

## BlindX におけるかな漢字変換 LLM の実装と評価

田中 慎太郎  
東京理科大学

6324521@ed.tus.ac.jp

鈴木 海友  
東京理科大学

kaiyu.suzuki@rs.tus.ac.jp

松澤 智史  
東京理科大学

matsuzawa@rs.tus.ac.jp

### 要旨

PC など計算機への日本語入力には、様々な日本語入力システム (IME) が利用されている。入力効率改善のため、かな漢字変換の精度向上などの技術が長らく研究され、特に近年は、深層学習や大規模言語モデル (LLM) によるアプローチも増えている。著者らが開発している新たな日本語入力システム「BlindX」は、LLM を用いた高精度なかな漢字変換を基軸に、入力ミスの自動修正や用途・ユーザに最適化した変換などの様々な機能を柔軟に構成できるよう設計している。

本研究では、かな漢字変換に用いる LLM を、3 種類のテキストデータをもとに実装した。実験により変換精度を評価した結果、モデル訓練時と同じドメインの入力に対する変換では既存 IME や LLM と同等以上の精度を示した。異なるドメインの入力に対しては精度が低下したが、学習した知識量や文体の違いが変換精度に影響することが示唆された。今後は、特定ドメインの変換に特化したモデルを複数併用するしくみを具体化することで、ドメイン知識を統合し、多様な入力に対する最適な変換に繋げる。

### 1. はじめに

現在、PC など計算機に日本語を入力する際には、様々な日本語入力システム (IME) が利用されている。ユーザは入力したい文章の読みをひらがなで打ち込み、変換キーを押すことで IME のかな漢字変換機能呼び出しで変換し、必要ならば修正を経て、変換を確定する。

変換の修正はユーザが手動で行う必要があるため、一度で正確に変換できることが望ましいが、実際は同音異義語や低頻度語などで誤変換が生じる場合がある。また

キーボード入力時に発生するユーザの誤入力も、手動での修正が必要な場合が多く、入力効率の低下に繋がる。

このため、かな漢字変換の精度向上や誤字の自動修正、予測変換など、IME の各要素に関する技術については長らく研究されてきた。特に変換精度の向上については、近年の深層学習や大規模言語モデル (LLM) の発展から、深層学習による言語モデルを用いた研究が増えている [1, 2, 3]。一方で、こうした技術を実用化した例は著者らの知る限りまだ少ない。また、かな漢字変換以外の機能も含む日本語入力全体を、LLM によって包括的に高度化するアプローチはまだみられない。

著者らは 2023 年から AX テックケア株式会社<sup>1</sup>と共同で、新たな日本語入力システム「BlindX」の開発を行っている。このシステムでは LLM を用いることで、高精度なかな漢字変換を核としつつ、入力ミスの自動修正や用途・ユーザへの最適化などの周辺機能を高度に実現し、入力効率の改善を目指している。またこれらの機能拡張・変更を柔軟に行えるように設計し、個々のユーザの要求や好みに応じた日本語入力の実現をねらう。

本稿では、BlindX の設計について紹介するとともに、核となるかな漢字変換モデルについて、LLM を用いた現実装とその評価、今後の課題について述べる。

### 2. 関連研究

かな漢字変換技術は長らく研究されていて、当初はヒューリスティックや人手で設定された規則に従う方式で行っていたが、次第に言語モデルを用いた手法が研究されるようになった。例えば森ら [4] は、確率的言語モデルを用いたかな漢字変換を提案し、Google 日本語入力<sup>2</sup>の OSS 版である Mozc [5] の実装にも取り入れられた。

<sup>1</sup><https://axtechcare.com/>

<sup>2</sup><https://www.google.co.jp/ime/>

また識別的手法に関する研究も存在する [6, 7].

近年、自然言語処理分野においては深層学習を用いた言語モデルの研究が進んでいる。特に Transformer[8] 以降、そのアーキテクチャをベースとした LLM の研究が急速に発展し、大量のパラメータと Attention 機構による高度な文脈把握力を活かして、幅広いタスクへの応用が進んでいる。

こうした発展を踏まえ、日本語入力においても RNN[1] や Transformer[2] などの深層学習モデルを適用した研究例がある。実際の IME でも適用されていて、例えば深層学習により変換や入力ミス修正の精度を改善した ATOK<sup>3</sup>や、LLM の一種である GPT-2 を変換ベースとした azooKey on macOS[3] が挙げられる。

### 3. BlindX の設計

BlindX はクライアントサーバシステムとして設計する。主機能であるかな漢字変換は、サーバ上の「Core モデル」が担う。

かな漢字変換の基本的な処理の流れは次の通りである (図 1)。

1. クライアント上でユーザーが入力したひらがな文を、API によりサーバへ送信する
2. Core モデルにより、サーバが受信したひらがな文をかな漢字交じり文へと変換する
3. 変換したかな漢字交じり文をクライアントへ送信する
4. クライアントがそれを受信し UI 上に表示する

