E-049

変更履歴に着目した機械学習による論文校正支援方法の考察

Support for Proofreading Papers Based on Machine Learning Using Change Histories

松井 亮介[1)] 渡辺 開斗[2)] 蜂巣 吉成[1)] 吉田 敦[1)] 桑原 寛明[1)]
Ryosuke Matsui Kaito Watanabe Yoshinari Hachisu Atsushi Yoshida Hiroaki Kuwabara

1 はじめに

学生の卒業研究や学会での論文投稿の際には，教員による論文校正が行われる．この論文校正作業では，学生が論文を執筆して教員へ提出し，教員は添削を行い学生へ論文を返却する．論文を完成させるまでこの作業を何度も繰り返す．添削観点は内容と表現の 2 つがある．内容添削とは論文の主張や論証といった意味的な内容に対する指摘を指し，表現添削とは誤字脱字や論文に相応しくない表現の指摘といった文章表現についての添削を指す．

教員が添削する論文の数が増えると，添削作業量が多くなることがある．文章表現の指摘といった複数の論文に対して同じような修正指示をする作業が多くなると，教員は論文の本質的な内容に対する指摘に時間をかけることができないという問題点がある．

既存の添削支援方法として，2 章で示すようなルールベースの文章校正方法がある．ルールベースの場合，校正ルール構築と適宜更新する必要がある．研究分野や教員によって適切とされる表現には違いがある．複数に渡る研究分野や教員に対応でき，それぞれに最適化されたルールを構築するには，各個のルールを定めることは避けられず，実現には膨大な作業量となる．

本研究では，変更履歴に着目した論文の校正支援方法について考察する．具体的には単語についての校正ルールを自動的に構築しながら，論文の中から添削が必要な文を発見する方法を検討し，有用性について考察する．対象言語は日本語である．添削対象文の発見を実現するために，論文の変更履歴を利用してデータを収集し，機械学習によって未知の文の修正必要性を判定する．

変更履歴に着目した理由は，変更前が誤りを含む表現，変更後を誤りが修正された文とすれば，機械学習によって，人がルールを記述せずともルールを構築できると考えたことにある．構築されたルールは変更履歴を増やすほど充実するので，校正の精度もデータの数に応じて向上することが見込める．特定分野の論文や特定の教員が添削した論文の変更履歴を用いれば，それぞれの研究分野や教員ごとに特化したルールの構築もできる．

変化の種類に基づいて，修正が必要な文の判定を実現するのに必要な研究課題を次のように設定した．

課題 1:添削前後の文の対応関係と，変化の種類の特定方法
課題 2:単語レベルで表現を校正する方法
課題 3:文レベルで誤り文を判定する方法

添削前後の文章間では，文の位置が移動している場合や文が分割したり統合する場合，新しい文の追加や，元々あった文の削除など様々な変化が起こっている．校正にあたって添削前後の文を対応づけて置換，分割，統合などの変化の種類を特定し，添削に利用できる文を抽出する．

単語レベルでの表現を校正する方法は，文脈に依らず規則的に変化する表現があると考え，それらを評価するために設定した．これは課題 1 の変化の種類が特定できれば実現可能である．

文レベルでの誤りを判定する方法については，単語レベルではわかりにくいものをはじめとした，明記が難しいルールを構築する際，機械学習を活用できるか検討する課題として設定した．

本論文では文単位での変化の種類の特定，形態素の比較による単語レベルでの評価，機械学習による文レベルの評価の 3 プロセスによって文の修正必要性を示す方法を提案する．提案方法の有効性を示すために各プロセスに対応した実験を行い，結果について考察する．

2 既存ツール・関連研究

textlint[1] はルールベースの文章校正方法で，手動でルールを作成し，機械的に文を評価する．あらかじめ複数のルールが構築されたルールプリセットが用意されているが，新しいルールの作成には利用者が記述する．学問分野や教員ごとに適切な表現は異なるので，論文校正を目的としたルールベース校正手法においてはルール作成と更新作業は必要となる．本研究では，変更履歴を活用してルール作成と更新作業の自動化を図った．

山田ら [2] は文書作成履歴に着目し，過去に作成した文をすべて正しい校正知識として蓄積する．本研究での修正対象が文章表現であるのに対し，修正対象は文章の表層の誤りである．ここでの表層の誤りとは，文字や単語の誤入力，カタカナの表記ゆれなどを指す．入力された文章と過去の類似した文章を比較し，差分を取ることで表層の誤り検出を実現する．ルールとなる校正知識は過去の論文との違いに依存するので，ルールを定義することは想定されない．

3 変更履歴に着目した論文校正支援方法

本研究の最終目標は，添削対象となる文の検出，および検出ルール構築の自動化である．ここでの添削対象として，論文として適切でない表現を想定する．提案方法については設計時点での有用性がわからないことから，基礎的な発想による方法を考案し，実験によってその有用性の確認を行う．

文の修正必要性の判定方法として次の 3 つを定めた．

1. 文の変化の種類の特定
2. 形態素の比較による単語レベルでの評価
3. 機械学習による文レベルでの評価

文における変化の種類の特定では，文の類似度を定義し添削前後の文での対応づけを行う．このプロセスで，

1） 南山大学 Nanzan University
2） 株式会社インディーズゼロ

ルール構築に必要となる添削前後の文の対応関係と変化の種類を明らかにし，添削情報として使用可能なデータを確保する．

形態素の比較による単語レベルでの評価では，変更履歴から変化している単語のファイルを作成することで，単語表現の添削情報を確保する．

機械学習による文レベルでの評価では，文法表現や品詞間の関係性に着目して，機械学習を用いて文の修正必要性を判定する．添削前を誤り，添削後を正しい文として学習させ，文を正しい/誤りに分類する問題として文検出を行う．

