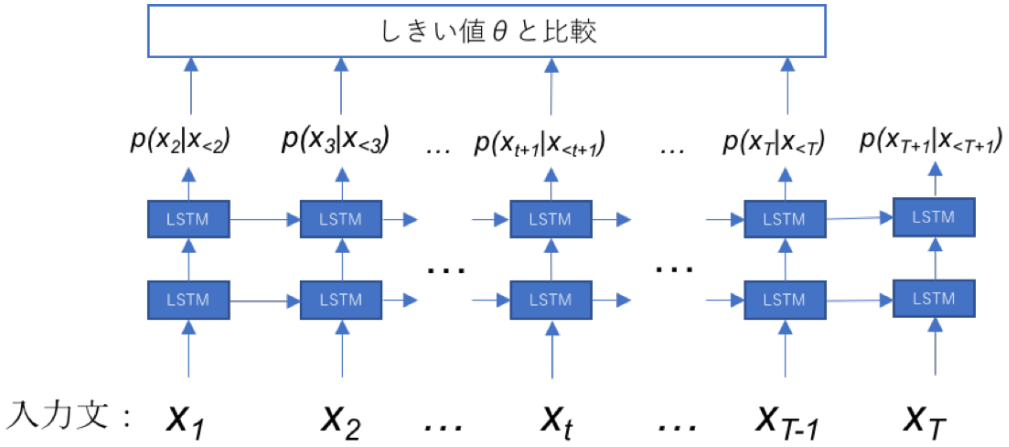


図 1: エラー検出の概略図

奥らは複合語に対してその複合語に同音異義語が存在する場合，文字 $n$-gram を用いて同音異義語誤りを検出する手法を提案した [4]．本研究と日本語母語話者が記述した日本語テキストの誤変換の検出を行った点は共通するが，我々は複合語のみではなく全ての誤変換対象とし，また文字 $n$-gram ではなく言語クラス 2-gram を用いた．

梶谷らは入力文を平仮名列変換しその後かな漢字変換ソフトを用いて変換候補を出力，その変換候補で Google 検索を行い上位 $N$ 件の Web ページ内の共起を用いて変換候補を選択し，誤変換を検出する手法を提案した [5]．本研究と再変換を用いて誤変換検出を行った点は共通するが，我々が実験時のデータセットを自作せず既存のデータセットを用いた点，我々の提案手法の，再変換を用いた誤変換検出の手法において誤警報率最小となるように上界を調べた点で異なる．

# 3　RNN 言語モデルを用いた誤り検出

## 3.1　RNN 言語モデル

言語モデルとは以前に入力された文字・単語から次の文字・単語を予測するモデルである．RNN 言語モデルを用いた誤字・脱字検出において我々は Zaremba ら [3] が提案した RNN 言語モデルを用いた．彼らが提案した RNN 言語モデルでは隠れ層として Long Short-Term Memory（LSTM）を 2 層に重ねたものが用いられている．1 層目の LSTM に文字を 1 つずつ入力する．そして各 2 層目の LSTM が前の 2 層目の LSTM と 1 層目の LSTM から情報を受け取り，次の文字の確率分布を予測する．

図 1 に誤字・脱字検出を行う提案手法の概略図を示す．$x_t$ は時間 $t$ に入力される文字を示す．また $p(x_{t+1}|x_{<t+1})$ は時間 $t$ における次の入力文字 $x_{t+1}$ より前の文字を考慮した条

件付き確率を示す．

$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \text{sigm} \\ \text{sigm} \\ \text{sigm} \\ \tanh \end{pmatrix} \odot T_{2n,4n} \begin{pmatrix} D(h_t^{l-1}) \\ h_{t-1}^l \end{pmatrix} \tag{1}$$

$$c_t^l = f \odot c_{t-1}^l + i \odot g \tag{2}$$

$$h_t^l = o \odot \tanh(c_t^l) \tag{3}$$

$h_t^l \in \mathbb{R}^n$ は時間 $t$ の $l$ 層目の LSTM における $n$ 次元の隠れ層を表す．加えて，$T_{n,m} : \mathbb{R}^n \to \mathbb{R}^m$ は，$Wx + b$ のように入力 $x$ に対して重み行列 $W$ による $n$ 次元から $m$ 次元への線形変換にバイアス $b$ を加えたものである．

