情報処理学会論文誌 Vol.64 No.4 871–876 (Apr. 2023)

テクニカルノート

タッチデバイスにおける文章訂正効率化手法の提案

佃 充宏[1] 宮尾 秀俊[1,a)] 丸山 稔[1,b)]

受付日 2022年11月28日, 採録日 2023年1月10日

概要：スマートフォンのような小型のタッチデバイス上における文章訂正では，訂正箇所の選択や選択箇所の消去に，反復的かつ緻密な操作が要求される．本研究では，漢字の誤変換を含む文章を対象にこれらの操作なしに文章訂正が可能なシステムを提案し，さらにその有効性について示す．英文を対象にした同様のシステムは提案されているが，文章構造の違いから，その手法を日本語に適用することはできない．そこで本研究では，ファインチューニングした自然言語モデル ALBERT を用いることで，この問題を解決した．また，ALBERT の学習過程で生じた学習データ不足に対してタイピング傾向空間モデルに基づくデータ増強を適用した．この効果についても示す．提案システムは，iPhone 上で動作するアプリとして実装し，文章訂正に掛かった時間と必要タップ回数の計測およびユーザの評価により，本手法の有効性を検証した．

キーワード：文章訂正，タッチデバイス，自然言語処理，データ増強

An Efficient Technique for Text Correction on Touch Devices

Mitsuhiro Tsukuda[1] Hidetoshi Miyao[1,a)] Minoru Maruyama[1,b)]

Received: November 28, 2022, Accepted: January 10, 2023

Abstract: Text correction on touch screen devices such as smartphones requires repetitive and precise operations in order to select and delete parts of text. In this paper, we present a technique that can correct text containing kanji conversion error without those operations and show the effectiveness of the technique. A similar technique for English text has been presented, but it cannot be applied to Japanese text due to differences in language structure. Therefore, our system sloves this problem by fine tuning the natural language model ALBERT. We also present a data augmentation technique with patial model applied to the lack of training data, and show its effectiveness. We implemented presented technique as an application running on the iPhone, and evaluated technique in text correction task.

Keywords: text error correction, touch screen device, natural language processing, data augmentation

1. はじめに

今日，スマートフォンに代表されるタッチデバイスが広く普及し，情報発信に活用されている．端末の小型化にともない携帯性が高まった一方で，タッチパネル上に表示されたボタン等の UI が指に対して小さいことで正確な入力が困難になる fat finger 問題 [1] が生じるようになった．この問題はタッチデバイス上での文字入力の際にも発生し，文章訂正では，狭い画面領域に表示された小さな文字列の合間にカーソルを移動する操作や，小さな消去キーを反復的に押す操作が要求されるため，頻繁に誤操作が発生する．この問題に対して様々な手法が提案されている．

Apple iOS キーボード [2] では，キーボード領域を長押しすることによりトラックパッドのようにカーソルを移動できる機能や，カーソルを長押しすると指で隠れた部分が吹き出しで拡大表示され，精密な選択ができる機能等がある．AndroidOS 搭載のキーボード [3] では，スペースバー上を左右にスワイプすることで，カーソルを移動できる機能がある．これらの手法では，より精密にカーソルを移動できるものの，依然としてカーソルを正確な位置に移動す

[1] 信州大学大学院総合理工学研究科
Graduate School of Science and Technology, Shinshu University, Nagano 380–8553, Japan
a) miyao@cs.shinshu-u.ac.jp
b) maruyama@cs.shinshu-u.ac.jp