文に着目する理由として，「適用」や「適応」が相応しいかどうか判定する場合のように，どんな情報から誤用を判定するかわからないとき，文ならばその文全体の文脈情報まで扱えるからである．段落程度まで単位を広げると，誤りを指摘をしてもどこが誤りかわからない．したがって文が最も妥当だと考えた．

3.1 文の変化の種類の特定

3.1.1 変化の種類の定義

文の変化の種類を次のように定義する．

置換 文中の一部分または複数の部分的な変化
分割 添削前に存在する 1 つの文が，添削後で複数の文に分かれている変化
統合 添削前に存在する 2 つの文が，添削後で 1 つの文にまとめられている変化
追加 添削前の文章内に存在しない文が添削後に新たに加わっている変化
削除 添削前に存在している文が添削後に消去されている変化

論文の添削前後の文には，これらの変化と変化なしの文がある．このうち，追加と削除の変化に関しては内容的な観点で起こっていると考えられる．表現の観点における変化としては，置換，分割，統合の 3 つの変化が該当する．

3.1.2 LCS アルゴリズムを用いた類似度算出

文の変化の対応関係特定のために，添削前後の文を総当たりして，すべてのペアに対して類似度を計算する．本研究では，類似度を計算するにあたって，添削前後での共通部分の長さを求められる，LCS[1)] アルゴリズムを用いる．

LCS とは，2 つの列のうち順序を保ったまま共通する最長の部分列であり，部分列は連続でなくてもよいので，置換された文の対応を取りやすい．文の各ペアの LCS の長さを求め，添削前後の各文の長さで割ることで，2 つの文字列の共通部分の割合を求められる．本研究ではこの値を 2 文の類似度とする．類似度は文の形態素に対しての LCS から算出する．形態素数と文字数では形態素単位の方が比較する要素が減り，形態素解析を行う時間を含めても形態素単位で類似度算出した方が実行時間が早いと判断したからである．

類似度を求める際に，添削前の文の形態素数で割った値と，添削後の文の形態素数で割った値の 2 つの数値が得られる．便宜上，図 1 のように前者を $ratioA$，後者を $ratioB$ とする．

1） Longest Common Subsequence：最長共通部分列

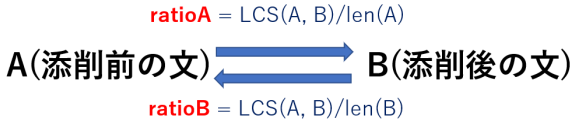

図 1 類似度の表記

算出した類似度に閾値などの条件を設け，添削後の文が添削前に対して置換されているものを検出する．

3.1.3 類似度から変化の種類を判定

類似度に閾値を設定し，3 段階の判定によって置換と分割，統合を判定する．1 回目の判定は閾値条件を厳しく設定し，適合率が高い，つまり，正しい可能性の高い置換のペアを確保する．2 回目の判定では，1 回目の結果を判定対象から除き閾値条件を緩めて判定し，1 回目の判定で対応付かなかった文を判定し，再現率が高くなるようにする．条件を緩めたことで 1 文あたり置換の候補が複数出現することがある．それらに対し 3 回目の判定として，添削前後の対応関係の個数に応じて分割統合を判定する．以降，3 回の判定を 1 次判定，2 次判定，3 次判定と名付ける．それぞれの判定における条件および理由を次に示す．

1 次判定

(1)$ratioA \geq t_1$ かつ $ratioB \geq t_1$ (t_1 : 1 次判定の閾値)
(2)$ratioA \neq 1.0$ または $ratioB \neq 1.0$
(3)添削前後の文の位置関係が前後 10 文以内

1 次判定は，誤判定の少ない置換判定のペアの確保が目的である．条件 (1) で採用する閾値 t_1 の値は，誤判定を減らしたいので適合率が最も高くなる値が望ましい．ここでの適合率とは 4.1.2 項のものを指す．本研究で定める t_1 の値は実験を行う 4 章で説明する．条件 (2) は $ratioA$，$ratioB$ 共に 1.0 の場合は，完全一致であるので変化なしとして扱い，置換判定からは除外する．条件 (3) は位置制限を加えるために設定した．これは前提として，添削前後で対応関係がある文は論文中の同じような位置に出現し大きく移動はしないと考えたからである．誤判定の回避と同時に，判定対象となる文の削減による実行時間の短縮を図る．

2 次判定

(4)$ratioA \geq t_2$ かつ $ratioB \geq t_2$ (t_2 : 2 次判定の閾値)
(5)$ratioA \neq 1.0$ または $ratioB \neq 1.0$
(6)1 次判定で置換と判定された文ではない

2 次判定では，1 次判定で判定できない置換のペアを確保することが目的である．条件 (4) の閾値 t_2 に採用する値は，1 次判定より多くの判定を得たいので，$t_1 > t_2$ を満たし，再現率が高くなることが望ましい．ここでの再現率とは 4.1.2 項のものを指す．本研究で定める値は実験を行う 4 章で説明する．条件 (5) は 1 次判定と同様である．1 次判定では離れた文同士の対応関係は考慮しなかったが，より多くの置換判定を出すためにその条件をここでは除外した．条件 (6) によってすでに置換判定された文に対する誤判定を防ぐことができる．