我々が用いた RNN 言語モデル内の LSTM は，直前のステップの，LSTM の隠れ層の値とメモリセルの値，1 つ前の層における隠れ層の値を受け取り，隠れ層とメモリセルを求める．$i$, $f$, $o$, $g$, $c$ と $h$ はそれぞれ入力ゲート，忘却ゲート，出力ゲート，入力判断ゲート，メモリセルと隠れ層である．$D$ は dropout である．

隠れ層 $h_t^2$ を以下のように線形変換しソフトマックス関数を使い，次の文字の確率分布 $p_t$ を獲得する．

$$p_t = \text{softmax}(W_h h_t^2 + b_h) \tag{4}$$

$W_h \in \mathbb{R}^{v \times n}$ は重み行列であり，$b_h \in \mathbb{R}^{v \times 1}$ はバイアスである．また，$v$ は語彙サイズの次元数である．

RNN 言語モデルの誤差 $loss$ は以下の式で算出される．

$$loss = -\sum y_t \log p_t \tag{5}$$

$y_t$ は次の入力を表す正解ベクトルである．入力文字の次の文字を答えとし、次の文字の確率を最大化するように学習を行う．

LSTM を用いた言語モデルは長期的な関係性を考慮することができるので長期依存関係を考慮しなければ検出できない誤りを検出することができると考える．

### 3.2 誤り検出

本項では 3.1 項で述べた RNN 言語モデルを用いた誤り検出について説明する．時間 $t$ までの入力 $x_1 ... x_t$ を受け取り，次の入力の予測確率分布 $p_t$ を出力する．その予測確率分布から次の入力文字 $x_{t+1}$ の予測確率 $p(x_{t+1}|x_{<t+1})$ を取り出し，しきい値 $\theta$ と比較し，しきい値よりも低かった場合，その文字が誤っているかその文字を出力する前に別の文字が必要であると推測し，誤りとして検出する．

## 4 再変換を用いた日本語テキストの誤変換検出

本項では本研究における誤変換検出の提案手法について説明する．提案手法は以下に示す手順によって行われる．

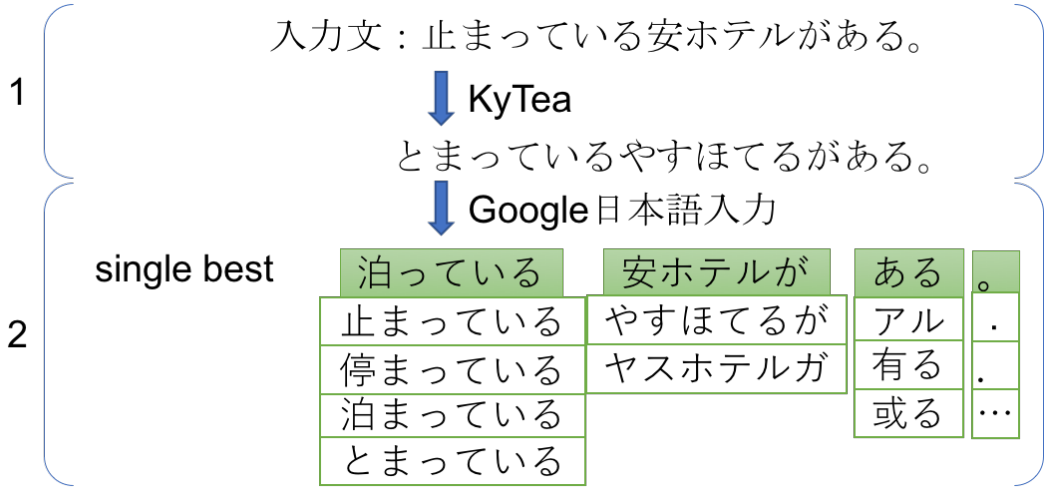


図 2: 再変換の概略図

1. 入力文を KyTea を用いて平仮名列に変換する．
2. 前項で平仮名列に直した入力文を，Google 日本語入力 API に入力し，変換候補の中から予測確率が一番高いものを選び，出力する．また，このとき変換候補は文節ごとに出力される．
3. 入力文の各文字に対応する出力文の各文字との編集距離が最小となるように動的計画法を用いて 2 文間のアライメントとそのときの保持，置換，挿入，削除の処理を取得する．
4. 手順 3 で得られた処理から保持でないものを誤りとして検出する．