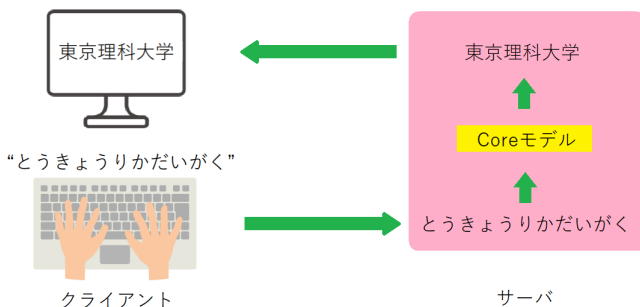


図 1. 基本の処理の流れ

### 3.1. Core モデル

Core モデルは、ひらがな文を入力として受け取り、かな漢字交じり文に変換し出力する。BlindX では、かな漢字変換タスクを学習した LLM を利用することで高精度な変換を目指す。実装の詳細は 4 章で述べる。

一般に、「かな」から「漢字」への変換では複数の候補が考えられる。この中には、語法や前後の文脈に照らして不適切なものも含まれ、これらはかな漢字変換 LLM による変換で排除されることが期待される。しかし、これだけで変換を一意に決定できるとは限らない。すなわち、ユーザーが入力する文章の目的や使用場面<sup>4</sup>、ユーザー個人の変換の好みなども、変換の決定に関係する。

変換速度が重要な指標であるかな漢字変換において、大規模な変換モデルを扱うことは現実的に難しい。そのため、専門用語・固有名詞・スラング等を含めたあらゆる言葉の変換や、変換に関わる様々な要素の反映を、1 つのモデルでまかなうことは困難である。

そのため、BlindX における Core モデルは、複数の変換モデルによる構成を可能とする。そのうえで、これらのモデルの使い分けや統合による、状況や入力に応じた変換最適化のしくみを検討している (図 2)。なお、個々の変換モデルは「ひらがな文を受け取りかな漢字交じり文に変換して出力する」仕様を満たす限り、LLM 以外のモデルを使用することもできる。

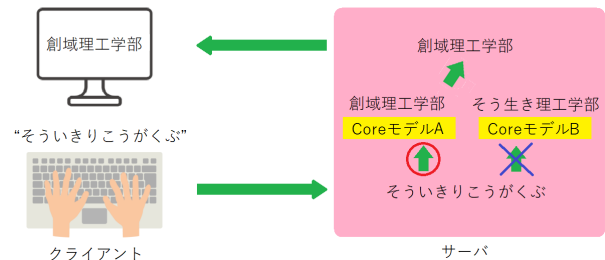


図 2. 複数の Core モデルの利用

### 3.2. 周辺機能と構成の拡張

BlindX は、様々な入力方式や構成の拡張を考慮し、変換に必要な個々の処理を分離して行う設計となっている。

<sup>4</sup>例えば、業務文書の作成と、友人との気軽なチャットでは、入力する文章の文体は大きく異なる。

<sup>3</sup><https://atok.com/>

日本語入力的方式には、ローマ字入力またはかな入力によるキーボード入力のほか、スマートフォンでのフリック入力、音声認識などが存在する。しかし、いずれの方式でも入力はすべてひらがな文の形式に変換されたうえで、Core モデルで漢字に変換される。クライアントサーバのもと、変換処理を共通とすることで、デバイスや環境によらず前述の Core モデルの利便性を受けることができる。

これに限らず、ユーザの入力から Core モデルに与えられるまでの文字列、および Core モデルの出力から UI に表示されるまでの文字列に対しては、任意の処理を加えることができる。これにより、Core モデルによるかな漢字変換を基軸としながら、多様な機能拡張が考えられる。

例として、Core モデルへの入力前にひらがな文中の誤字を修正するしくみが考えられる。ユーザの誤入力や不正確な音声認識など、どの入力方式でも得られるひらがな文に誤りが含まれる可能性があるが、その特性は方式によって異なる。そのため、各方式によって異なる機構を用いることで、修正精度を高めることができる (図 3)。

日本語入力における各機能の処理を分離し、機能ごとに個別の機構とすることにより、構成の追加や変更が柔軟に行えるという利点がある。

## 4. Core モデルの実装

本章では、BlindX の Core モデルに使用するかな漢字変換 LLM の実装について述べる。

### 4.1. 使用するモデル

モデルとして、Transformer アーキテクチャを持つ Encoder-Decoder 型 LLM の一種である T5[9] を用いる。本研究では園部が作成・公開した事前学習済 T5-base (約 2.5 億パラメータ) および tokenizer<sup>5</sup> を使用する。