3 次判定

3 次判定では，2 次判定までで判定した置換の中から，分割と統合判定のペアを確保することを目的とする．置

換判定されたペアの個数から 1 対多の対応を取っているものを分割，多対 1 の対応を取っているものを統合と判別する．多対多の対応を取るものについては，判別不能として扱う．3 次判定が終了した時点で，対応関係がないと判定されたものは追加または削除された文であると判定する．

3.2 形態素の比較による単語レベルでの評価

論文の文変化には，単語の変化のように規則的に変化する表現がある．そのような変化については，機械学習などを用いずともルール構築は可能である．形態素の比較では，単語単位で機械的に変化するような表現を対象に，変更履歴から添削情報を確保する．

方法として 3.1 節で置換と判定された文のペアから，良い単語/単語群と悪い単語/単語群を特定する．例えば，「事」という部分が「こと」に変化した場合，「事」を悪い単語，「こと」を良い単語とする．このような，正しい単語/単語群や誤り単語/単語群をデータとして蓄積することで，未知の文章に対して単語レベルの校正を可能にする．

本研究においてここで期待される結果は，添削されがちな単語レベルの間違いデータと修正案を保存して添削ルールを構築することである．

検出対象の変化は，次の 2 つである．

1. 1 つの独立した形態素が置換されている
2. 2 つ以上の連続した形態素が置換されている

単語レベルで変化を特定するには，添削前後の単語同士の対応が明らかである必要がある．そこで置換判定されたペアの差分を形態素レベルで取る．差分を取ることで，単語同士の対応を取りつつ，変化している部分を抽出できる．差分を取った結果から，置換されている形態素/形態素群を，添削前のものは誤り，添削後のものは正解として添削ルールを保存する．

3.3 機械学習による文レベルでの評価

3.3.1 方法概略

単語レベルの変化の評価では形態素の表層の形に着目したが，その中でも接続詞や助詞などは文脈によって正しいかが決まるので，表層のみを用いる単語レベルでの指摘は難しい．そこで品詞や活用形を用いて，機械学習で文の評価を実現する．

本研究で期待する結果は，文の修正が必要か不必要を判定できることである．そのために機械学習における教師あり学習によって，文を正しい/誤りの 2 クラスと対応した 2 値に分類する．ここで対象とする変化は，添削ルールとして記述しづらい表現である．ここでは前提として，添削ルールとして人間が想定できない部分の誤りやルールとして明記できない誤りがあると仮定し，そうした文の誤りを対象に含んで機械学習での分類を行う．

データは文単位での変化の種類の特定において置換と判定されたペアから作成する．その際，添削前の文を誤り，添削後の文を正しいものとみなしてタグ付けする．タグは誤りを 0，正しいものを 1 と対応づける．作成したデータを訓練データとして，添削ルールの学習を行う．

文をモデルに入力するために，前処理で文のベクトルを作成する．文のハッシュ化にあたって，形態素の表層だけでは正しいか判断しにくい表現に対して，傾向や特徴が現れると考え，表層，品詞，活用形の情報を用いる．

作成したベクトルを機械学習モデルを用いた分類器に入力する．機械学習モデルには，形態素ごとの特徴を畳み込める CNN[2] と形態素の系列情報を扱える RNN[3] の 2 つを用いる．

3.3.2 前処理方法

文をハッシュ化する前処理方法として，表層，品詞，活用形といった形態素情報を各種類ごとに設定した固有 ID にする処理を行う．品詞や活用形に着目した理由は，前項で述べたとおりである．品詞情報を用いることで助詞や接続詞といった表現に対しての特徴の出現を期待する．活用形を用いることで，能動態や受け身など表層の形を参照するだけでは正しいか判断しにくい表現の傾向や特徴が現れることを期待する．

単語の表層の形については，過去の論文から実行の度に単語辞書を作成，更新し動的に単語の固有 ID を付与する．データを増やした場合に未知の単語の出現に対応できるようにした．品詞情報と活用形情報については，形態素解析器の辞書に保存されている種類に ID を振る．今回は MeCab[3] の IPA 辞書を利用した．

各形態素についての表層，品詞，活用形を ID 表現にした後，それらを表層，品詞，活用形の順に 1 列に並べることでベクトルを作成する．図 2 がベクトル作成例である．

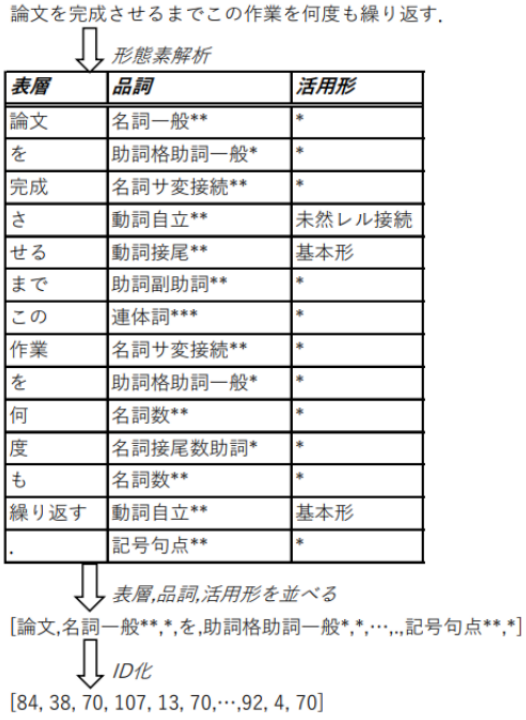

図 2 文のベクトル作成例

例として，図 2 の「論文を完成させるまでこの作業を何度も繰り返す.」という文に対して作成されるベクトルは，[’ 論文’, 名詞一般**. ’ を’, 助詞格助詞一般*, ...