手順 1，2 の概略図を図 2 に示す．

## 4.1 平仮名への変換

本研究では入力文から平仮名列への変換のために，単語の読み推定を行うことができる京都テキスト解析ツールキット KyTea [1] を用いた．KyTea は読み推定を行うためにまず入力文に対して単語分割を行う．その後，入力文の各単語が既知のものであればコーパスまたは辞書内の情報と $n$-gram を用いて読みを推定する．未知のものである場合文字から読みを推定する．

KyTea はかな漢字混じりの日本語文を入力すると，その日本語文の読みを推定し平仮名列として出力する．本研究にはこの出力された平仮名文を再変換することによって誤変換検出を行った．

表 1: 実験データの文数．括弧内は誤っている文字数を示す．

| | RNNLM | 再変換 |
|---|---|---|
| 学習 | 668,631 文 | |
| 開発 | 29,716 文（22） | 29,642 文（24） |
| 実験 | 29,715 文（33） | 29,638 文（31） |

表 2: 実験データの誤りの種類の内訳．文字単位で誤りを数えている．

| 誤りの種類 | 誤変換 | 誤変換以外の誤字 | 脱字 | 余字 | 人名誤り | 総数 |
|---|---|---|---|---|---|---|
| 誤りの数 | 15 | 16 | 3 | 14 | 7 | 55 |

### 4.2 漢字を含む文字列への変換

本研究では平仮名列から漢字を含む出力文への変換のために Google 日本語入力 API を用いた．Google 日本語入力 API は平仮名列をクエリとして送ると，その平仮名列を漢字を含む文に変換したものが出力される．また出力は文節単位で行われ，各文節ごとに変換候補を最大 5 個まで出力する．再変換を用いた誤変換検出ではそれらの変換候補を用いて誤変換検出を行う．

Google 日本語入力 API の変換システムには統計的かな漢字変換システム Mozc [6] が使われている．Mozc では言語クラスバイグラムと単語-読みユニグラムを用いて読みを推定している．言語クラスとは一般的には品詞や活用形などで，Mozc においてはいくつかのルールに基づき約 3,000 クラスに分類している．

## 5 提案手法を用いた誤字・脱字および誤変換の検出実験

### 5.1 実験データ

本研究では実験データに日本語書き言葉均衡コーパス（BCCWJ）を用いた．RNN 言語モデルを用いた誤字・脱字検出における RNN 言語モデルの学習データには BCCWJ の非コアデータを用い，開発およびテストデータには誤りのアノテーションされているコアデータを用いた．学習の高速化のために学習データの文長を 100 文字で制限した．また，再変換を用いた誤変換検出に用いたデータは Google 日本語入力 API の文字制限が 50 文字であることから，平仮名に直した入力文を句読点で分割した際，50 文字を超える文は実験には用いなかった．

RNN 言語モデルで用いた文長制限する前の文数，文長制限した後の文数，学習データ，開発データ，テストデータの分割，再変換を用いた提案手法の実験データを表 1 に示す．また，55 件の誤りの種類の内訳を表 2 に示す．

## 5.2 実験設定

RNN 言語モデルを用いた誤字・脱字検出における RNN 言語モデルの埋め込み層と隠れ層は 650 次元でパラメータは 0.1 から −0.1 の間のランダムな値で初期化した．また，dropout の係数は 0.5 で，勾配の大きさは 5 でクリップし，バッチサイズは 150 で学習した．最適化には SGD を用いた．RNN 言語モデルでは，出現回数が 20 回未満の文字は未知文字として処理し，語彙サイズは 6,704 文字である．未知文字は同一文字<unk>として処理した．開発データを用いて，しきい値 $\theta$ は $10^{-2}$ から $10^{-6}$ に決めて実験した．