### 4.2. データセット

訓練・評価に用いるデータセットは、ひらがな文とそれに対応するかな漢字交じり文のペアを 1 サンプルとし、サンプルを多数収集して構築する。ソースとなる日本語

テキストとして、本研究では 3 種類のテキストデータを用いる。

- Wikipedia 日本語版  
フリーのインターネット百科事典であり、様々な事柄に関する説明が掲載されている。本文データのダンプファイル<sup>6</sup>からテキストを抽出して利用する。なおダンプファイルは定期的に更新されており、本研究では 2024 年 12 月 1 日時点のものを用いる。1 段落を 1 サンプルとして扱い、サンプル数は約 1700 万である。
- おーぶん 2 ちゃんねる対話コーパス  
稲葉 [10] が構築したコーパスであり、「なんでも実況(ジュピター)」「ニュー速 VIP」「ニュース速報+」各掲示板について、開設時から 2019 年 7 月 20 日までに投稿された対話を収集したものである。1 対話を 1 サンプルとして扱い、サンプル数は約 800 万である。
- 「Yahoo!知恵袋データ (第 3 版)」2024 年度提供版  
Yahoo!知恵袋は、LINE ヤフー株式会社が運営する、ユーザ同士で質問やそれに対する回答を投稿し合う Web サービスである。本研究では、国立情報学研究所 (IDR) 様から提供いただいたデータセット [11] を利用する。これは、Yahoo!知恵袋において解決済みとなった 2019~2021 年度の質問とその回答の一部を抽出したものである。質問と回答の区別はせず、1 つの質問または 1 つの回答を 1 サンプルとして扱う。サンプル数は約 700 万である。

ただし、これらのテキストデータから直接得られるのはかな漢字交じり文のみであり、ひらがな文は含まれない。データセットの構築には、かな漢字交じり文に対応するひらがな文の取得が必要である。ただし手動での付与はコストが高く非現実的であるから、アルゴリズムによる自動での付与が求められる。

そこで、形態素解析を用いてテキストから読みを取得する。日本語の形態素解析は、ある文章を単語に分割し、各単語に関する品詞や読み、発音などの情報を推定する。本研究では、形態素解析エンジンとして MeCab[12]、辞書として mecab-ipadic-NEologd[13, 14, 15] を用いる。本構成による形態素解析の例を図 4 に示す。推定された

<sup>5</sup><https://huggingface.co/sonoisa/t5-base-japanese-v1.1>

<sup>6</sup><https://dumps.wikimedia.org/jawiki/latest/jawiki-latest-pages-articles.xml.bz2>

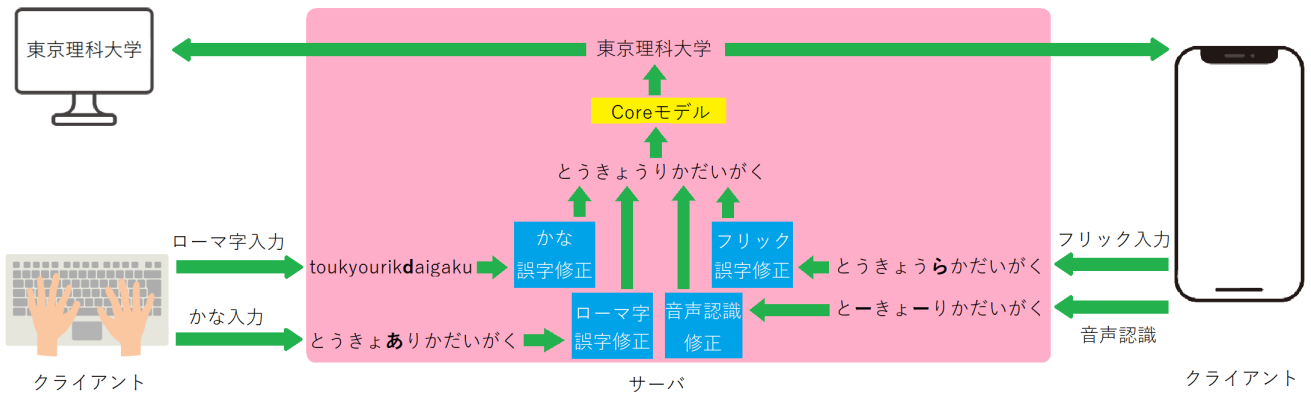


図 3. 複数入力方式への対応と誤字修正の拡張

各単語の読みを結合し、ひらがなに変換することで、ひらがな文を得る。

以上の方法により、3種類のテキストデータからデータセットをそれぞれ構築する。このとき抽出したサンプルの9割を訓練セット、1割をテストセットに分割する。

なお本研究では、計算環境の制約により、訓練セットの文字数上限を256字、テストセットを512字として、極めて長いサンプルをデータセットから除外した。除外したサンプルはWikipediaの1.9%、おーぷん2ちゃんねるの0.1%、Yahoo!知恵袋の4.9%にあたる。

### 4.3. ファインチューニング

構築したデータセットのうち訓練セットを使用してT5をファインチューニングすることで、かな漢字変換タスクを学習する。具体的には、訓練セットに対し、以下の手順を繰り返し行う。

1. サンプルをランダムに取り出す
2. サンプルに含まれるひらがな文とかな漢字交じり文を、tokenizerでそれぞれトークン列  $I, T$  に変換する
3.  $I$  をT5に入力し、出力としてトークン列  $O$  を得る
4.  $O$  と  $T$  のクロスエントロピー誤差に基づき、T5のパラメータを更新する