2） Convolutional Neural Network：畳み込みニューラルネットワーク
3） Recurrent Neural Network：再帰型ニューラルネットワーク

'．', 記号句点**] となる．これらを対応する ID にラベリングする．

表層，品詞，活用形のうち，活用形情報は動詞のみが持つ情報であり，活用形情報が多く出現しない文もある．出現数が少なく学習できないことによって学習成果に影響が出ることを考慮して，4 章では活用形情報を使わず，表層と品詞情報を用いてベクトルを作成した場合の実験も行う．活用形を使用しない場合，表層，品詞，活用形の並びにはせず，表層，品詞のみを形態素ごとに並べてベクトルにする．

ベクトルを分類器に入力する際には，添削前後で品詞情報が変化しない場合は学習の阻害になることから，添削前後を比較して品詞情報が一致する文を取り除く．

3.3.3 学習モデルの設計

実験のために CNN 分類器と RNN 分類器の 2 つを用意した．

CNN について

CNN では，入力したベクトルを embedding 層で各 ID の分散表現にエンコードする．それぞれの分散表現について，表層，品詞，活用形の 3 つをフィルターによって畳み込む．畳み込んで得られたフィルタにマックスプーリング処理を行う．次に 2 つずつ要素の畳み込みとプーリングを行い，出力層を全結合する．形態素情報を畳み込みつつプーリングによる情報の損失を避けたいと考えて 2 回の畳み込みを行った．モデル概略を図 3 に示す．

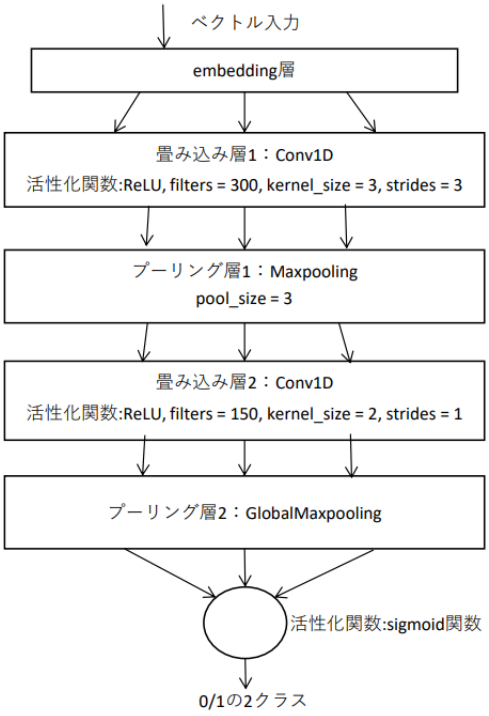

図 3　CNN モデルについて

損失関数は，2 値分類問題を行うことから binary cross entropy を採用した．評価関数には，分類の精度を測定したいことから accuracy を採用した．最適化アルゴリズムは，実験では示さないが最も学習成果が良いことが確認された adam を採用した．

4 章の実験では活用形情報を用いる場合と用いない場合についてそれぞれ実験を行う．その際，CNN では畳み込み層 1 での処理が変わる．ここでは，活用形情報を用いる場合のために畳み込み層 1 での kernel_size と strides を 3 としたが，活用形情報を用いない場合は，kernel_size と strides は 2 とする．これは，入力するベクトルに活用形に該当する要素がなくなるので各形態素で畳み込む情報が表層と品詞の 2 つになったことからこの数字を設定したモデルで分類実験をする．

RNN について

RNN では，入力したベクトルを embedding 層で各 ID の分散表現にエンコードする．それぞれの分散表現を RNN 層に入力する．RNN 層からの出力を全結合で出力層に連結する．モデル概略を図 4 に示す．

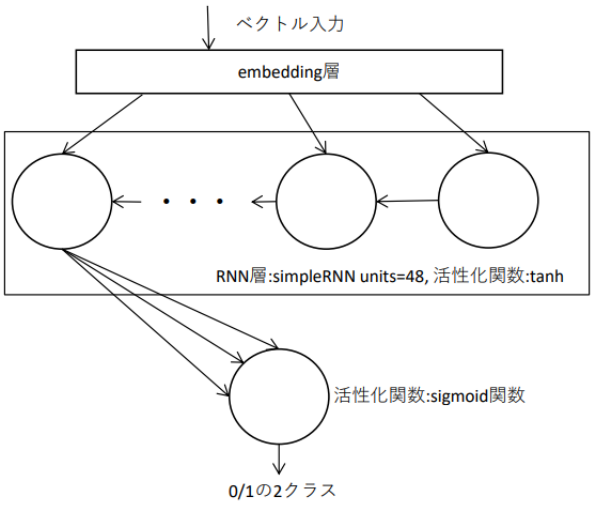

図 4　RNN モデルについて

RNN では，CNN 同様に入力したベクトルを embedding 層で各 ID の分散表現にエンコードする．そこで得られた単語分散表現を RNN 層に入力する．ここでの活性化関数は RNN モデルに一般的に用いられる tanh 関数を利用する．RNN からの出力ユニット数は 48 とした．これは実験で使用した各文の形態素数の最大値が 48 であったことから，各形態素の情報を各ユニットに学習させるのに十分な数として設定した．RNN の出力を全結合層に連結し，2 値に分類する．