## 5.3 評価

本研究の提案手法を評価するために，我々は再現率と誤警報率で評価した．再現率，誤警報率は以下の式で算出される．

$$\text{再現率} = \frac{(\text{検出できた誤り})}{(\text{検出できた誤り}) + (\text{検出できなかった誤り})} \tag{6}$$

$$\text{誤警報率} = \frac{(\text{誤検出})}{(\text{検出できた誤り}) + (\text{誤検出})} \tag{7}$$

## 5.4 文字単位でのアライメント

再変換を用いた誤変換検出では評価のために 3 文間でアライメントを取る必要がある．本研究では，入力文，正解文，出力文間のアライメントをとるために編集距離を用いた．本研究では以下の手順で 3 文間のアライメントを取得した．

1. 入力文と正解文間で 4 節の手順 3 と同じくアライメントを取り，処理を取得する．
2. 入力文と出力文のアライメントを取得した文字のペア $w_1$，$w_2$ における処理と，入力文と正解文の $w_1$ における文字と処理を 3 文間のアライメントとして取得する．

この手順の概略図を図 3 に示す．

## 5.5 オラクル

本研究における提案手法の検出性能の上界を調べるために，正解文を用いて出力文の誤警報率が最も低くなるように漢字への変換を行った．これをオラクルと呼称し以下にオラクルの作成手順を示す．