## 5. Coreモデルの評価

Coreモデルによるかな漢字変換の精度を評価するため、実験を行う。本研究では、ひらがな文をCoreモデル

に入力して得た出力の、もとのかな漢字交じり文に対する一致度を、変換精度として定義し、定量的指標により測る。

評価対象は、Wikipedia、おーぷん2ちゃんねる、Yahoo!知恵袋でそれぞれ学習した3つのCoreモデルである。また比較対象として、既存のIME (MS-IME, Google日本語入力) およびLLM (OpenAI GPT-4o, GPT-4.5)<sup>7</sup>を用いる。

変換については、次の規則に従う。

- 入力するひらがな文は、MS-IMEでローマ字入力する際に表示される形式に従う。すなわち、カタカナや英数字記号空白は全角に統一し、「う`」は「ヴ」とする。
- 既存IMEでの変換は、ひらがな文に対する再変換によって行う。このとき、IMEに搭載されている学習機能は無効とする。ただし一度に再変換できるのは、MS-IMEは50字まで、Google日本語入力は200字までであるため、それよりも長い入力については、句点など適当な位置で区切って変換する。
- LLMへの入力については、かな漢字変換タスクを行わせるよう、次のようなプロンプトを与える。  
「以下のかな文を正しいかな漢字交じり文に変換し、変換した文のみ出力してください。

(ひらがな文)」

- Coreモデルで用いるtokenizerの仕様上、英数字記号空白がいずれも半角で出力される。これに合

<sup>7</sup>モデル名はそれぞれ gpt-4o-2024-08-06, gpt-4.5-preview-2025-02-27

福島県福島市	名詞,固有名詞,地域,一般,*,*,福島県福島市	フクシマケンフクシマシ	フクシマケンフクシマシ
に	助詞,格助詞,一般,*,*,*,に	ニ	ニ
ある	動詞,自立,*,*,五段・ラ行,基本形,ある	アル	アル
生涯学習	名詞,固有名詞,一般,*,*,*,生涯学習	ショウガイガクシュウ	ショウガイガクシュウ
施設	名詞,サ変接続,*,*,*,*,施設	シセツ	シセツ
で	助動詞,*,*,*,特殊・ダ,連用形,だ	デ	デ
ある	助動詞,*,*,*,五段・ラ行アル,基本形,ある	アル	アル
。	記号,句点,*,*,*,*,。	。	。

図 4. 例文「福島県福島市にある生涯学習施設である。」に対する形態素解析。赤色部分が、推定された読みを表す。

せ、既存 IME や LLM の出力に含まれる英数字記号空白も半角に変換する。またカタカナは全角に統一する。

評価に用いるデータセットは、以下の 4 種類である。

- Wikipedia
- おーぷん 2 ちゃんねる
- Yahoo!知恵袋

各テストセットから無作為に 150 サンプルずつ抽出し、3 種類の評価データを構築した。これらのひらがな文は形態素解析エンジンの推定によって求めているため、一部に誤りを含む。正確な評価のため、これらは予め手で修正した (表 1)。

- Anthy コーパス

かな漢字変換エンジン「Anthy」で利用されているコーパス<sup>8</sup>から無作為に抽出した 150 サンプルを使用する。このコーパスはひらがな文とかな漢字交じり文の双方を含む。一般的なドメインの入力が多く、専門用語や固有名詞は少ない。サンプルに含まれるひらがな文の文字数は最大 54 字であり、上記 3 種類の評価データと比べて短い。

また、ひらがな文の誤りのみ修正したものを「入力形式 A」、さらに以下の修正をひらがな文に加えたものを「入力形式 B」とし、両方の形式で実験を行う。形式 B は、誤りとは言えないものの、既存 IME との公平な比較のために設定した。

<sup>8</sup><https://github.com/netsphere-labs/anthy/blob/master/corpus/corpus.1.txt>

表 1. 評価データの手動修正

	Wiki	おーぷん 2ch	知恵袋
修正サンプル数	91	62	102
編集距離	1090	319	2198
文字誤り率 (CER)	3.17%	2.09%	5.87%

- 一部の英数字記号に対応するひらがな文  
例えば、かな漢字交じり文が「2025 年」のとき、これに対応するひらがな文が「にせんにじゅうごねん」となっている場合がある。これは誤りではないが、既存 IME を用いる場合は「2025ねん」と入力することが多く、前者の入力では精度が低下する場合がある。したがって形式 B では、数字のほか英字や記号に対応するひらがな文は後者の形式で統一した。
- 記号や空白の入力省略箇所  
本研究で用いた形態素解析辞書では、一部の固有名詞等の読みについて、発音しない記号や空白が省かれているものがある (例えば「イエス・キリスト」の読みが「いえすきりすと」となっている)。こうした例は既存 IME にも一部存在する (例えば MS-IME では「らきすた」と入力すると「らき☆すた」と変換できる) が、形態素解析辞書にはより多く登録されている。ゆえに形式 B では、省略されている記号や空白を補った。