損失関数は，CNN と同様に 2 値分類問題を行うことから binary cross entropy を採用し，精度を測定するために accuracy を採用した．最適化アルゴリズムは最もスコアがよいことが確認できた adam を採用した．

4　実験

4.1　実験 1：文単位での変化の種類の特定

実験 1 では実験 1-1 と実験 1-2 の 2 つの実験を行う．

4.1.1 実験目的・方法

実験 1-1

目的は，閾値あたりの分類精度を測定すること，および各論文ごとの 1 次，2 次判定での閾値を設定することである．

方法は，1 次判定での条件下で閾値を 0.4 から 0.8 まで 0.1 刻みで置換分類精度を測定する．閾値ごとの判定分類精度の比較から，論文ごとの 1 次判定閾値 t_1，2 次判定閾値 t_2 の値を定める．

実験 1-2

目的は，実験 1-1 で決定した閾値での分類精度を測定することである．加えて実験 1-1，1-2 の結果から 3.1.3 項での t_1，t_2 として採用する値を設定する．

方法は，実験 1-1 によって定めた閾値で 3 次判定まで行い，置換判定精度を測定する．実験 1-1 の精度と比較し，精度の変化を確認する．

実験 1-2 で算出した後，サンプルとして用意した論文と同様の筆者が執筆した論文で置換判定を行った．判定にあたっての条件は実験 1-2 と同様である．判定結果が置換されたものとして扱って良いかを目視で確認する．

4.1.2 指標算出

置換であるペアを置換と判定できたものを True Positive(以下 TP) とし，置換でないペアを無視できたものを True Negative(以下 TN) とする．置換でないペアを置換と判定したものを False Negative(以下 FN)，置換であるペアを無視したものを False Positive(以下 FP) とする．

サンプルデータから手作業で作成した置換正解データを参照して正しく置換されたかを判定する．

このプロセスでの分類精度を測定するにあたって，再現率，適合率，F 値の指標を用いる．

再現率 $(Recall) = \frac{TP}{TP+FN}$

適合率 $(Precision) = \frac{TP}{TP+FP}$

F 値 $(F-measure) = \frac{2\times Precision\times Recall}{Precision+Recall}$

テストデータとして我々の研究室の 2020 年度卒業研究の中間発表予稿 A，B，C，D を用意した．中間発表予稿とは，卒業研究の中間発表用原稿であり，研究概要を 4 枚にまとめたものである．添削は全て同じ教員が行なっている．これらを実験対象として用いる．

4.1.3 実験結果

実験 1-1　実験 1-1 の結果を示す．表 1，2，3，4 はそれぞれ論文 A，B，C，D の結果である．

表 1　論文 A における閾値ごとの置換判定精度

閾値	判定数	正解数	正答数	再現率	適合率	F 値
0.4	171	166	116	0.70	0.68	0.69
0.5	170	166	127	0.77	0.75	0.76
0.6	157	166	120	0.72	0.76	0.74
0.7	147	166	109	0.66	0.74	0.70
0.8	124	166	86	0.52	0.69	0.59

表 2　論文 B における閾値ごとの置換判定精度

閾値	判定数	正解数	正答数	再現率	適合率	F 値
0.4	81	97	40	0.41	0.49	0.45
0.5	74	97	48	0.49	0.65	0.56
0.6	62	97	44	0.45	0.71	0.55
0.7	46	97	35	0.36	0.76	0.49
0.8	37	97	28	0.29	0.76	0.42

表 3　論文 C における閾値ごとの置換判定精度

閾値	判定数	正解数	正答数	再現率	適合率	F 値
0.4	70	110	41	0.37	0.59	0.45
0.5	68	110	52	0.47	0.76	0.58
0.6	65	110	55	0.50	0.85	0.63
0.7	55	110	49	0.45	0.89	0.60
0.8	46	110	40	0.36	0.87	0.51

表 4　論文 D における閾値ごとの置換判定精度

閾値	判定数	正解数	正答数	再現率	適合率	F 値
0.4	83	154	61	0.40	0.73	0.52
0.5	86	154	72	0.47	0.84	0.60
0.6	78	154	72	0.49	0.96	0.65
0.7	72	154	70	0.45	0.97	0.61
0.8	64	154	62	0.40	0.97	0.57

これらの結果から，各論文における閾値 t_1 と t_2 の値を設定する．t_1 は適合率が高くなる値を設定し，t_2 は再現率が高くなる値を設定する．よって論文 A，B，C，D における閾値は次のように設定した．

論文 A　$t_1 = 0.6,\ t_2 = 0.5$
論文 B　$t_1 = 0.7,\ t_2 = 0.5$
論文 C　$t_1 = 0.7,\ t_2 = 0.6$
論文 D　$t_1 = 0.7,\ t_2 = 0.6$

実験 1-2　次に，実験 1-2 の結果を表 5 に示す．

表 5　3 次判定終了時の置換判定結果

論文	判定数	正解数	正答数	再現率	適合率	F 値
論文 A	183	166	135	0.81	0.74	0.77
論文 B	46	97	35	0.36	0.76	0.49
論文 C	55	110	49	0.45	0.89	0.60
論文 D	72	154	70	0.45	0.97	0.61

実験 1-1 および表 5 の結果より，本研究における変化の種類の特定で採用する閾値として，$t_1 = 0.7$，$t_2 = 0.6$ が良いと考えた．今回の 4 つのサンプルでは，$t_1 = 0.7$ の場合に最も高い再現率を示した論文が多く，$t_2 = 0.6$ の場合に最も高い適合率を示した論文が多かったことが理由である．