る操作が要求される．

Cui ら [4] は，カーソル操作なしにスペルミスを含む英文を訂正可能な JustCorrect を提案している．この手法は，スペルミスを含む英文とその末尾に入力した訂正先単語をシステムに入力し，出力として得られる訂正先単語の適当な挿入および置換先の候補をリスト形式で提示しユーザに選択させる方式である．候補は，次の手順によりユーザに提示される．1）スペルミスを含む英文をスペース区切りで単語ごとに分割し，2）訂正先単語と分割した単語それぞれを置換する処理，または分割した単語間に挿入する処理によって，候補文を生成，3）各候補文の尤度および，訂正方法が置換の場合は，加えて置換対象の単語と訂正先単語の編集距離，埋め込み表現による意味的な距離をスコアリング関数として利用し，スコアが上位の文章を候補として提示する．システムを用いた実験結果より，カーソルを用いた方法に比べて，訂正効率の向上が認められ，ユーザの評価も高いことが示された．しかしながら JustCorrect では，単語間にスペースを挟む英文の構造を元にアルゴリズムが構成されている．日本語文章においても，SentencePiece や MeCab 等のトークナイザーによる単語分割が可能であるが，誤りを含む文章に対しては正確に機能しないことが考えられるうえ，分割方法はコーパスに依存するため，英文に比べてルールが明確ではない．そのため，日本語文章を対象とした場合は，この手法を適用することはできない．

自然言語モデルを活用した日本語文章の誤り訂正および誤り認識に関しても様々な研究が行われている．田中ら [5] は，日本語 Wikipedia の編集履歴を元に誤りを含む文章を収集し，衍字や脱字，誤変換といった誤りカテゴリを付与したデータセットを生成し，自然言語モデル BART のファインチューニングに利用した，日本語誤り訂正システムを提案している．また加藤ら [6] は，日本語文法誤り訂正システムを構築するうえで不足しがちな，入力データおよび正解ラベル等の学習データの増強手法を提案している．加藤らの論文で述べられているとおり，日本語誤り訂正システムを構築するうえでの課題は学習データの不足で，この問題の解決の 1 つにデータ増強があげられる．Zhang ら [8] は，Zhu ら [9] によって作成されたスマートフォンの qwerty 入力形式のソフトウェアキーボード上におけるタイピング傾向の空間モデルを利用して，スペルミスを含む英文を生成し学習データに加える手法が，自然言語モデルの精度向上に有効であることを示している．

本研究は，Cui ら [4] が提案するカーソルを使用せずに英文中のスペルミスを訂正することが可能な JustCorrect を，日本語文章中に含まれる漢字の誤変換の訂正に適用することにより，小型タッチデバイス上の日本語文章訂正の効率化を目指す．しかしながら，この手法の実現にあたっては，2 つの課題があげられる．1 つ目は，日本語と英語の文章構造の違いにより，JustCorrect の候補予測アルゴリズムを日本語文章に適用できないことである．そのため，田中らによって公開されている日本語 Wikipedia の編集履歴に基づく入力誤りデータセットを用いて ALBERT [7] をファインチューニングし，JustCorrect 同様の入出力を行うシステムを構築する．2 つ目の課題は，ファインチューニング時のデータ不足である．この課題を解決するために，Zhang ら [8] の手法を応用し，スマートフォンの qwerty 入力形式のソフトウェアキーボード上でのタイプミスによって発生する日本語の誤変換を擬似的に再現して増強データを作成し，モデルの予測精度の向上に有効であることを示す．提案システムは，iPhone 上で動作するアプリとして実装し，文章訂正の所要時間・タップ回数・ユーザの評価を計測することで本システムの有効性を検証する実験を行った．

2. 訂正候補提案モデルの作成

2.1 自然言語モデル ALBERT

システムの実装には，誤りを含む日本語文章と，訂正先単語の 2 入力から訂正先文章を出力する構造が必要になる．本研究では，日本語 Wikipedia のデータで事前学習された自然言語モデル ALBERT [10] をファインチューニングすることで実現した．ファインチューニングは，事前学習済みモデルのパラメータを初期値とした学習が可能であり，少量の学習データでも高い精度が得られ，学習コストを下げられる利点がある．また，レイヤーを 1 層追加することで，文ペア分類問題，質疑応答，系列ラベリング問題等のタスクに特化したモデルが生成可能である．本手法で作成したモデルの構成を**図 1** に示す．ファインチューニングにあたっては，入出力構成がシステムの要件に合致する質疑応答タスク用のレイヤーを追加した．質疑応答タスクは，課題文と質問文の 2 つをモデルに入力し，課題文中から質問に対する回答が含まれる箇所の始点と終点の座標を返すタスクである．レイヤー中の start scores と end scores には 1 × 訂正前文章の文字列長のテンソルに，確率値が格納されており，最大の確率値を持つ要素のインデックスをそれぞれ始点・終点として取得し，該当箇所を訂正先単語で置換することで訂正先文章を得る．なお，出力について