評価指標には、文字誤り率 (CER) のスケールを反転した 1-CER、および BLEU[16] を用いる。いずれも 0 以上 1 以下の数値であり、1 に近いほどものかな漢字交じり文との一致度が高いことを表す。これを 150 サンプ

ルそれぞれについて求め、その平均を最終的な指標値とする。

結果は表 2 (1-CER)、表 3 (BLEU) の通りである。Wikipedia・おーぷん 2 ちゃんねる・知恵袋の評価データについては、同じ種類の訓練セットで学習した各 Core モデルが高い精度を示し、多くの場合では既存 IME や LLM を上回り最も高くなった。一方で、訓練時と異なる種類の評価データに対しては、既存 IME や LLM と同程度かそれより低い値を示した。

また Anthy に対しては、GPT-4.5 や MS-IME が特に高い精度を示した。Core モデルでは、知恵袋で学習したモデルが Google 日本語入力とほぼ同等の値となった一方、Wikipedia やおーぷん 2 ちゃんねるのモデルでは精度が低下した。

## 6. 考察

### 6.1. ドメインと精度の関係

本実験で用いた 3 つの Core モデルは、それぞれ異なる種類のデータをもとに、かな漢字変換タスクを一から学習している。ゆえに、変換結果はそれぞれの訓練時のデータに強く依存しており、このことが変換精度にも大きく影響している。すなわち、訓練データと評価データが同じ種類、すなわち同じドメインである場合に精度が最も高く、そうでなければ精度は低下する。

同じドメインの場合、Core モデルは既存 IME や LLM と同等以上の精度であることから、そのドメインに特化した高精度な変換モデルとしては有用である。以降は、ドメインが異なる場合について述べる。

訓練時と異なるドメインの評価データでも、種類により精度低下の程度に差がある。表 4 は、各評価データについて、同じドメインで学習したモデルでの変換精度を 100 としたときの、他のモデルの相対的な精度の値を示したものである。まず、Wikipedia の評価データに対して、知恵袋で学習したモデルでは Wikipedia で学習したモデルの約 91% (1-CER)、87% (BLEU) の変換精度を示した。一方、おーぷん 2 ちゃんねるで学習したモデルでは Wikipedia で学習したモデルの約 82~84% (1-CER)、76~78% (BLEU) にまで精度が下がっている。いずれも Wikipedia とは異なるドメインのデータで学習したモデルであるが、知恵袋で学習したモデルがおーぷん 2 ちゃんねるよりも高い精度となった。

次に、おーぷん 2 ちゃんねるの評価データに対しては、知恵袋のモデルが約 96% (1-CER)、94% (BLEU)、Wikipedia のモデルでは約 92% (1-CER)、88% (BLEU) であり、こちらも知恵袋のモデルが Wikipedia のモデルを上回った。ただし、前述の Wikipedia に対する 2 つのモデルの場合よりも、精度低下の幅は小さい。

最後に、知恵袋の評価データに対しては、Wikipedia のモデルが約 98% (1-CER, BLEU)、おーぷん 2 ちゃんねるのモデルが約 95% (1-CER)、93~94% (BLEU) となった。Wikipedia のモデルがおーぷん 2 ちゃんねるのモデルを上回ったが、精度低下の幅はさらに縮まった。

以上の結果から、3 つのドメイン間の関係が読み取れる。Wikipedia とおーぷん 2 ちゃんねるは、一方で学習するともう一方に対する精度が大きく下がることから、かな漢字変換を行ううえで求められる要素が大きく異なる、離れたドメインである。他方で知恵袋は、他の種類のデータで学習しても変換精度の劣化が比較的小さく、ドメインとしては前述 2 つの中間に位置している。

これらのドメインの違いとは、具体的には以下のものであると考える。

- 知識量

Wikipedia は、多様な分野の事柄を扱い、専門分野の用語や固有名詞などの知識が訓練データに多く含まれている。「果実連」「ヴェーダ」「志賀重昂」など一般的には出現頻度の低い語も含め、正しく認識・変換することが多い。

知恵袋やおーぷん 2 ちゃんねるでも幅広いジャンルの投稿があるが、専門的な内容まで多く含むわけではない。そのため、特定のドメインによらない一般的な知識は学習できても、専門的な知識を得るまでには至らなかったと考える。

ただし、本研究では 3 種類の訓練セットのサンプル数を揃えておらず、Wikipedia とそれ以外とでサンプル数には大きな開きがある。そのため、知識量による影響の程度についてより正確に測るためには、サンプル数を合わせたうえでの実験も求められる。

- 文体

百科事典ゆえ書き言葉によるかたい文章が多い Wikipedia に対し、知恵袋やおーぷん 2 ちゃんねるはより砕けた表現も多く用いられる。「ほんまに」「言うて」などは Wikipedia のモデルでは変換できないことがある。

