Comprehensive Evaluation on Lexical Normalization: Boundary-Aware Approaches for Unsegmented Languages

Anonymous ACL submission

Abstract

Lexical normalization research has sought to tackle the challenge of processing informal expressions in user-generated text, yet the absence of comprehensive evaluations leaves it unclear which methods excel across multiple perspectives. Focusing on unsegmented languages, we make three key contributions: (1) creating a large-scale, multi-domain Japanese normalization dataset, (2) developing normalization methods based on state-of-the-art pretrained models, and (3) conducting experiments across multiple evaluation perspectives. Our experiments show that both encoder-only and decoder-only approaches achieve promising results in both accuracy and efficiency.

1 Introduction

User-generated text (UGT) is invaluable textual content produced by users' activities on web platforms such as review sites and social media. UGT's informality—its frequent use of colloquial expressions—poses substantial challenges to accurate analysis in natural language processing applications. Consequently, researchers have explored lexical normalization (LN), the task of converting non-standard word forms into standard ones. LN has been actively studied particularly for space-delimited languages such as European languages (Baldwin et al., 2015; van der Goot et al., 2021). In this study, we investigate LN for *unsegmented languages*, focusing primarily on Japanese.

Major issues in existing LN research are summarized as a lack of comprehensive evaluation, leaving unclear which methods excel under different evaluation aspects. Specifically, (i) comparative evaluations of recent model architectures are absent, and (ii) multi-perspective analyses—examining required training data size, inference cost, and domain-specific accuracy across diverse domains—have not been conducted. These issues are common to LN research but are particularly severe for underexplored unsegmented languages. This study addresses these gaps through three key contributions: (1) dataset construction, (2) method development based on cutting-edge pre-trained models, and (3) comprehensive experiments.

First, we introduce the Japanese Multi-Domain Lexical Normalization Dataset (JMLN), a large collection of 21,402 sentences drawn from a variety of UGT sources. JMLN's size exceeds existing Japanese LN datasets (Higashiyama et al., 2021b; Kondo et al., 2025), and its domain diversity surpasses that of any current LN datasets. These properties enable both the development of methods suitable for Japanese and multi-perspective evaluation.

Second, we develop LN methods based on three modern Transformer (Vaswani et al., 2017) architectures—encoder-only, encoder-decoder, and decoder-only—including a novel encoder-based infilling approach, as well as variants of generative approaches. While these *boundary-aware* methods are tailored for unsegmented languages, they remain broadly applicable.

Third, we evaluate these methods on JMLN and an existing Thai dataset. Multi-perspective experiments on JMLN yield in-depth insights into methods' characteristics and trade-offs, while experiments on the Thai dataset further validate cross-lingual applicability and generalizability.

Our evaluation reveals three main findings. First, compact encoder-only models deliver the highest inference throughput, and decoder-only models excel in normalization recall—while both yield high normalization precision. Second, training models on 4k–8k sentences yields reasonable precision of around 0.7, and cutting-edge decoder-only models deliver superior recall even with fewer instances. Third, domains rich with unknown informal words exhibit low performance, especially the typo-correction domain.[1]

[1]Our dataset and code will be available at {URL}.

2 Related Work

Text normalization has been studied for some representative purposes: mapping dialectal variants to standard language (Kuparinen et al., 2023), modernizing historical writings (Bollmann, 2019), and verbalizing semiotic expressions for text-to-speech (TTS) synthesis (Zhang et al., 2019). Specifically, lexical normalization (LN) refers to the normalization task of converting UGT at the lexical level.

To date, research on UGT normalization can be broadly divided into three categories based on the dominant methodologies of each period: rule-based and statistical methods (Aw et al., 2006; Choudhury et al., 2007; Han and Baldwin, 2011), pre-Transformer neural methods (Chrupała, 2014; Ikeda et al., 2016; Lusetti et al., 2018; Lourentzou et al., 2019), and Transformer-based methods (Muller et al., 2019; Samuel and Straka, 2021; Bucur et al., 2021; Bikaun et al., 2024).

To achieve robust word segmentation (WS) for UGT, previous studies have often addressed LN jointly with WS, as exemplified by research on Japanese (Sasano et al., 2013; Kaji and Kitsuregawa, 2014; Saito et al., 2014, 2017; Higashiyama et al., 2021a), Chinese (Wang et al., 2013; Qian et al., 2015), and Thai (Haruechaiyasak and Kongthon, 2013). Some recent studies have adopted Transformer masked language models (MLMs) (Ueda et al., 2023; Pankam et al., 2023).

Decoder-only Transformer models have recently made remarkable advances and have been applied to text normalization for TTS (Zhang et al., 2024b; Shen et al., 2024) and other sequence transduction tasks (Kaneko and Okazaki, 2023; Shi et al., 2024). However, these models remain underexplored in UGT normalization, resulting in a lack of comparative evaluation of state-of-the-art models for this task.

Regarding evaluation domains, many studies have focused on building short message and social media datasets for European (Choudhury et al., 2007; Han and Baldwin, 2011; Baldwin et al., 2015; Plank et al., 2020; van der Goot et al., 2021) and Asian languages (Kaji and Kitsuregawa, 2014; Limkonchotiwat et al., 2021; Nguyen et al., 2024; Kondo et al., 2025), which has led to extensive model development and evaluation in these domains. Some studies have focused on other domains (Higashiyama et al., 2021b; Bikaun et al., 2024). However, cross-domain evaluation research covering three or more domains remains scarce.