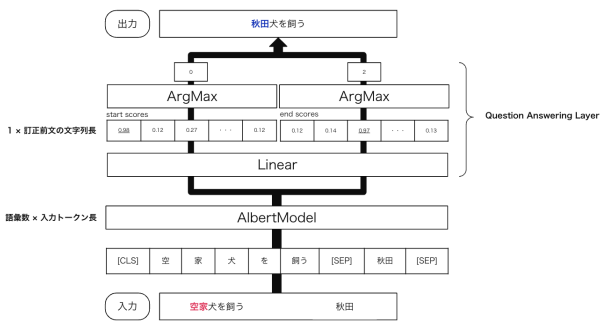


図 1 訂正候補提案モデルの構成

Fig. 1 Illustration of the correction candidate proposal model.

表 1 誤りカテゴリの例文
Table 1 Error category examples.

	例文
誤変換（a）	…大学院に【以降→移行】…
誤変換（b）	…交代【龍→理由】が…

は，始点が終点よりも必ず小さくなるように調整し，修正方法を置換のみに限定する．

2.2 学習用データ

モデルの学習には，日本語 Wikipedia 入力誤りデータセット（v2）[5] を使用した．このデータセットには，データの一単位として訂正前の誤りを含む文章，訂正後の文章，訂正箇所の差分，そして誤りカテゴリが記録されている．本研究では，データにフィルタリングをかけ，文字数が 19 文字以上 50 文字未満の短文かつ誤りカテゴリが誤変換（a）と誤変換（b）に該当し，1 文につき 1 つの誤りを含むデータのみ約 10 万件（誤変換（a）：108,356 件，誤変換（b）：11,556 件）を抽出し，学習に使用した．ここで，誤りカテゴリの誤変換（a）は，訂正箇所の差分が同一の読みを持つ漢字の誤変換，誤変換（b）は，訂正箇所の差分が近い読みを持つ誤変換である（**表 1**）．

2.3 タイピング傾向空間モデルを用いたデータ増強

学習データのうち，誤りカテゴリ誤変換（b）のデータ件数が少なく，モデルの精度不足が考えられたため，Zhang ら [8] の手法を元に下記手順によって，正解文から，qwerty 入力形式のソフトウェアキーボード上でのタイプミスによって発生する漢字の誤変換を含む文章を生成するデータ増強を行った．

(1) 入力文を形態素解析ツール MeCab [*1] により形態素に分解後，漢字を含む形態素のうち 1 つを置換対象として選択し，pykakasi [*2] によりローマ字列に変換する．
(2) ローマ字列をタイピング傾向空間モデルに入力し，タイプミスを再現したローマ字列を生成し，romkan [*3] によりひらがな文字列に変換する．
(3) ひらがな文字列を Google 日本語入力 API [*4] により漢字に変換し，入力文の選択箇所を置換する．

上記手順により，「秋田犬を飼う」という入力文から「空家犬を飼う」のような漢字の誤変換を含む文章が生成される．

日本語 Wikipedia 入力誤りデータセット（v2）の，誤りカテゴリ誤変換（a），誤変換（b）に該当しないデータの訂正後文章のうち，約 8 万文を使用し，誤変換（a）2,947 件，誤変換（b）139,891 件の増強データを生成した．なお，Google 日本語入力 API は入力したひらがな文字列を補完し最適な変換候補を出力する場合があるため，増強データ中に誤変換（a）のデータが含まれる．

[*1] https://taku910.github.io/mecab/
[*2] https://github.com/miurahr/pykakasi
[*3] https://github.com/soimort/python-romkan
[*4] https://www.google.co.jp/ime/cgiapi.html

表 2 入力誤りの訂正結果
Table 2 Input error correction result.