この閾値を設定して，サンプルと同じ著者らが執筆した論文を分類した．置換判定として得られたペアは計 228 個であり，そのうち 221 個において置換判定として問題ないと確認できた．

4.2 実験 2

4.2.1 実験目的・方法

実験目的は，変化した形態素と出現回数を添削ルールとして保存することである．

実験方法は，提案した方法を実際に実行して，得られる辞書の単語と出現回数の確認を行う．この実験では，我々の卒業研究で作成した中間発表予稿の文を用いて実験する．

4.2.2 実験結果

表 6 は，実験 2 の結果の一部である．本研究の方法によって表層の単語の変化を辞書に保存できた．1 回のみ

表 6　実験 2 の結果

添削前	添削後	出現回数
文章	文	32
判断	判定	3
チェックする	調べる	3
本項	本稿	3
することで	して	2
チェック	評価	2
に着目	を対象と	1
したい	する	1

出現の変化が多く保存されたが，それらはすべて一般的な変化とは呼べない変化であったことから，出現回数でフィルタリングを加える必要がある．例えば「文章」が「文」への変化は表現変化であるが，一般的に他の論文でも出現するとまでは言えない．そうした変化については，論文ごとに作成したデータをすべて統合したときに，出現回数は相対的に少なくなると考えられる．そこで何回の出現があったらルールとして適用するかを検証する必要がある．

4.3　実験 3

実験 3 では 3.3.3 項で説明したモデルを用いて文の分類実験を行う．実験では，次に示す文集合 A，文集合 B，文集合 C を用いる．

文集合 A： 人為的に作成した文である．これは「適用」と「適応」が誤用されている文といった特定の誤り表現を含んだ文とそれが添削された文の集合である．文集合 A で用意した表現は表 7 の通りである．

表 7　文集合 A の表現一覧

パターン	番号	対象とする表現
1	1	することができる
	2	ため
	3	体現止め
	4	し
	5	たり
	6	適用，適応
	7	主語述語非対応
	8	1-1 から 1-7 まですべての文
	9	1-8 のうち誤りが 1 箇所の文
2	1	パターン 1 の表現が 1 文に 2 箇所以上現れる文
3	1	1-9 と 2-1 のすべての文
	2	パターン 1,2 すべての文

表現ごとの学習精度を測定するために 1 文あたりの添削箇所が 1 種類の文をパターン 1，添削箇所が 2 種類の文をパターン 2，パターン 1，2 の集合をパターン 3 として区別している．

そのうち，1-8 の文は 1-1 から 1-7 までの文をまとめたものである．1-8 では，誤り文の 1 文中に複数の添削対象表現が出現したとき，どれか 1 つを添削している．

1-9 の文は，1-8 の誤り文を添削対象箇所が 1 種類になるように修正した文である．例として「〜のため，... することができる」のように 1 文に複数の添削対象箇所を含む文があったとき「〜のため，... できる」と修正して 1 文あたり 1 種類の変化となるようにする．これは検証の上で，添削されずに残る表現の存在は学習の阻害になると考えて行う．

文集合 B： 我々の研究室の卒業研究の中間発表予稿の論文に対し，本研究の置換判定方法を適用して得られた置換判定結果で作成した文集合である．

文集合 C： 我々の研究室の卒業研究の最終予稿の論文に対し，本研究の置換判定方法を適用して得られた置換判定結果で作成した文集合である．

中間発表予稿とは 4.1.2 項にて説明したものである．最終予稿とは，卒業研究の本稿の内容についての概要を 4 枚にまとめたものである．ここで用意したサンプルはすべて同一の 1 人の教員から添削を受けたものである．

4.3.1　実験目的・方法

実験 3-1

実験目的は，文の学習自体そのものが可能か検証すること，およびどういった表現が学習できるか検証する．

方法として，人為的に作成した文集合 A を用いて分類を行い，その判定精度を測定する．それぞれのパターンごとに個別で分類実験を行う．各パターンの文において，無作為に選択した 70%を訓練用に，30%をテスト用に用いる．

実験 3-2

実験目的は，人為的に加工されていない実際の論文を用いて文の分類は可能か検証することである．

方法として，文集合 B を用いて分類を行い，判定精度を測定する．文集合 B のうち無作為に選択した 70%を訓練用に，30%をテスト用に用いる．

実験 3-3

実験目的は，ルール学習用データを用意して学習したとき，実際の論文で分類ができるか検証することである．

方法として，文集合 A を訓練用に用いて学習させ，文集合 B をテスト用に用いて分類し判定精度を測定する．

実験 3-4

実験目的は，時系列的に過去の論文で学習して，文が分類できるか検証することである．

方法として，文集合 B を訓練用に用いて学習させ，文集合 C をテスト用に用いて分類し，判定精度を測定する．

表層，品詞のみでハッシュ化したものをハッシュ方法 1，それに活用形を追加したものをハッシュ方法 2 とする．

それぞれの実験で CNN と RNN を用いて検証する．CNN，RNN ともにバッチサイズを 16，エポック数を 20 と設定して分類を行った．バッチサイズを 16 に設定した理由は，実験でのデータ数が少なく，バッチサイズ 32 以上での学習ができなかったからである．エポック数は暫定的に設定している．

4.3.2　評価方法

このプロセスでの分類の精度を評価するにあたって，正答率，再現率，適合率，F 値の指標を用いる．誤り文を誤りであると予測できたものを TP とし，正しい文を正しいと予測したものを TN とする．したがって，誤り文を正しいと予測したものを FN，正しい文を誤りと予測したものを以下 FP として扱う．