3 Japanese Dataset Construction

For our primary target language, Japanese, existing datasets (Kaji and Kitsuregawa, 2014; Osaki et al., 2017; Higashiyama et al., 2021b; Kondo et al., 2025) have limited domain diversity and size—covering one or two domains with approximately 1,000 to 6,000 sentences/posts. In this study, we have constructed JMLN, as a large-scale dataset sourced from a variety of UGT.

Data Sources and Size We sampled original texts from various sources: Q&A site, blog site, review site, recipe site, video site, online forum, and social media platform, as well as Wikipedia edit history and conversation transcriptions (details in Appendix A.2). The constructed dataset includes 21,402 sentences with 8,885 normalization instances, i.e., non-standard and standard form pairs (details in Appendix A.1). The large data size and domain diversity of our dataset are advantages over existing Japanese datasets, enabling multi-perspective evaluations, as shown in §6.

Basic Designs We followed Higashiyama et al. (2021b)'s annotation criteria; annotation information includes word boundaries based on the short unit word criterion (Maekawa et al., 2014) and word attributes such as part-of-speech, lemma, predefined word categories (details in Appendix A.3), and standard forms of non-standard words.[2]

Annotation Process As data preparation, the first author extracted sentences with a reasonable length (10–300 characters) from each original 14 datasets and divided the sentences into two candidate sets: a random (Rand) set and a manually curated (Cur) set. Then, four experienced annotators, including an annotator manager, at a data annotation company performed the annotation process as follows.

- Sentence selection: From the Rand-set of each original dataset, annotators sequentially selected sentences unless they contained ethically problematic contents or were unclear in meaning. From each Cur-set, annotators intentionally selected sentences containing UGT-specific category words (Appendix A.3).
- Word information annotation: Annotators annotated the selected sentences with word information by modifying auto-analyzed results

[2]Non-standard forms are those with distinctive orthographic features whose frequency in the reference corpus falls below a threshold, whereas standard forms are those whose frequency exceeds a threshold. See details in Appendix A.4.

by a morphological analyzer, MeCab (Kudo et al., 2004) with UniDic (Den, 2009).[3]

- Standard form annotation: We adopted the separated steps for standard form (SForm) annotation—assigning an SForm ID for each non-standard word and associating a set of valid standard forms with each SForm ID, where the later step for all sentences was performed by the annotation manager.

Inter-Annotator Agreement During the annotation process, manager A selected a total of 240 sentences from Cur-sets, and then two annotators—B and either C or D—independently annotated those sentences with boundaries and attributes. Inter-annotator agreement (IAA) for non-standard word recognition on these sentences, as measured by F_1 score, was 0.836 (See Appendix A.6).[4] The datasets' high annotation consistency and large size suggest its usefulness, and this is further demonstrated by the experiments in §6, where the evaluated models achieve high normalization accuracy.

4 Task Definition

Following previous studies (Sasano et al., 2013; Baldwin et al., 2015), we define LN as a task of *boundary-aware* span extraction and conversion, in which a system not only generates a normalized text but also identifies the original spans of each informal words (or phrases). This task can be performed on any LN dataset, provided that informal-to-formal alignments are annotated.[5] Unlike the text-to-text conversion task (Ikeda et al., 2016), the boundary-aware task enables fine-grained evaluation at the normalization span level and offers better interpretability of system outputs. An example input-output pair for our task is shown in Figure 1.

Formally, an LN system takes as input a source sentence, namely, a sequence of n character (or subword) tokens $\boldsymbol{x} = \boldsymbol{x}_{0:n} = [x_0, \ldots, x_{n-1}]$, and is required to predict the set of non-standard word spans and their standard forms $S = \{(b, e, s)\}$. Here, (b, e) $(0 \leq b \leq e \leq n)$ indicates a span of an non-standard word $x_{b:e}$ with length $e - b$ in the source sentence, and s indicates its standard form. Each standard form s is a string with length ≥ 0,

[3] cwj-3.1.0 (https://clrd.ninjal.ac.jp/unidic/)

[4] Plank et al. (2020) and van der Goot et al. (2020) used Cohen's kappa to evaluate IAA in informal word classification, based on whether each given word is normalized or not. This metric is not directly applicable to unsegmented languages.

[5] Otherwise, a non-trivial alignment processing is needed. We provide an alignment examples in Appendix A.5.

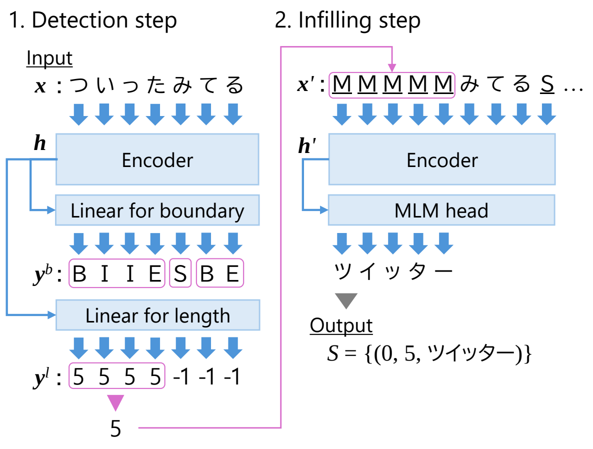

Figure 1: Flow of our detect&infill approach for an input text "ついったみてる," which means "(I'm) looking at Twitter." "M" and "S" represent the MASK and SEP token, respectively. The original characters "ついった" follow the SEP token, but are omitted in the Figure.

where length = 0 indicates that the non-standard word should be deleted in the normalized sentence. When $b = e$, a zero-length span indicates some token(s) should be inserted into the position b.