オラクル作成手順の概略図を図 4 に示す．オラクルは 4 節と同手順で作成した出力文と正解文を用いて出力される．

1. 4 節と同手順で作成した出力文と，正解文のペアを作成する．

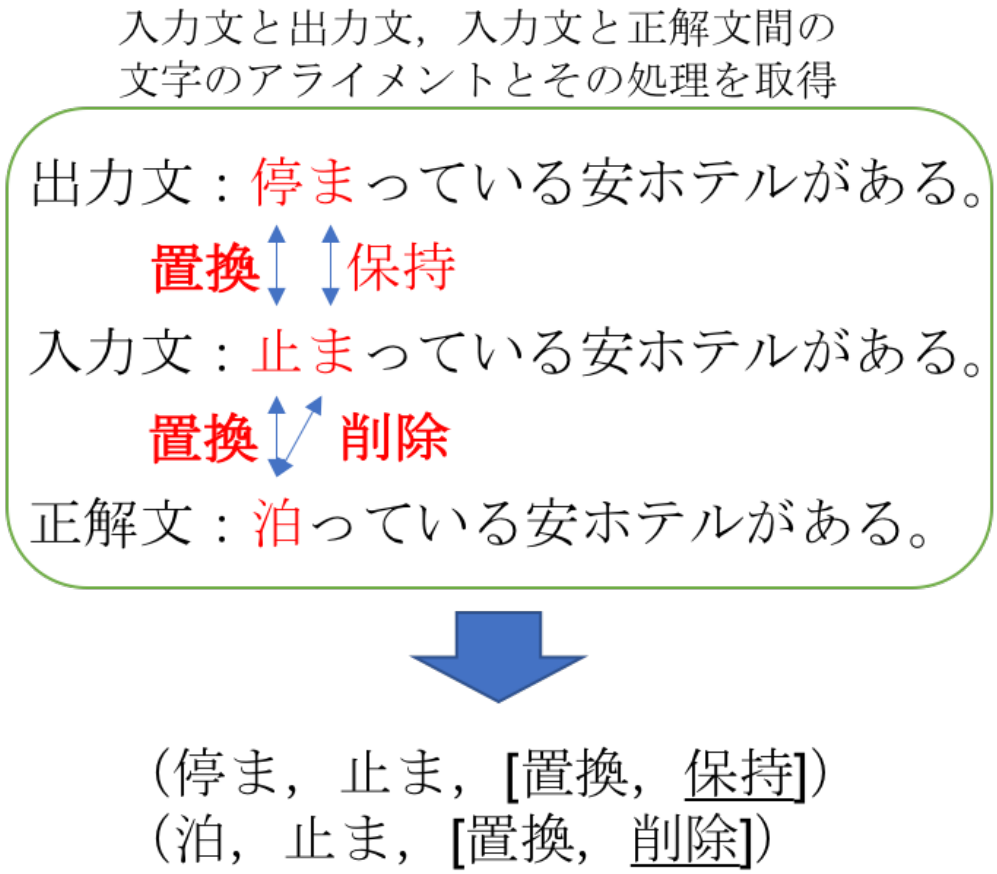


図 3: アライメント取得の概略図

2. 変換候補が全て文節で出力されるため，正解文を動的計画法を用いて文全体で合計した編集距離が最小となるように文節ごとに分ける．このとき分ける文節の数は出力文における文節の数と同数になるように分ける．