正答率の指標は，次の数式で算出する．再現率，適合率，F 値の指標は，4.1.2 項のものと同様である．

$$\text{正答率 } (Accuracy) = \frac{TP+TN}{TP+FP+FN+TN}$$

ここでの指標の意味として，正答率は予測したものがどれだけ答えと一致したかの割合となり，再現率は誤り文のうち誤りと予測ができたものの割合となる．適合率は，誤りと予測したもののうち実際に誤り文であるものの割合となる．F 値はそれらから算出される．

本研究では，暫定的に正答率 0.7 を目標として設定し，それを超えた場合学習できたと判断する．

4.3.3 実験結果

以降の実験結果に示す表では，表示の都合上，訓練データ数とテストデータ数を *train* と *test* と表現する．

実験 3-1 の結果

実験 3-1 の結果を次の表に示す．CNN は表 8 が活用形を含むハッシュ方法 1 の前処理であり，表 9 が活用形を含まないハッシュ方法 2 で処理したものである．

表 8　ハッシュ方法 1 による CNN での実験 3-1 の結果

番号	*train*	*test*	正答率	再現率	適合率	F 値
1-1	51	23	0.74	0.92	0.69	0.79
1-2	74	33	0.85	0.94	0.79	0.86
1-3	109	47	0.96	1.00	0.93	0.96
1-4	225	97	0.72	0.37	0.5	0.42
1-5	42	19	0.21	0.14	0.1	0.12
1-6	124	54	0.19	0.19	0.19	0.19
1-7	54	24	0.38	0.29	0.44	0.35
1-8	686	294	0.51	0.32	0.37	0.34
1-9	684	294	0.52	0.46	0.47	0.46
2-1	252	109	0.59	0.46	0.59	0.52
3-1	937	402	0.65	0.55	0.64	0.59
3-2	1623	696	0.79	0.73	0.80	0.76

表 9　ハッシュ方法 2 による CNN での実験 3-1 の結果

番号	*train*	*test*	正答率	再現率	適合率	F 値
1-1	51	23	1.00	1.00	1.00	1.00
1-2	74	33	0.94	1.00	0.89	0.94
1-3	109	47	0.96	0.96	0.96	0.96
1-4	225	97	0.81	0.48	0.88	0.63
1-5	42	19	0.42	0.10	0.33	0.15
1-6	124	54	0.54	0.48	0.58	0.53
1-7	54	24	0.42	0.50	0.43	0.46
1-8	686	294	0.63	0.69	0.55	0.61
1-9	684	294	0.62	0.60	0.54	0.57
2-1	252	109	0.71	0.68	0.73	0.70
3-1	937	402	0.77	0.73	0.71	0.72
3-2	1623	696	0.91	0.88	0.90	0.89

RNN での実験 3-1 の結果を表 10，表 11 示す．

CNN と RNN の比較においては，CNN の方が結果が良いことがわかる．CNN ではハッシュ方法 2 の結果が精度が高いことが多い．CNN のハッシュ方法 2 に着目すると，1-6 から 1-9 のパターンにおいて正答率が 0.7 以下となり，学習できなかったと言える．

実験 3-2 の結果

実験 3-2 の結果を次の表 12 に示す．ここでは，CNN と RNN におけるハッシュ方法 1，2 の結果を示す．訓練データ数は 792 文であり，テストデータ数は 340 文であった．

表 10　ハッシュ方法 1 による RNN での実験 3-1 の結果

番号	*train*	*test*	正答率	再現率	適合率	F 値
1-1	51	23	0.91	0.83	1.00	0.91
1-1	74	33	0.21	0.20	0.18	0.19
1-3	109	47	0.98	1.00	0.96	0.98
1-4	225	97	0.57	0.04	0.05	0.05
1-5	42	19	0.42	0.38	0.33	0.35
1-6	124	54	0.22	0.23	0.21	0.22
1-7	54	24	0.67	0.69	0.69	0.69
1-8	686	294	0.52	0.36	0.45	0.40
1-9	684	294	0.52	0.46	0.44	0.45
2-1	252	109	0.58	0.58	0.54	0.56
3-1	937	402	0.59	0.48	0.51	0.50
3-2	1623	696	0.69	0.61	0.66	0.64

表 11　ハッシュ方法 2 による RNN での実験 3-1 の結果

番号	*train*	*test*	正答率	再現率	適合率	F 値
1-1	51	23	1.00	1.00	1.00	1.00
1-2	4	33	0.27	0.33	0.26	0.29
1-3	109	47	0.98	1.00	0.96	0.98
1-4	225	97	0.67	0.48	0.45	0.47
1-5	42	19	0.53	0.40	0.57	0.47
1-6	124	54	0.20	0.32	0.20	0.25
1-7	54	24	0.83	0.93	0.81	0.87
1-8	686	294	0.56	0.46	0.48	0.47
1-9	684	294	0.57	0.50	0.48	0.49
2-1	252	109	0.58	0.63	0.53	0.57
3-1	937	402	0.68	0.64	0.61	0.62
3-2	1623	696	0.86	0.84	0.81	0.83

表 12　実験 3-2 の結果

モデル	ハッシュ	正答率	再現率	適合率	F 値
CNN	1	0.33	0.33	0.31	0.32
CNN	2	0.34	0.34	0.33	0.33
RNN	1	0.44	0.44	0.43	0.44
RNN	2	0.32	0.39	0.33	0.35