5 Methods

We present boundary-aware LN methods based on three Transformer architectures: encoder-only, encoder-decoder, and decoder-only. Our encoder-based infilling approach is a novel method for unsegmented languages, and comparing multiple approaches across different architectures offers a valuable, novel evaluation.

5.1 Infilling Approach

Among encoder-based methods, including text editing (Ueda et al., 2023) and MLM infilling (Muller et al., 2019), the latter directly leverage the capabilities of pretrained MLMs to insert any token from the vocabulary. As a representative study, Muller et al. (2019) proposed a two-step approach for space-delimited text, which predicts the infilling lengths and infilling tokens from subword-tokenized text. While we follow the two-step *detect-and-infill* framework, we propose a solution tailored to unsegmented languages. Our method jointly predicts word boundaries and infilling lengths from the input character sequence, thereby identifying non-standard word spans and their corresponding normalized spans, without the explicit alignment step (Muller et al., 2019).

As shown in Figure 1, the detailed workflow of our approach is as follows. In the *detection* step, the encoder takes as input a character token sequence $\boldsymbol{x}_{0:n}$ and output the sequence of token hidden representations $\boldsymbol{h}_{0:n}$. Then, a linear layer for boundary prediction and that for length prediction predict a chunk boundary tag sequence $\boldsymbol{y}^b_{0:n}$ and a length value sequence $\boldsymbol{y}^l_{0:n}$, respectively. A series of boundary tags[6] (e.g., $y^b_{0:4} = [\texttt{B}, \texttt{I}, \texttt{I}, \texttt{E}]$) identifies a chunk corresponding a standard or non-standard word. A positive length value (e.g., "5") for the specified non-standard word chunk (e.g., $x_{0:4}$) indicates the numbers of tokens comprising standard forms that should be filled in the later step, the value "0" indicates that the chunk should be removed, and the value "-1" indicates that the chunk has no need for normalization. Notably, we estimate a length value per character token, so an non-standard word chunk of m characters yields m redundant length values (e.g., $y^l_{0:4} = [5, 5, 5, 5]$). We determine a single length value by taking a majority vote within the chunk. Thus, the combination of two sequences specifies non-standard word spans and the lengths of their standard forms.

In the *infilling* step, the encoder takes as input the source text $\boldsymbol{x}'$, in which tokens in non-standard word spans are replaced by the `MASK` tokens, and the MLM head predicts appropriate tokens for the masked positions from the hidden representations $\boldsymbol{h}'$. We apply input extension to the masked source text by concatenating the original characters of the specified non-standard tokens, with `SEP` tokens inserted between them, similarly to existing sequence transduction methods (Qiang et al., 2021; He et al., 2023).

We refer to the above method as the FULL-SEG approach.[7] During training, the model is optimized using the sum of cross-entropy losses over multiple subtasks from both steps.

5.2 Generative Approaches

Encoder-decoder models have been extensively used in normalization research for text-to-text conversion, which we refer to the *plain full-text* (PLAIN) approach. To eliminate the informal-to-formal alignment step required for this approach, Bikaun et al. (2024) generates outputs in which non-standard words and their corresponding normalized forms are each surrounded by distinct special tokens. We introduce two generative approaches for encoder–decoder and decoder-only models, one of which—STRUCT—is essentially equivalent to Bikaun et al. (2024)'s method. Because few studies compare multiple generative approaches to LN—and none include decoder-only models—it is valuable to evaluate and contrast these methods across both architectures.

[6]We adopt the BIESO tagging schema in our experiments.

[7]We report preliminary experiments on additional variants in the Appendix B

Approach	Target Text
PLAIN	ツイッターみてる
STRUCT	[[ついった>>ツイッター]]みてる
SPAN	ついった>>ツイッター>>0

Table 1: Expected output text of each generative approach for an input text "ついったみてる."

As shown in Table 1, the *structured full-text* (STRUCT) approach generates a full normalized text with specifying the substrings before and after normalization and their spans, using symbols "`[[`," "`>>`," and "`]]`." The other *normalization span-only* (SPAN) approach generates not full-text but only substrings before and after normalization using a symbol "`>>`." The number (≥ 0) succeeding the normalized substring represents how many times the same original substring occurred before that of interest in the original text, which is used for specifying the exact span of the original substring. A symbol "`||`" is used to separate multiple normalization instances within the input text. Only "`NONE`" is output when no normalization is necessary.

Encoder-Decoder The model is trained to generate the target text from each input source text via the standard sequence-to-sequence training.

Decoder-only The model takes as input a prompt with an instruction and a source text, like "Instruction: \n`{inst}`\n\nInput: \n`{src}`\n\nOutput: \n", and is trained to generate the target text "`{tgt}` `EOS`" via the standard instruction tuning. Here, `{inst}`, `{src}`, and `{tgt}` are placeholders, and `EOS` represents the end of text token. We use English instruction texts explaining the corresponding content for STRUCT and SPAN as described above; the exact wording is provided in Appendix C.4.

6 Experiments

We set the following experimental questions (EQs):

1. Across different model architectures, backbone models, and normalization approaches, which methods excel in normalization accuracy (precision and recall), and which methods are efficient in terms of inference cost?