3. 文節ごとに分けた正解文と対応する変換候補を比較して，誤警報率が最小になる変換候補を出力する．

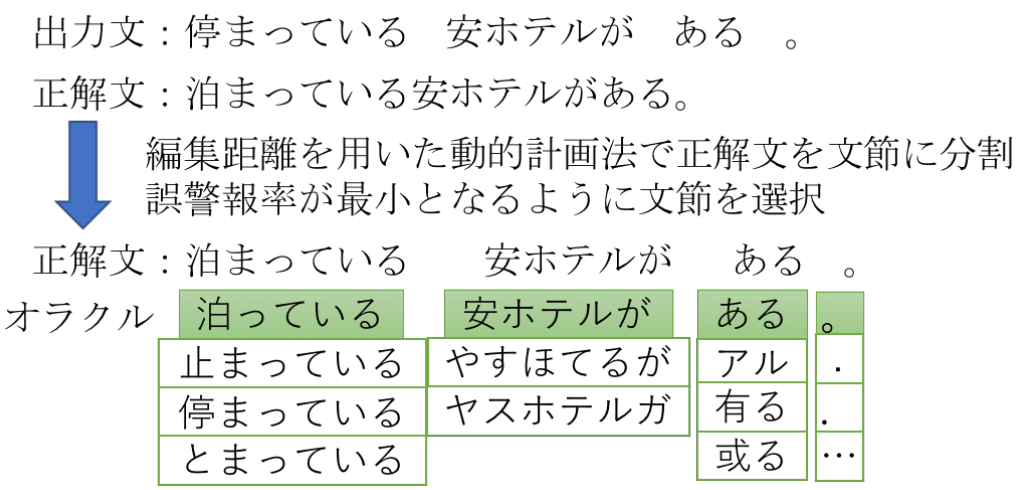


図 4: オラクル作成手順の概略図

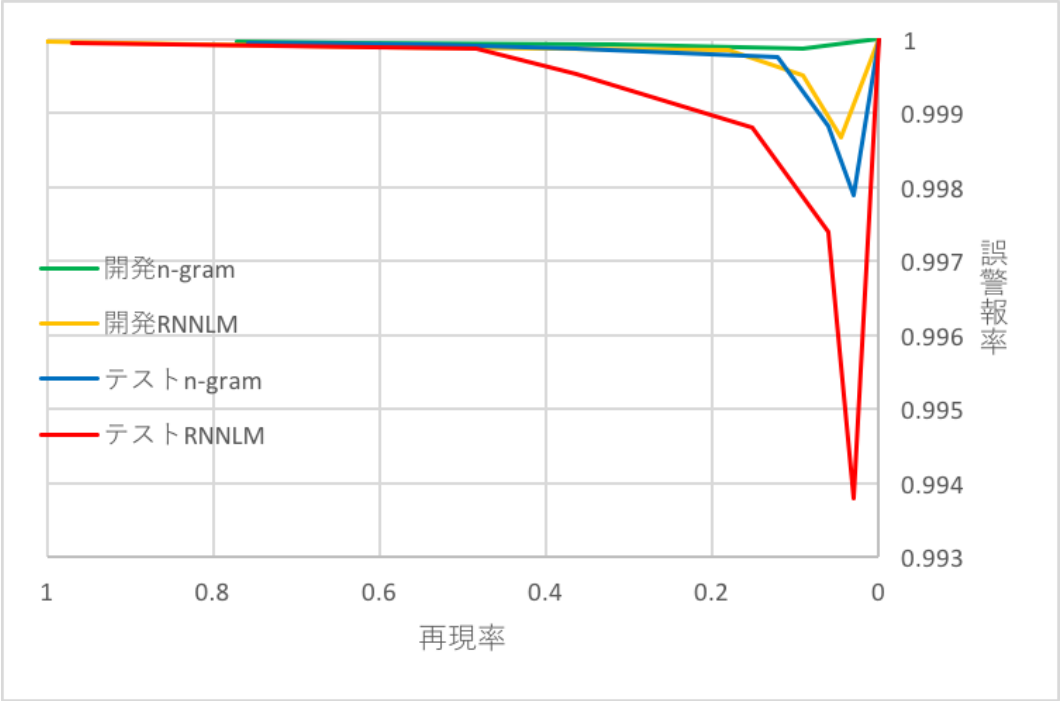


図 5: 言語モデルを用いた誤り検出結果

表 3: 再変換を用いた誤変換検出の実験結果

| | 開発 | | テスト | |
|---|---|---|---|---|
| | 最尤候補 | オラクル | 最尤候補 | オラクル |
| 再現率 | 0.39 | **0.47** | 0.46 | **0.47** |
| 誤警報率 | 0.21 | **0.16** | 0.19 | **0.16** |

### 5.6 実験結果

言語モデルを用いた誤字・脱字検出の実験結果を図 5 に示す．図 2 を見ると我々の提案手法がベースラインよりも再現率が同じときに誤警報率が低いことが分かる．一方で提案手法に関して誤警報率がすべての結果において 0.99 を超えていることが分かる．これはデータ中の誤りが少ないために誤検出が多く出てしまったものと考えられる．そして，しきい値を下げても誤警報率が下がらないということは，誤字・脱字と正しい文字の区別ができていないことを示している．この原因としてしきい値での検出は絶対的な値を参考にするため，文頭直後のような次の文字の予測確率分布が全体的に低いものはうまく検出できないのではないかと考える．

誤変換検出の実験結果を表 3 に示す．再変換を用いた誤変換検出では RNN 言語モデルを用いた誤字・脱字検出と比較して誤警報率が低くなっていることが分かる．これは再変換が入力文の平仮名列を取得しており，言語モデルを用いた手法よりもより多くの情報を考慮できるからだと考える．また，最尤候補とオラクルを比較するとそれほど差異がないことが分かる．このことから最尤候補の出力がオラクルと同等の性能を有していると考える．

## 6 実験結果の考察

### 6.1 RNN 言語モデルによる誤字・脱字検出

提案手法における誤字・脱字検出では文頭記号の直後の文字を誤りとして検出してしまうことが非常に多くあった．これは文の最初が様々な語から始まるために，ほとんど全て文字の予測確率が低くなってしまったためだと考える．追加実験としてしきい値ではなく RNN 言語モデルが計算した最尤候補と実際の入力の生成確率を負の対数尤度を用いて減算し，その値をしきい値と比較して検出を行う手法を試したが，文頭の誤検出は少なくなったが，他の誤検出はあまり変わらなかった．これは日本語の表記の自由度が高いため，選択する文字の候補が多くうまく推測できなかったのではないかと考える．

### 6.2 再変換を用いた誤変換検出

再変換を用いた誤変換検出における誤変換訂正の最尤候補をとるものは数字を全角で出力してしまう傾向にあった．よって元文の数字が半角であれば誤りとして誤認識してしまっていた．また，英字に関しても大文字かつ全角で出力する傾向にあった．また，記号

は平仮名の読みがついてしまうと漢字に変換され「■」などの記号は「資格」と誤変換されてしまう傾向にあった.「…」は平仮名にした際「てんてんてん」と読みがつき再変換の結果「 … 」となってしまうものも多く見られた．また，今回誤検出の対象にはしていないが人名の漢字も誤って出力する傾向にあった.