表 2. 実験結果：1-CER

評価データ	形式	Core モデル			既存 IME		LLM	
		Wikipedia	おーぶん 2ch	知恵袋	MS-IME	G 日入	GPT-4o	GPT-4.5
Wikipedia	A	<b>0.96610</b>	0.79135	0.88246	0.86412	0.88418	0.90482	0.90171
Wikipedia	B	<b>0.96584</b>	0.80926	0.88222	0.92869	0.93480	0.92411	0.92606
おーぶん 2ch	A	0.85956	<b>0.93665</b>	0.90297	0.86895	0.89208	0.87610	0.88942
おーぶん 2ch	B	0.86119	<b>0.93775</b>	0.90422	0.88155	0.90396	0.88762	0.89730
知恵袋	A	0.92968	0.89574	<b>0.94559</b>	0.92153	0.93393	0.92338	0.92084
知恵袋	B	0.93276	0.89651	0.94772	0.93512	<b>0.94802</b>	0.93272	0.93743
Anthy	A	0.92671	0.92804	0.94963	0.95893	0.95081	0.94333	<b>0.96303</b>
Anthy	B	0.92671	0.92804	0.94963	<b>0.95988</b>	0.95176	0.94333	0.95802

表 3. 実験結果：BLEU

評価データ	形式	Core モデル			既存 IME		LLM	
		Wikipedia	おーぶん 2ch	知恵袋	MS-IME	G 日入	GPT-4o	GPT-4.5
Wikipedia	A	<b>0.94730</b>	0.72139	0.82244	0.80774	0.83637	0.86705	0.87306
Wikipedia	B	<b>0.94894</b>	0.74152	0.82214	0.87792	0.88909	0.90047	0.90007
おーぶん 2ch	A	0.78856	<b>0.90082</b>	0.85017	0.79790	0.82754	0.79637	0.81713
おーぶん 2ch	B	0.79149	<b>0.90193</b>	0.85178	0.81398	0.84459	0.81302	0.83089
知恵袋	A	0.88974	0.85303	<b>0.91081</b>	0.87420	0.88920	0.86721	0.85770
知恵袋	B	0.89389	0.85374	<b>0.91360</b>	0.89258	0.90843	0.87938	0.87918
Anthy	A	0.88646	0.89048	0.91357	0.92795	0.91242	0.91560	<b>0.93630</b>
Anthy	B	0.88646	0.89048	0.91357	0.93065	0.91512	0.91560	<b>0.93626</b>

表 4. 訓練時と同じドメインのモデルでの変換精度に対する、他モデルの相対精度（単位：%）

評価データ	形式	訓練データ					
		Wikipedia		おーぶん 2ch		Yahoo!知恵袋	
		1-CER	BLEU	1-CER	BLEU	1-CER	BLEU
Wikipedia	A	-	-	81.91	76.15	91.34	86.82
	B	-	-	83.79	78.14	91.34	86.64
おーぶん 2ch	A	91.77	87.54	-	-	96.40	94.38
	B	91.84	87.76	-	-	96.42	94.44
Yahoo!知恵袋	A	98.32	97.69	94.73	93.66	-	-
	B	98.42	97.84	94.60	93.45	-	-

特におーぶん2ちゃんねるは、知恵袋以上に砕けた表現が多用され、Webの匿名掲示板に特有の言い回しやスラングも含まれる。「ワイ」などが一例であり、知恵袋のモデルでも変換を誤ることがある。

一方で知恵袋は、匿名で気軽に参加できることから砕けた表現も見られるが、他者とのコミュニケーションを行うため丁寧な文章での投稿も多く、投稿によって文体はかなり異なる。このため、知恵袋のモデルは多様な文体を学習しているとも言える。

知恵袋は「広く一般的な知識」と「硬軟混ざった文体」を含む点で、知識量および文体のバランスが良いと考える。実際、知恵袋で学習したCoreモデルは他2つのモデルと比べ、訓練時のドメイン以外の入力に対する精度が最も安定している。したがって、これらの特徴は「一般ドメイン」の変換の要件と捉えることができ、これに加えて「専門分野の知識」や「特定の使用目的」などに応じたモデルを併用することが、複数のCoreモデルの効果的な併用のあり方であると考えられる。

## 6.2. 変換速度

Coreモデルの変換にかかる処理時間は、実行環境により異なる。図5は、以下の環境において、3つのCoreモデルがWikipedia評価データ（入力形式B）の各サンプルの変換にかかった時間をそれぞれ計測し、プロットしたものである。

- CPU：AMD Ryzen 5 7600
- RAM：DDR5-3600 128GB
- GPU：NVIDIA Geforce RTX 5060 Ti 16GB
- OS：Microsoft Windows 11 Home

GPUを利用した推論（図の青色）と、CPUのみによる推論（図の赤色）それぞれで処理時間を計測した。

Coreモデルでの変換にかかる処理時間は、入力するひらがな文の長さにはほぼ比例して増加した。結果にはばらつきがあるが、入力が100字増えるごとに、GPUを使用した推論では約0.35秒ずつ、CPUのみの推論では約1.3秒ずつ処理時間が増加した。特にCPUでの推論では数十文字の入力で、変換までに1秒以上のレイテンシが発生し、ユーザー体験に大きな悪影響を及ぼすと考える。