Lang	Dataset	Set	#Sent	#Norm
ja	JMLN	train	13,196	5,879
		dev	1,880	791
		test-C	3,786	1,705
		test-R	2,540	510
th	VISTEC-2021	train	40,000	130,790
		test	10,000	32,819

Table 2: Dataset statistics: The number of sentences (#Sent) and normalization instances (#Norm).

2. How many training instances are required to achieve reasonable performance, and does this requirement vary by methods?
3. What domains and other instance characteristics are particularly challenging to normalize?

To address these EQs, we evaluated various LN methods using our Japanese dataset and an existing Thai dataset. Specifically, experiments in §7.1–7.3, §7.4, and §7.5–7.6 correspond to EQ1, EQ2, and EQ3, respectively.

As evaluation metrics, we use precision and recall at the normalization-span level for individual non-standard words, as well as the $F_{0.5}$ score, which emphasizes precision over recall (see details in Appendix C.1).

6.1 Datasets

Table 2 shows the statistics of two experimental datasets: JMNL and Thai VISTEC-2021 (Limkonchotiwat et al., 2021).[8] For JMLN, we divided the Cur-set for each domain (01–14) into train, dev, and test-C sets, and merged the all train, dev, and test-C sets into unified train, dev, and test-C sets across all domains, respectively.[9] For VISTEC-2021, we followed the provided training/test split and regarded randomly-sampled 5% sentences in the training set as a dev set.

6.2 Models

As the backbone of LN systems, we used the following Japanese or multilingual pre-trained models in Japanese experiments: BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), and DeBERTa (He et al., 2021) as character-level encoder-only models, T5 (Raffel et al., 2020) and mT5 (Xue et al., 2021) as encoder-decoder models, and Llama 3.1/3.2 (Grattafiori et al., 2024), Qwen2.5 (Qwen Team, 2025), Lllama-3.1-Swallow (Fujii et al., 2024), TinySwallow (Shing et al., 2025) and Sarashina2/2.2 (Intuitions, 2024a,b) as decoder-only models. Similarly, we used the following pre-trained models in the Thai experiments: RoBERTa (Yasuoka, 2023) as a character-level encoder-only model, T5 and mT5 as encoder-decoder models, and Llama 3.1/3.2, Qwen2.5, Typhoon 2 (Pipatanakul et al., 2024), OpenThaiGPT 1.5 (Yuenyong et al., 2025), and SeaLLMs 3 (Zhang et al., 2024a) as decoder-only models. Specific model instances are listed in Appendix C.2.

We fine-tuned each model, with applying LoRA (Hu et al., 2022) or QLoRA (Dettmers et al., 2023) to some decoder-only models, and selected the model checkpoint with the best $F_{0.5}$ score on the dev set. We fine-tuned all models twice and report mean scores for two runs. The hyperparameter settings are listed in Appendix C.3.

7 Results and Analysis

7.1 Normalization Accuracy for Japanese

We evaluated LN methods with three type of architectures on the JMLN test-C set. Table 3 shows the performance of encoder-only models, and encoder-decoder and decoder-only models with both STRUCT and SPAN approaches.[10]

The observed results are as follows. (1) Among encoder-only models, the large models outperformed base models, while different backbone models showed similar performance. (2) Among generative methods, the SPAN approach basically achieved performance comparable to or better than the STRUCT approach in many cases.[11] (3) Within the same model series—T5, mT5, Llama-3.2, Qwen2.5, and Sarashina2.2—performance improved with increasing model size up to 8B (See Figure in Appendix D.1). However, in our preliminary experiment on the dev set, we observed no salient additional gains from the 13B–15B models. (4) Performance within groups of similarly sized models was not equivalent; certain series—specifically Sarashina2.2 series, followed by Swallow, demonstrated saliently superior performance. (5) These strong decoder-only models outperformed models with the other two architec-

[8] `https://github.com/mrpeerat/OSKut`

[9] We created the unified Rand-set across all registers similarly, but omit experimental results on this set.

[10] The PLAIN approach yielded performance similar to the other two approaches on the dev set, as shown in §D.3.

[11] The only exception is the Qwen2.5 model series; the models generated many nonsensical text flagments—such as "str1>>str2>>0," where "str1" did not appear in the original input text—until reaching the maximum output length, resulted in low precision and long inference time.