オラクルでは，最尤候補に見られた記号の誤変換は一部正しく出力できていたが，数字や英字に関しては最尤候補と同様に全角の出力が多く見られた.

# 7　おわりに

本研究では日本語母語話者が記述した日本語テキストの誤り検出に関する２つの手法を提案した．RNN 言語モデルを用いた提案手法においては，誤警報率が非常に高く，文字ベースの RNN 言語モデルでは誤字・脱字を正しく認識できないことが分かった．再変換を用いた提案手法においては再現率，誤警報率ともに高いスコアを記録することができ，また，最尤候補をそのまま出力したものがオラクルとほぼ同等の性能を持っていることが分かった.

RNN 言語モデルを用いた誤字・脱字検出においては日本語の表記の自由度が高いため，最尤候補と比較する際に最尤候補だけではなくいくつかの上位候補と比較し検出を行うことにより，出力されやすい候補を多く考慮することができ，より良い結果が得られると考える．再変換を用いた誤変換検出については最尤候補を用いたものが誤変換検出の上界とほぼ同等の性能を有しているため，改善の余地は少ないと考える．今後は教師あり学習を用いて誤り検出を行えるよう，誤りがより多くアノテーションされたデータセットの作成を考えている.

# 謝辞

何度もご迷惑をおかけしましたが根気強く未熟な自分にご指導してくださった小町先生，ありがとうございました．今後ともご指導ご鞭撻のほど，よろしくお願いいたします．加えて，RA として様々なことを教えてくださった金子正弘さん，コードについて様々な知識を教えてくださった小平知範さん，困った時に相談に乗っていただいた研究室の同期・先輩の方々に深く感謝の意を表して謝辞とさせていただきます.

# 参考文献

[1] Graham Neubig and Shinsuke Mori. Word-based partial annotation for efficient corpus construction. *LREC*, 2010.

[2] Hisami Suzuki and Kristina Toutanova. Learning to Predict Case Markers in Japanese. In *ACL*, 2006.

[3] Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. Recurrent Neural Network Regularization. ICLR, 2015.

[4] 奥雅博, 松岡浩司. 文字連鎖を用いた複合語同音異義語誤りの検出手法とその評価. 自然言語処理, Vol. 4, No. 3, pp. 83–99, 1997.

[5] 梶谷貴士, 服部峻. 文章校正における共起語を用いた漢字の誤変換の検出 (情報ネットワーク). 電子情報通信学会技術研究報告 = IEICE technical report : 信学技報, Vol. 115, No. 405, pp. 19–22, 2016.

[6] 工藤拓, 小松弘幸, 花岡俊行, 向井淳, 田畑悠介. 統計的かな漢字変換システム Mozc. 言語処理学会 第 17 回年次大会, pp. 948–951, 2011.

[7] 今村賢治, 齋藤邦子, 貞光九月, 西川仁. 小規模誤りデータからの日本語学習者作文の助詞誤り訂正. 言語処理学会論文誌, Vol. 19, No. 5, pp. 381–400, 2012.

[8] 林秀治, 山本和英. 漏れのない漢字変換誤り検出と誤り可能性によるレベル分け. 言語処理学会 第 22 回年次大会, pp. 1145–1148, 2016.

[9] 新納浩幸. 平仮名 n-gram による平仮名文字列の誤り検出とその修正. 情報処理学会論文誌, Vol. 40, No. 6, pp. 2690–2698, 1999.

[10] 水本智也, 小町守, 永田昌明, 松本裕治. 日本語学習者の作文自動誤り訂正のための語学学習 SNS の添削ログからの知識獲得. 人工知能学会論文誌, Vol. 28, No. 5, pp. 420–432, 2013.

[11] 南保亮太, 乙武北斗, 荒木健治. 文節内の特徴を用いた日本語助詞誤りの自動検出・校正. 情報処理学会研究報告自然言語処理（NL）, Vol. 2007, No. 94, pp. 107–112, 2007.