本研究で用いたモデルは約2.5億パラメータのT5-baseであり、T5-small（約6000万パラメータ）などより軽

量なモデルの利用や量子化が対策として考えられるが、変換精度とのバランスを確認する必要がある。また実システムに組み込む際には、関連研究[2]で提案されているようなデコーディングの工夫なども検討すべきである。

## 6.3. データセットの構築に関して

学習に用いるデータセットの構築において、Coreモデルの精度に悪影響を与えうる要素がいくつか挙げられる。

- ソースの日本語テキストに含まれる誤字  
本研究で用いた3種類のテキストデータは、誰でも投稿や編集を行えるWebサービスであり、誤字や誤変換が含まれていることがある。本研究ではこうした誤字に関する対処はしていないため、Coreモデルの変換の誤りに繋がっている可能性がある。
- データセットからの長文の除外  
本研究では、極めて長いサンプルをデータセットから除外した。よって、除外したサンプルに含まれる変換を学習できていない可能性がある。
- 形態素解析アルゴリズムにより推定した読みの利用  
本研究では、形態素解析アルゴリズムによってかな漢字交じり文から自動的にひらがな文を求めた。この精度は完全ではなく、新語<sup>9</sup>や低頻度語を辞書が網羅していない場合などで推定の誤りが発生する。  
また本研究では、形態素解析においてMeCabとNEologd辞書による推定のみ、かつ最も確からしい結果のみを用いたが、正しい読みが複数存在する場合に特定の読みばかり推定され、変換精度に悪影響を及ぼすことがある。例えば「おとことおとこのぶつかりあい」→「漢と漢のぶつかり合い」は、「漢」に「おとこ」と付与された例が訓練セット中に存在しない（すべて「かん」と付与されている）ため、3種類のCoreモデルはいずれも変換できなかった。

ただし、本研究で構築したデータセットは多くのサンプルを含むため、上記による変換精度への悪影響は大きくないと考える。実際、形態素解析の読みに関して、評価データのうち人手で修正を加えた箇所については、正しく変換されるか無変換（ひらがなのまま出力）が多く、一部に誤変換が見られた。ただし無変換や誤変換につい

<sup>9</sup>本研究で用いた辞書NEologdも、2020年以降更新されていない。

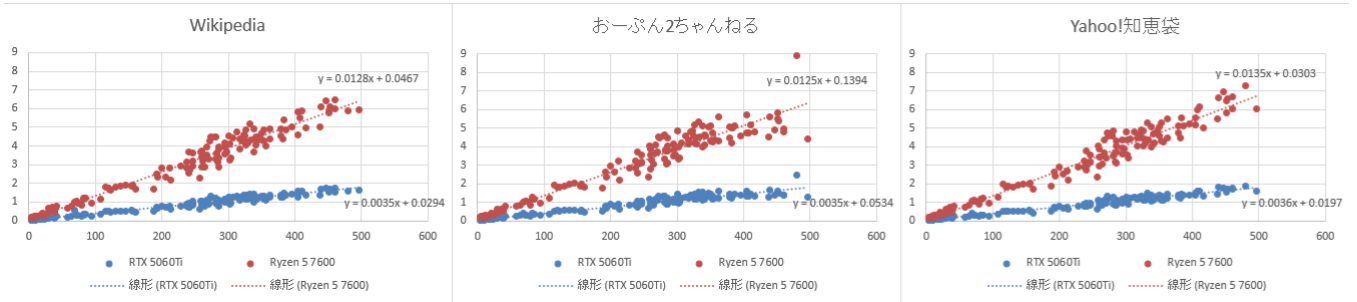


図 5. Wikipedia 評価データ（入力形式 B）に対する、各 Core モデルの処理時間。横軸＝入力ひらがな文の長さ、縦軸＝処理時間（秒）

でも、T5 の Decoder が出力する各トークンの確率分布を確認すると、正しい変換が上位に現れる場合が多かったので、正しいひらがなでの変換の学習自体は行えている。すなわち、訓練セット中の他のサンプルと同じ漢字があり、そこで正しい読みが付与されていれば学習できる（前述の「漢」の例はこれに該当しない）。これは、もとのテキスト中の誤字や、除外したサンプルに含まれる変換についても同様のことが言える。

最後に、各要素への対策を挙げる。ソースに含まれる誤字については、データセット構築時に誤字検出のアルゴリズム等を用いて少しでも減らす。長文に関しては、形態素解析で「句点」と品詞推定された箇所をテキストを分割し、別々のサンプルとする。読み推定では、高精度なアルゴリズム・辞書の選別とそれらの複数併用により、ひらがな文の精度向上と多様性確保を目指す。

## 7. 今後の展望

複数の Core モデルの併用について、具体的な手法が求められる。すなわち、ユーザの入力内容等に応じて、変換に用いる Core モデルを自動的に選択する仕組みである。この際、選択するモデルは1つとは限らない。変換すべき内容が複数のドメインと関係する場合、複数の Core モデルに変換させ、各出力を適切に統合することで最終的な変換結果を得る。モデル選択のための分類器や、複数出力の統合アルゴリズムの構築が課題である。