	Backbone	Size	FULL-SEG Approach P	FULL-SEG Approach R	FULL-SEG Approach $F_{0.5}$	STRUCT Approach P	STRUCT Approach R	STRUCT Approach $F_{0.5}$	SPAN Approach P	SPAN Approach R	SPAN Approach $F_{0.5}$
E	BERT-base	91M	0.713	**0.529**	0.667	–	–	–	–	–	–
E	RoBERTa-base	100M	0.718	0.523	0.669	–	–	–	–	–	–
E	DeBERTa-base	100M	**0.729**	0.506	**0.670**	–	–	–	–	–	–
E	BERT-large	310M	**0.762**	**0.570**	**0.714**	–	–	–	–	–	–
E	RoBERTa-large	320M	0.755	0.550	0.702	–	–	–	–	–	–
E	DeBERTa-large	330M	0.750	0.568	0.705	–	–	–	–	–	–
S	T5-base	250M	–	–	–	0.701	0.459	0.634	0.689	0.508	0.643
S	mT5-base	580M	–	–	–	0.606	0.406	0.551	0.655	0.427	0.592
S	T5-large	780M	–	–	–	0.728	0.509	0.670	0.704	0.525	0.659
S	mT5-large	1.2B	–	–	–	0.600	0.421	0.553	0.718	0.467	0.648
D	Llama-3.2-1B	1.2B	–	–	–	0.654	0.489	0.612	0.626	0.480	0.590
D	Sarashina2.2-1B	1.4B	–	–	–	0.722	**0.612**	0.697	**0.761**	0.585	**0.717**
D	Qwen2.5-1.5B	1.5B	–	–	–	0.623	0.459	0.580	0.338	0.516	0.363
D	TinySwallow-1.5B	1.5B	–	–	–	0.605	0.556	0.593	0.667	0.569	0.645
D	Qwen2.5-3B	3.1B	–	–	–	0.598	0.537	0.583	0.376	0.537	0.400
D	Llama-3.2-3B	3.2B	–	–	–	0.680	0.522	0.641	0.666	0.530	0.633
D	Sarashina2.2-3B	3.4B	–	–	–	0.774	**0.668**	0.751	**0.781**	0.660	**0.754**
D	Sarashina2-7B	7.3B	–	–	–	0.743	0.649	0.722	0.744	**0.657**	**0.724**
D	Qwen2.5-7B	7.6B	–	–	–	0.717	0.558	0.678	0.379	0.589	0.408
D	Llama-3.1-8B	8.0B	–	–	–	0.738	0.538	0.687	0.719	0.549	0.677
D	↪Swallow-8B	8.0B	–	–	–	**0.749**	0.605	0.715	0.741	0.602	0.708

Table 3: JMLN test results of Japanese LN models (E: encoder, S: seq2seq, D: decoder). "↪" indicates the continual pre-trained model derived from the base model listed in the previous row. The best score within each size group is shown in **bold**. For each backbone model, the better of the STRUCT and SPAN approaches is underlined. Scores where the SPAN approach shows a +5%, +10%, or -10% increase/decrease compared to the STRUCT approach are highlighted with 🟦 blue, ◻ light blue, and 🟪 pink backgrounds, respectively.

tures in recall.

In conclusion, encoder-only models demonstrated high performance despite its small size, and Sarashina2.2-3B model achieved the highest performance overall, indicating that the high capability of the backbone model was beneficial for this task.

7.2 Normalization Accuracy for Thai

On the Thai VISTEC test set, we evaluated LN methods, with only the SPAN approach for generative methods. Table 4 shows the results.

Similarly to the Japanese results, performance improved with increasing model size within the same model series. Additionally, continually pre-trained models focusing on Thai outperformed their base models. Overall, the small encoder-only RoBERTa-base exhibited the best precision, while all encoder-decoder and decoder-only models surpass it in recall. This introduces a precision-recall trade-off in model selection.

7.3 Inference Throughput

For selected models with high normalization accuracy, we measured their inference throughput using the JMLN test set, on both an NVIDIA V100 GPU with 32 GiB memory and an H200 GPU with 140 GiB memory (See detailed settings in Appendix C.6). Table 5 shows the results.

	Backbone	Size	P	R	$F_{0.5}$
E	RoBERTa-base	88M	**0.713**	**0.529**	**0.666**
S	T5-base	250M	**0.645**	0.618	**0.640**
S	mT5-base	580M	0.642	**0.629**	0.639
S	mT5-large	1.2B	**0.660**	0.609	**0.649**
D	Llama-3.2	1.2B	0.628	0.628	0.628
D	↪ Typhoon2	1.2B	0.644	0.634	0.642
D	SeaLLMs3	1.5B	0.462	0.689	0.495
D	Qwen2.5	1.5B	0.472	**0.702**	0.505
D	Qwen2.5	3.1B	0.457	**0.706**	0.492
D	Llama-3.2	3.2B	0.641	0.647	0.642
D	↪ Typhoon2	3.2B	**0.656**	0.668	**0.658**
D	SeaLLMs3	7.6B	0.465	0.705	0.499
D	Qwen2.5	7.6B	0.461	**0.709**	0.496
D	↪ ThaiGPT1.5	7.6B	0.653	0.672	0.657
D	Llama-3.1	8.0B	0.653	0.659	0.655
D	↪ Typhoon2	8.0B	**0.661**	0.678	**0.664**

Table 4: VISTEC test results of Thai LN models.

We observed: (1) Encoder-only models were the fastest, followed by T5, the smaller Sarashina model, and finally the larger Sarashina model. (2) The SPAN approach yielded modest gains over STRUCT, except for T5 on the H200. Since instruction text occupies a large proportion of total out-

Model	V100	H200	V100	H200
	FULL-SEG			
BERT-large	508.8	1420.9	–	–
RoBERTa-large	561.4	1407.2	–	–
DeBERTa-large	395.0	1038.3	–	–
	STRUCT		SPAN	
T5-large	136.0	312.3	159.0	208.4
Sarashina2.2-1B	66.5	202.1	75.4	243.1
Sarashina2.2-3B	32.6	118.1	36.0	117.4
Sarashina2-7B	17.8	73.3	19.8	83.4

Table 5: Throughput: the number of sentences processed per second, measured on a V100 and H200 GPU.