訂正先単語	増強データ	カテゴリ	適合率	再現率	F 値
なし	なし	誤変換（a）	36.2%	36.2%	36.2
		誤変換（b）	30.9%	30.9%	30.9
なし	あり	誤変換（a）	39.4%	39.4%	39.4
		誤変換（b）	24.7%	24.7%	24.7
あり	なし	誤変換（a）	66.9%	67.8%	67.4
		誤変換（b）	56.8%	60.5%	58.6
あり	あり	誤変換（a）	82.0%	82.9%	82.4
		誤変換（b）	74.1%	78.9%	76.4

図 2 訂正先単語の入力方法
Fig. 2 How to input the correct word.

2.4 訂正候補提案モデルの学習結果

モデルの学習にあたり，訂正先単語の入力の効果および増強データの効果を検証するための実験を行った．ファインチューニングしたモデルを使用して，学習用データと同様のフィルタリングを行って抽出したテスト用データ，誤変換（a）744 件，誤変換（b）81 件に対して誤り箇所検出を行い，適合率・再現率・F 値の指標を算出した．結果を**表 2** に示す．なお，訂正先単語無しの条件では**図 2** のモデル構成において，訂正先単語に空文字列を入力した．結果より，学習データに増強データを加えることおよび入力に訂正先単語を与えることで予測精度が改善した．一方で，誤変換（b）を対象とした訂正先単語なしの条件では，増強データを加えることで予測精度が低下した．原因については今後調査する予定である．

3. 文章訂正システムの構成

3.1 訂正候補の生成方法

モデルの学習では，start scores と end scores のテンソルから最大値が格納される要素のインデックスを取得することで訂正箇所の始点・終点を求めていたが，システムとして使用する際は，両者のテンソルから値の大きさが上位

図 3 リスト型表示の例：第 1 候補で編集中の文章を置換する．第 2 候補から第 4 候補はリスト表示され，領域外の候補はスクロールにより表示される

Fig. 3 Display example of the list proposal technique: The first candidate replaces the currently edited text. Candidates 2 to 4 are displayed in a list, and the candidates outside the area are displayed by scrolling.

図 4 循環型表示の例：第 1 候補で編集中の文章を置換する．画面右下の鉛筆ボタンをタップして順番に候補を切り替える

Fig. 4 Display example of the circular proposal technique: The first candidate replaces the currently edited text. Tap the pencil button in the bottom right of screen to switch the candidates in order.

2 位までの要素のインデックスを取得し，組合せを考えることで 4 つの候補を得る．それらの候補を下記 2 通りの方法で提示する．

3.2 訂正方法および提示方式

3.2.1 カーソルベース

本研究では，従来のカーソルを使用して文章を訂正する手法をカーソルベースと呼ぶことにする．画面タップにより誤り箇所にカーソルを移動し消去キーで文字を消去，訂正先単語を入力する手順で訂正を行う．

3.2.2 リスト型提案方式

リスト型提案方式では，JustCorrect で提案された形式を踏襲した．漢字の誤変換を含む文章の末尾に訂正先の単語を入力し（図 2），画面右下の鉛筆マークのボタンをタップすることで，訂正先の文章候補を得る（**図 3**）．モデルから出力された候補のうち，第 1 候補で編集中の文章を置換し，第 2 候補から第 4 候補はリストで表示する．リスト表示内の文章は，置換箇所が緑色の文字で表示される．リスト表示された候補をタップすると，編集中の文章が置換される．