また、複数ドメインの知識を統合するためのアプローチは他にも、各ドメインの訓練セットを混合させ学習、あるドメインで学習したモデルを別のドメインでファインチューニング、複数のモデルのマージ [17, 18] などが挙げられ、これらの手法との比較評価も求められる。

Core モデルの容易な構築も望まれる。特に、ユーザによりオリジナルの Core モデルを容易に構築できるようになれば、ユーザへの変換最適化や入力効率改善に繋がる。本研究では変換タスクを一から学習するため、数百万行以上のテキストデータを用いたが、一般にサイズはより小さく、一からの学習は現実的でない。代わりに、学習済モデルをベースとした追加学習が考えられる。計算コストを抑えた高効率な学習のため、通常ファインチューニングや LoRA [19] など、学習手法も検討する。

さらに3章にて挙げた、様々な入力方式への対応、入力ミス修正などの周辺機能について、具体的な実装に取り組む。また、クライアントのアプリケーションの開発を行い、システム全体としての有用性の評価を行う。

## 8. おわりに

本研究では、新たな日本語入力システム「BlindX」を設計し、T5 を用いたかな漢字変換の実装を行った。また、様々なドメインの評価データを用いて、変換精度を評価する実験を行った。同一ドメインの入力に対しては既存 IME や LLM と同等以上の変換精度を示し、一方で異なるドメイン間においては知識量や文体の違いが変換性能に影響することがわかった。複数の Core モデルを併用するアプローチにおいては、知識量と文体においてバランスよく対応した「一般ドメイン」の変換と、特定の分野や使用目的に特化した変換を組み合わせるのが効果的であることが示唆された。

今後は、複数の Core モデルの併用手法の具体化や、Core モデルの学習の効率化に取り組む必要がある。また BlindX のシステムとしては、周辺機能やアプリケーションの実装を行い、有用性を評価する。

## 参考文献

- [1] Yoh Okuno. Neural ime: Neural input method engine. In *The 8th Input Method Workshop*, 2016.
- [2] Armin Sarhangzadeh and Taro Watanabe. Alignment-based decoding policy for low-latency and anticipation-free neural Japanese input method editors. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 8043–8054, August 2024.
- [3] 三輪敬太, 高橋直希. ニューラルかな漢字変換システム Zenzai. 言語処理学会 第 31 回年次大会, pp. 271–276, 2025.
- [4] 森信介, 土屋雅稔, 山地治, 長尾真. 確率的モデルによる仮名漢字変換. 情報処理学会論文誌, Vol. 40, No. 7, pp. 2964–2953, 1999.
- [5] 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.
- [6] Jianfeng Gao, Hisami Suzuki, and Bin Yu. Approximation lasso methods for language modeling. In Nicoletta Calzolari, Claire Cardie, and Pierre Isabelle, editors, *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pp. 225–232. Association for Computational Linguistics, July 2006.
- [7] 徳永拓之, 岡野原大輔. 日本語かな漢字変換における識別モデルの適用とその考察. 言語処理学会 第 17 回年次大会, pp. 959–962, 2011.
- [8] Ashish Vaswani et al. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, Vol. 30, 2017.
- [9] Colin Raffel et al. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, Vol. 21, No. 1, January 2020.
- [10] 稲葉通将. おーぶん 2 ちゃんねる対話コーパスを用いた用例ベース対話システム. 第 87 回言語・音声理解と対話処理研究会 (第 10 回対話システムシンポジウム), 人工知能学会研究会資料 SIG-SLUD-B902-33, pp. 129–132, 2019.
- [11] LINE ヤフー株式会社. Yahoo! 知恵袋データ (第 3 版), 2024. 国立情報学研究所情報学研究データリポジトリ. (データセット). <https://doi.org/10.32130/idr.1.3>.
- [12] Applying conditional random fields to Japanese morphological analysis.
- [13] Sato Toshinori. Neologism dictionary based on the language resources on the web for mecab, 2015.
- [14] 佐藤敏紀, 橋本泰一, 奥村学. 単語分かち書き用辞書生成システム neologd の運用 — 文書分類を例にして —. 自然言語処理研究会研究報告, NL-229-15. 情報処理学会, 2016.
- [15] 佐藤敏紀, 橋本泰一, 奥村学. 単語分かち書き辞書 mecab-ipadic-neologd の実装と情報検索における効果的な使用方法の検討. 言語処理学会第 23 回年次大会 (NLP2017), NLP2017-B6-1. 言語処理学会, 2017.
- [16] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, ACL '02*, p. 311–318. Association for Computational Linguistics, 2002.
- [17] Enneng Yang et al. Model merging in llms, mllms, and beyond: Methods, theories, applications and opportunities, 2024.
- [18] Mitchell Wortsman et al. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time, 2022.
- [19] Edward J. Hu et al. Lora: Low-rank adaptation of large language models, 2021.