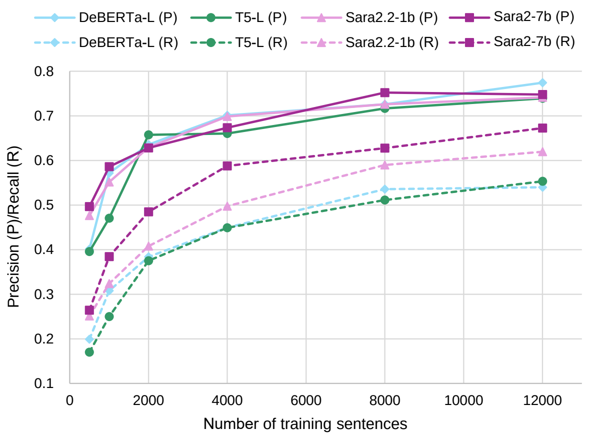

Figure 2: JMLN test results for each training data size.

put tokens, more concise instruction prompts can improve throughput; however, it is necessary to explore prompts that preserve normalization accuracy. (3) Sarashina models exhibited substantially lower throughput on the V100 than on the H200. Their low throughput on the V100 is a critical drawback, but they run much faster on the H200. Thus, when high-spec GPUs are available, Sarashina models are viable options in accuracy-critical scenarios.

7.4 Investigation of Training Data Size

We generated size-N training sets by sampling random $N \in \{500, 1k, 2k, 4k, 8k, 12k\}$ sentences from the entire JMLN training set, and we then fine-tuned each of DeBERTa, T5, and Sarashina models twice for each size-N training set. As shown in Figure 2, the results are as follows.

First, a general trend across all models is that precision and recall improve as the data size increases. From sizes 8k to 12k the gains are more gradual, but performance is not yet saturated. Within the evaluated range, more data yields better results; however, even a 4k to 8k-size dataset can achieve reasonable precision around 0.70 when creating large amounts of annotated data is impractical.

Second, in model-specific comparisons, precision and recall follow different patterns. Precision shows no clear differences across models. In contrast, recall is consistently highest for Sarashina2-7b, followed by Sarashina2.2-1b, and lower for DeBERTa and T5; this indicates that the Sarashina-series models generalize well in terms of coverage, even when the training data size is small.

7.5 Results Across Domains

As shown in Table 6, we evaluated performance of selected models—DeBERTa-large, T5-large (SPAN), and Sarashina-2.2-3B (SPAN)—for each domain test set of JMLN.

First, to explore what makes a domain difficult, we examined an indicator: the proportion of non-standard surface tokens in the test set that are not found among the training set's non-standard surfaces (Surf-Outside-Train rate). We then computed the Pearson correlation coefficients r between the indicator and the average $F_{0.5}$ scores across the three models, obtaining a strong negative correlation ($r = -0.78$). Notably, for 9 out of 10 domains with a Surf-Outside-Train rate below 0.5 had average $F_{0.5}$ scores above 0.7, whereas all 4 domains with a rate above 0.5 had average scores below 0.7.

Next, model performance comparisons revealed the following. (1) In all domains, all models exhibited higher precision than recall, showing a desirable characteristic because invalid normalizations would degrade downstream task performance. (2) Across most domains, Sarashina-2.2 achieved higher recall than the other models, resulting its superior overall performance. Notably, this model achieved recall over 0.5 across typical UGT domains (01–11). (3) All models exhibited very low recall below 0.3 in domain 11—a specialized domain data originated from Tanaka et al. (2020)'s typo correction dataset. Training with the task-specific dataset would improve performance, but we leave this for future.

7.6 Error Analysis

For the three models evaluated in §7.5, we analyzed output patterns on the JMLN dev set. Specifically, we counted predictions, true positives (TP), false positives (FP), and false negatives (FN) for each model, and measured (i) the proportion of predicted normalized forms appearing in the UniDic lexicon (Norm-In-Lex rate) and (ii) the proportion of origi-

Data	Register	Surf-Out-Train	Avg-3M $F_{0.5}$	DeBERTa-L P	DeBERTa-L R	T5-L P	T5-L R	Sarashina2.2-3B P	Sarashina2.2-3B R
01 `BJ-OC`	Q&A site	0.59	0.595	0.654	0.416	0.572	0.369	**0.717**	**0.578**
02 `BJ-OY`	Blog	0.48	0.713	0.720	0.543	0.741	0.551	**0.802**	**0.662**
03 `RC-BLG`	Blog	0.37	0.775	0.801	0.691	0.793	0.664	**0.803**	**0.728**
04 `RC-REV`	Reviews	0.43	0.790	**0.868**	0.731	0.790	0.593	0.808	**0.741**
05 `RK-ICB`	Reviews	0.52	0.680	**0.808**	0.458	0.693	0.500	0.764	**0.521**
06 `RK-TRV`	Reviews	0.40	0.715	0.763	0.490	**0.820**	0.433	0.795	**0.663**
07 `RK-RCP`	Recipes	0.35	0.792	0.815	0.684	0.817	0.663	**0.834**	**0.734**
08 `AM`	Reviews	0.32	0.798	**0.862**	0.728	0.776	0.655	0.832	**0.748**
09 `NC-VID`	Video desc.	0.39	0.645	0.627	0.538	**0.783**	**0.677**	0.730	0.629
10 `NC-PED`	Forum	0.27	0.795	**0.850**	0.747	0.666	0.492	0.808	**0.772**
11 `TW`	Social media	0.60	0.633	0.613	0.511	0.600	0.464	**0.763**	**0.677**
12 `JW`	Wiki hist.	0.69	0.312	**0.528**	0.140	0.197	0.095	0.509	**0.280**
13 `NU`	Conv. trans.	0.17	0.758	**0.845**	**0.745**	0.683	0.596	0.800	**0.745**
14 `SK`	Conv. trans.	0.43	0.718	0.771	0.550	0.728	0.457	**0.830**	**0.664**
All		0.44	0.721	0.750	0.568	0.754	0.562	**0.781**	**0.660**

Table 6: JMLN test results of representative three models for each domain. "Surf-Out-Train" indicates the Surf-Outside-Train rate for each domain test set. Avg-3M indicates the average of $F_{0.5}$ scores of the three models. (Surf-Outiside-Train rate values above 0.5 and Avg-3M values below 0.7 are highlighted by underlining.)