3.2.3 循環型提案方式

訂正先の文章候補を得るまでの手順はリスト型提案方式と同様である．モデルから出力された候補のうち，第 1 候補で編集中の文章を置換し，画面右下の鉛筆マークをタップすることで，第 2 候補→第 3 候補→第 4 候補→訂正前の文章→第 1 候補と順番に編集中の文章が置換される（**図 4**）．

4. 提案システムの効果検証実験

提案システムの効果を検証するために，日本語の誤変換を含む文章を訂正するタスクを実施した．実験では，1 文を訂正する際の所要時間，テキスト領域のタップ回数，提案候補のエラー率を収集する．実験で使用したシステムの画面表示を図 2 を用いて説明する．画面上部，2 つの文章が灰色線で分割して表示される．上段の文章は目標文章との差分，下段の文章が訂正対象の文章である．上段の +○ は目標文章と比較して過足している箇所，−○ は目標文章と比較して不足している箇所を表し，指定された訂正手法を用いて上段の文章中の +− 表示がなくなるように，下段の文章を訂正することがタスクの内容である．

4.1 被験者

計 9 人の被験者（平均年齢 22 歳）を被験者とした．文章入力にスマートフォンを使う頻度は（5：頻繁に使う⇔1：まったく使わない）平均 4.7，ふだん使いするスマートフォンの機種は iOS が 78%，Android が 22%であった．

4.2 実験機材

提案を送信するサーバーとして MacBook Air（Processor Apple M1，Memory 16 GB）を使用した．文章を訂正するためのデバイスとして iPhone 6s を使用した．

4.3 実験用データ

日本語 Wikipedia 入力誤りデータセット（v2）の中から，モデルの学習およびデータ増強に使用していないデータを選択した後，誤りカテゴリ誤変換（a），誤変換（b）であるデータを抽出する．さらに，訂正箇所に常用漢字以外の漢字を含むデータを除外した（**表 3**）．各データを訂正箇所の編集距離別に集計した後，双方の誤りカテゴリから，編集距離 1 のデータを 7 件，編集距離 2 のデータを 6 件，編集距離 3 のデータを 2 件をランダムに抽出し総数 30 件のデータセットを作成する．この操作を 4 回繰り返し，合計 4 つのデータセットを作成する．1 番目のデータセットを練習用，2 番めのデータセットをカーソルベース用，3 番めのデータセットを循環型提案用，4 番目のデータセット

表 3 検証実験の際に取り除いた文章
Table 3 Sentences removed during the verification experiment.

訂正前	訂正後
... 治金工の組合に加入する... 「呵嘖生死」とは...	... 冶金工の組合に加入する... 「呵責生死」とは...

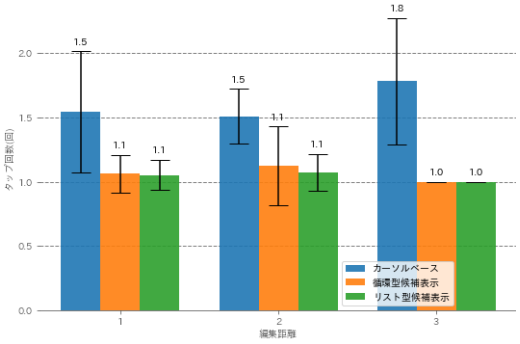


図 5 各手法のテキスト領域における平均タップ回数
Fig. 5 Average tap number in text area for each technique.

をリスト型提案用として使用した．

4.4 実験手順

最初に，年齢および，スマートフォンで文章入力を行う頻度，ふだん使用しているスマートフォンの機種を問うアンケートを実施した．2 番目に，ユーザインタフェースの見方や実験の流れ，手法の訂正方法を説明した．また，提案手法を使用する際，正解の候補が提示されないことがあり，その場合は文章をいったん初期状態に戻し，カーソルベースで訂正するよう指示した．なお，このようにして直した場合，該当手法のエラーとしてカウントする．続いて，訂正先単語を入力する際には，助詞や送り仮名等の訂正箇所前後の文字列を含んで入力してもよいことも説明した．3 番目に，練習用のデータ 30 件を用いて，各手法につき 10 件，操作確認を行った．4 番目に，各手法の実験を実施し，実験後に使用感を問うアンケートを実施した．なお，各手法の実験順序はバランスをとれるように被験者に割り当てた．また各被験者の実験終了後に，iPhone のキーボードの変換学習をリセットした．