CNN と RNN 共に正答率は 0.5 を超えることができず，学習ができなかったと言える．

実験 3-3 の結果

実験 3-3 の結果を次の表 13 に示す．CNN と RNN におけるハッシュ方法 1，2 の結果を示す．訓練データ数は 2319 文であり，テストデータ数は 1132 文であった．

表 13　実験 3-3 の結果

モデル	ハッシュ	正答率	再現率	適合率	F 値
CNN	1	0.52	0.25	0.53	0.34
CNN	2	0.51	0.19	0.53	0.28
RNN	1	0.49	0.39	0.49	0.43
RNN	2	0.50	0.28	0.50	0.36

CNN と RNN 共に正答率が 0.5 程度であり，学習できなかったと言える．

実験 3-4 の結果

実験 3-4 の結果を次の表 14 に示す．CNN と RNN におけるハッシュ 1，2 の結果を示す．訓練データ数は 1132 文であり，テストデータ数は 456 文であった．

表 14　実験 3-4 の結果

モデル	ハッシュ	正答率	再現率	適合率	F 値
CNN	1	0.52	0.61	0.52	0.56
CNN	2	0.53	0.42	0.54	0.47
RNN	1	0.53	0.41	0.54	0.47
RNN	2	0.51	0.93	0.51	0.65

CNN と RNN 共に正答率は 0.5 程度であり，学習したとは言えない．RNN のハッシュ化 2 で再現率が高い数字を示しているが，これはほぼ全ての文を誤りと指摘した結果である．

5　考察

実験結果からは，本研究の方法は有用だと言えるほどの結果が得られなかった．各プロセスについて今後の改善すべき点を考察する．

5.1　文単位の変化の種類の特定方法

本稿で提案した方法で添削前後での文の対応関係を特定することができたが，分割と統合については，サンプル内に十分な例がなかったので検証ができなかった．今後サンプルをさらに増やして検証をする必要がある．

5.2　形態素の比較による単語レベルでの評価

本稿の方法によって表層の単語の変化の内容と出現回数を保存することができた．1 回のみ出現の変化が多く保存されたが，それらはすべて一般的な変化とは呼べない変化であったことから，出現回数でフィルタリングを行うなど処理を加える必要がある．

5.3　機械学習による文の評価

今回行った実験では，誤り文を判定することができなかった．その理由として次のものがあると考えた．

- 教師データが少ない
- 前処理方法が十分でない
- 学習モデルが最適でない

自然言語処理において，機械学習に必要となるデータ数は数万件以上に及ぶことが多い．その点で，本研究では数千件程度のデータしか集められておらず，学習が十分に行われなかった可能性がある．

実際の論文において明確な規則性を持って変化している表現は少ない．その理由は，論文を執筆した学生が，添削指示を受けて適切な表現を学習することで添削が必要となる表現を使わなくなるからである．論文添削においては添削開始当初は内容に関しての修正が多くなりがちであることも表現変化が少なくなる原因と考えられる．

表現変化のデータが少ないことの影響は学習ができないことだけではない．検証用データの中に，学習させたルールで分類できる表現が少ないことにより，実験 3-3 のようにルール学習用の教師データを学習させても分類できない場合もある．今後も引き続きデータを集めることは必須であるが，教員や今後の学生に協力を仰いだり，添削途中段階が残っていない過去の論文の完成版を正しい表現の文としてデータに加えるといった工夫が必要である．

本研究の方法には改善の余地がある．前処理では 2 種類のハッシュ方法を採用し比較した．結果として実験 3-1，3-2 ともに表層，品詞を並べたものの方が良い結果を得られたが，これは文の形態素に活用形情報が多く出現しておらず，学習するだけの情報量がなかったからであると推測できる．学習のためには形態素に限らず，n-gram といった方法によるハッシュ化も検討すべきである．

学習モデルについては，embedding 層によって単語分散表現を獲得したが，ここで word2vec といった方法を用いれば分類精度に変化が現れると期待できる．実験 3 の結果を参照すると CNN が最もよい結果であったが，一部の表現では RNN の方が良いスコアを残している．分類器を組み合わせたアルゴリズムを試すことも検討したい．

本研究における実験では，指標算出にあたって作成した正答は手動で作成している．その点で作成者の主観が入ることが避けられず，妥当性の脅威を含んでいる．作成者の主観が入らないような方法で検証ができるように検証方法を見直すことが望ましい．

6　おわりに

本論文では変更履歴に着目した論文校正支援方法を提案し，実験によって有用性を検証した．論文から要修正となる文を発見して添削作業支援を行うことを目的とした．LCS アルゴリズムを用いて添削前後の対応関係を特定する．それをもとに，差分をとることで単語レベル，機械学習を用いて文レベルでの検出ルールを構築する．実際に判定を行い，その結果から文の検出精度を上げるための考察をした．

今後の課題として，分割や統合の変化をしているサンプルを充実させて検証することが必要である．また，機械学習での文の評価について，教師データをさらに増やすために，データセットの作成を行うことが挙げられる．

参考文献

[1] azu：textlint で日本語の文章をチェックする．
Web Scratch，入手先〈https://efcl.info/2015/09/10/introduce-textlint/〉(参照 2020-09-19).

[2] 山田洋志，竹元義美：文章作成履歴を利用した校正支援機能，
全国大会講演論文集，Vol.52，No. メディア情報処理，pp.281-282(1996).

[3] 工藤拓：MeCab:Yet Another Part-of-Speech and Morphological Analyzer，
入手先〈taku910.gihub.io/mecab/〉(参照 2020-09-19).