	DeBERTa	T5	Sarashina2.2
#Predictions	614.5	598.5	676.5
Norm-In-Lex rate	0.875	0.928	0.921
#TPs	469.0	449.0	535.5
Surf-In-Train rate	0.806	0.806	0.716
#FPs	145.5	149.5	141.0
Surf-In-Train rate	0.289	0.183	0.238
#FNs	322.0	342.0	255.5
Surf-In-Train rate	0.258	0.290	0.302

Table 7: Models' prediction statistics on JLMN dev set (DeBERTa: large, T5: large, Sarashina: 3B).

nal surface forms for TP, FP, and FN instances that matched any non-standard forms in the training set (Surf-In-Train rate). Table 7 shows the results averaged over two runs per model.

DeBERTa exhibited a notably lower Norm-In-Lex rate than the other models (0.921–0.928) and the gold standard (0.947). By manually inspecting error cases, we found that DeBERTa's restored tokens within spans sometimes formed nonsensical words.[12] Both suggest that the model's independent prediction at each `MASK` position makes it especially prone to such errors.

Regarding the Surf-In-Train rate, all models exhibited similar trends. For TPs, approximately 70–80% of the original surface forms were known (i.e., appeared in the training set), indicating that many correct predictions relied on the seen normalization instances. For both FPs and FNs, only approximately 20–30% were known, indicating that the majority of errors involved unseen expressions. This suggests considerable room for improving generalization in normalizing unseen cases. These results align with the findings in §7.5.

8 Conclusion

This paper presented our multi-domain Japanese LN dataset, LN methods based on three Transformer architectures for unsegmented languages, and multi-perspective experiments and analysis.

The answers to the three evaluation questions (§6) are summarized as follows. (1) Compact encoder-only models achieved high precision and offered the best inference throughput, while cutting-edge decoder-only models delivered high precision, notably high recall, and reasonable throughput on a high-spec GPU. (2) Normalization accuracy consistently increased with training data size, yet even 4k–8k training sentences yielded reasonable precision around 0.7. Sarashina-series models, in particular, achieved superior recall with fewer training sentences. (3) Domains with higher rates of unknown non-standard tokens correlated with decreased performance across models. Typo correction emerged as the most challenging category, reflecting the difficulty posed by diverse typo patterns.

In future work, we will evaluate the impact of LN on downstream tasks and explore the development of a general-purpose decoder model with robust normalization capabilities.

[12] E.g., かど (orig. ヶド, gold けど/けれど, "but") and だじりん (orig. だぁりん, gold ダーリン, "darling").

Limitations

Dataset Size The experimental results in §7.1—high precision up to 0.78—indicates that our dataset is large enough to train high-accuracy models. However, results in §7.4 show no clear saturation even at the maximum training size, suggesting that additional data could further improve performance. Given the cost limits of manual annotation, a promising direction is to explore methods for generating high-quality synthetic data.

Language Coverage We evaluated our methods only on Japanese and Thai datasets, but they are readily applicable to other unsegmented and space-delimited languages. Validation on additional languages remains future work.

Inference Throughput Settings To ensure fair comparison, we measured throughput using a single GPU via the Hugging Face Transformers (Wolf et al., 2020) library. However, throughput could be improved through multi-GPU parallelism, model quantization, or adoption of high-performance inference engine, such as vLLM (Kwon et al., 2023).

Encoder-Only Architecture Variants An state-of-the-art encoder-only model, ModernBERT (Warner et al., 2024), might achieve performance on par with or exceeding the models we evaluated. Due to computational and time constraints, this remains future investigation.

Word-Level Evaluation Metrics Our word-level normalization metrics treat any span mismatch as an error—even if the predicted normalization is semantically valid—which we observed especially in decoder-only model outputs (see Appendix D.5 for examples). Such span differences have little impact on most downstream applications, so the practical usefulness of the outputs may exceed the scores reported. To complement word-level metrics, we also provide sentence-level exact-match accuracy and the chrF score (Popović, 2015) in Appendix D.2 and D.2.

Ethics Statement

License of Resources MeCab is available under GPL, LGPL, and BSD License. UniDic ("unidic-cwj-3.1.0") is available under GPL v2.0, LGPL v2.0, and New BSD License. sacreBLEU is available under Apache License 2.0 (we used this software for preliminary experiments shown in Appendix D). The licenses for the datasets and pre-trained models are listed in Appendices C.2 and A.2 (Table 19). Our use of these resources for academic research aligns with their intended use. We will release our JMLN dataset for academic research in information science; it will include only annotation information and not the original texts.