4.5 実験結果

図 5，**図 6** に文章訂正に成功した場合の編集距離別のテキスト領域（画面上部・下段の文章領域）の平均タップ回数と平均所要時間の結果を示す．タップ回数について，カーソルベースは，編集距離ごとにタップ回数が変動する一方で，提案手法の 2 種はタップ回数を約 1 回に維持できることが分かった．所要時間については，編集距離が大きくなるにつれて，カーソルベースは所要時間が増加傾向である一方で提案手法の 2 種類は減少傾向であることが分かった．**図 7** に訂正成功率の結果を示す．カーソルベース

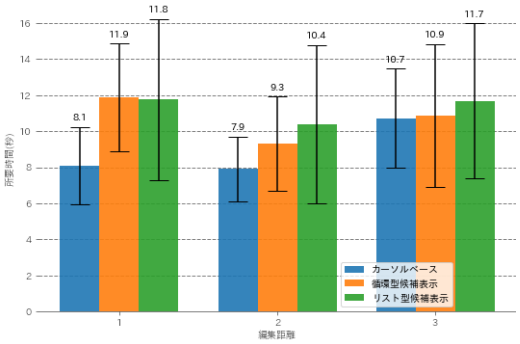


図 6 各手法の平均訂正時間
Fig. 6 Average text correction time for each technique.

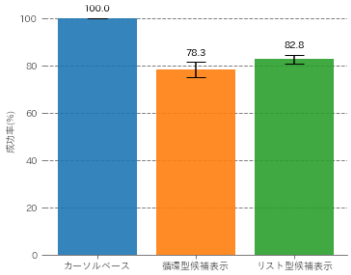


図 7 各手法の訂正成功率
Fig. 7 Correction success rate for each technique.

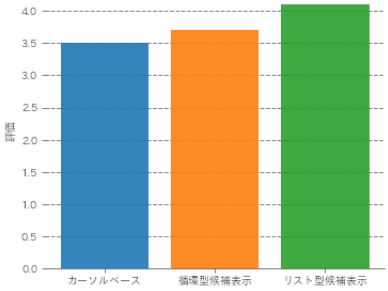


図 8 手法は使いやすかったか?（1：とても使いにくかった，5：とても使いやすかった）に対する回答
Fig. 8 Answers of the question “Is the technique easy to use?” (1: very difficult to use, 5: very easy to use).

は 100%である．同じモデルを用いているにもかかわらず，循環型提案手法とリスト型提案手法間で訂正成功率に差が現れたが 8 割程度で訂正ができることが分かった．**図 8** に実験後アンケート「各手法は使いやすかったか?（1：とても使いにくかった 5：とても使いやすかった)」の結果を示す．提案手法 2 種ともに，従来手法のカーソルベースに比べて被験者の評価が高いことが分かった．

4.6 考察

提案手法の 2 種類はいずれもカーソルベースに比べて，1 文を直すために要する時間は増加したもののタップ回数は減少し，ユーザの使用感は向上した．これにより，タッチデバイス上における文章訂正のしやすさが向上したと考えられる．また，訂正必要箇所の編集距離が大きくなるほ

ど，カーソルベースと提案手法 2 種の所要時間の差は縮まる一方で，タップ回数の差は広がる傾向が見られたことから，1 単語の訂正だけでなく，複数単語を同時に訂正可能なシステムを構築することで，より提案手法の有効性を見いだせると考える．提案手法 2 種間のユーザの評価および訂正成功率を比較すると，循環型よりもリスト型の方が評価が高かった．この原因として，循環型は，正しい候補が画面に表示されなかった場合，次の候補の表示に鉛筆ボタンをタップする手間が必要となる．これにより正しい訂正候補の見落としが発生して訂正成功率が低下し，同時に被験者の評価も低下したと考えられる．一方で，循環型はリスト型に比べて，画面の占有領域が少ないため表示領域が限られる場面では，有効であると考えられる．なお，エラーがユーザの見落としによるものか，正解の候補が提示されなかったためかを判断するためのより詳細な操作追跡の実装は，今後の課題である．

5. まとめ

本研究では，スマートフォンに代表される小型のタッチデバイス上における日本語文章の訂正を効率化する手法を提案し，実施した文章訂正タスクの結果より，カーソルベースに比べて，ユーザの使用感が向上すること，訂正箇所の編集距離が大きいほど有効であることを示した．学習用データ件数の都合上，対象の誤りを漢字の誤変換に絞ったうえで検証を行った．他の誤り（助詞の欠落・衍字）等に対してもシステムを対応させ有効性を検証することが，今後の課題である．

参考文献

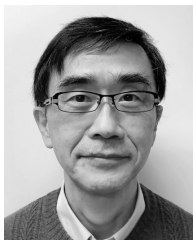
[1] Siek, K.A. et al.: Fat finger worries: How older and younger users physically interact with PDAs, *Proc. 2005 IFIP*, pp.267–280 (2005).
[2] iPhone でオンスクリーンキーボードを使って入力する：Apple サポート（日本），入手先 ⟨https://support.apple.com/ja-jp/guide/iPhone/iph3c50f96e/ios⟩（参照 2022-11-01）．
[3] キーボードを使用する：Android - Gboard ヘルプ，入手先 ⟨https://support.google.com/gboard/answer/2842292?hl=ja⟩（参照 2022-11-01）．
[4] Cui, W. et al.: JustCorrect: Intelligent Post Hoc Text Correction Techniques on Smartphones, *Proc. UIST 2020*, pp.487–499 (2020).
[5] 田中　佑ほか：日本語 Wikipedia の編集履歴に基づく入力誤りデータセットと訂正システムの改良，言語処理学会第 27 回年次大会発表論文集，pp.1541–1545 (2021).
[6] 加藤秀佳ほか：日本語文法誤り訂正におけるデータ増強および評価データ構築，言語処理学会第 27 回年次大会発表論文集，pp.101–106 (2021).
[7] Lan, Z. et al.: ALBERT: A lite BERT for self-supervised learning of language representations, arXiv preprint arXiv:1909.11942 (2019).
[8] Zhang, M.R. et al.: Type, Then Correct: Intelligent Text Correction Techniques for Mobile Text Entry Using Neural Networks, *Proc. UIST 2019*, pp.843–855 (2019).
[9] Zhu, S. et al.: Typing on Invisible Keyboard, *Proc. CHI 2018*, pp.1–13 (2018).
[10] ALINEAR/albert-japanese-v2 · Hugging Face, available from ⟨https://huggingface.co/ALINEAR/albert-japanese-v2⟩ (accessed 2022-11-01).

佃 充宏

1997 年生．2021 年信州大学工学部電子情報システム工学科卒業．同大学大学院修士課程在学中．モバイル端末の操作性向上・自然言語処理に興味がある．

宮尾 秀俊　（正会員）

1989 年長岡技術科学大学電子機器工学課程卒業．1991 年長岡技術科学大学電気電子システム工学専攻修了．現在，信州大学工学部電子情報システム工学科准教授．博士（工学）．パターン認識・学習，HCI の研究に従事．電子情報通信学会，ACM 各会員．



丸山 稔　（正会員）

1982 年東京大学工学部計数工学科卒業．同年三菱電機（株）入社．1990〜1991 年 MIT 人工知能研究所 visiting scientist．現在，信州大学工学部電子情報システム工学科教授．博士（工学）．コンピュータビジョン，機械学習等の研究に従事．電子情報通信学会，ACM，IEEE 各会員．