Human Annotators The annotation work was performed by annotators at a professional data annotation company. The payment amount to the company was based on the estimate submitted by the company. The actual annotators and the payment amount to each annotator were determined by the company. The annotation work was performed by four annotators, including an annotation manager, all of whom are native Japanese speakers. Under the contract for the annotation work, it was agreed that the intellectual property rights to the deliverables would be transferred to the authors' institution.

Potential Risks Appropriate normalization can facilitate NLP applications while preserving the core meaning of the original texts. However, it may diminish subtle nuances and intentions in the original text; for example, casual expressions may be rendered formal, dialectal expressions may be replaced by semantically similar standard language forms, or incorrect normalization may produce an entirely different meaning.

References

AiTi Aw, Min Zhang, Juan Xiao, and Jian Su. 2006. A phrase-based statistical model for SMS text normalization. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 33–40, Sydney, Australia. Association for Computational Linguistics.

Timothy Baldwin, Marie Catherine de Marneffe, Bo Han, Young-Bum Kim, Alan Ritter, and Wei Xu. 2015. Shared tasks of the 2015 workshop on noisy user-generated text: Twitter lexical normalization and named entity recognition. In *Proceedings of the Workshop on Noisy User-generated Text*, pages 126–135, Beijing, China. Association for Computational Linguistics.

Tyler Bikaun, Melinda Hodkiewicz, and Wei Liu. 2024. MaintNorm: A corpus and benchmark model for lexical normalisation and masking of industrial maintenance short text. In *Proceedings of the Ninth Workshop on Noisy and User-generated Text (W-NUT 2024)*, pages 68–78, San Ġiljan, Malta. Association for Computational Linguistics.

Marcel Bollmann. 2019. A large-scale comparison of historical text normalization systems. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3885–3898, Minneapolis, Minnesota. Association for Computational Linguistics.

Ana-Maria Bucur, Adrian Cosma, and Liviu P. Dinu. 2021. Sequence-to-sequence lexical normalization with multilingual transformers. In *Proceedings of the Seventh Workshop on Noisy User-generated Text (W-NUT 2021)*, pages 473–482, Online. Association for Computational Linguistics.

Monojit Choudhury, Rahul Saraf, Vijit Jain, Animesh Mukherjee, Sudeshna Sarkar, and Anupam Basu. 2007. Investigation and modeling of the structure of texting language. *International Journal of Document Analysis and Recognition (IJDAR)*, 10:157–174.

Grzegorz Chrupała. 2014. Normalizing tweets with edit scripts and recurrent neural embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 680–686, Baltimore, Maryland. Association for Computational Linguistics.

Yasuharu Den. 2009. A multi-purpose electronic dictionary for morphological analyzers (in Japanese). *Journal of the Japanese Society for Artificial Inteligence*, 34(5):640–646.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. QLoRA: Efficient finetuning of quantized LLMs. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional Transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Kazuki Fujii, Taishi Nakamura, Mengsay Loem, Hiroki Iida, Masanari Ohi, Kakeru Hattori, Hirai Shota, Sakae Mizuki, Rio Yokota, and Naoaki Okazaki. 2024. Continual pre-training for cross-lingual LLM adaptation: Enhancing Japanese language capabilities. In *Proceedings of the First Conference on Language Modeling*, COLM, University of Pennsylvania, USA.

Itsuko Fujimura, Shouju Chiba, and Meiko Ohso. 2012. Lexical and grammatical features of spoken and written Japanese in contrast: Exploring a lexical profiling approach to comparing spoken and written corpora. In *Proceedings of the VIIth GSCP International Conference*, pages 393–398. Speech and Corpora.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Kushal Lakhotia, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Maria Tsimpoukelli, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Ning Zhang, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohan Maheswari, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal

Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vítor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Dong, Annie Franco, Anuj Goyal, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Ce Liu, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Cynthia Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Eric-Tuan Le, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Filippos Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hakan Inan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kiran Jagadeesh, Kun Huang, Kunal Chawla, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shishir Patil, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaojian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yu Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. 2024. The Llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Bo Han and Timothy Baldwin. 2011. Lexical normalisation of short text messages: Makn sens a #twitter. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 368–378, Portland, Oregon, USA. Association for Computational Linguistics.

Choochart Haruechaiyasak and Alisa Kongthon. 2013. LexToPlus: A Thai lexeme tokenization and normalization tool. In *Proceedings of the 4th Workshop on South and Southeast Asian Natural Language Processing*, pages 9–16, Nagoya Congress Center, Nagoya, Japan. Asian Federation of Natural Language Processing.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. DeBERTa: Decoding-enhanced

BERT with disentangled attention. In *International Conference on Learning Representations*.

Zheyu He, Yujin Zhu, Linlin Wang, and Liang Xu. 2023. UMRSpell: Unifying the detection and correction parts of pre-trained models towards Chinese missing, redundant, and spelling correction. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10238–10250, Toronto, Canada. Association for Computational Linguistics.

Shohei Higashiyama, Masao Utiyama, Taro Watanabe, and Eiichiro Sumita. 2021a. A text editing approach to joint Japanese word segmentation, POS tagging, and lexical normalization. In *Proceedings of the Seventh Workshop on Noisy User-generated Text (W-NUT 2021)*, pages 67–80, Online. Association for Computational Linguistics.

Shohei Higashiyama, Masao Utiyama, Taro Watanabe, and Eiichiro Sumita. 2021b. User-generated text corpus for evaluating Japanese morphological analysis and lexical normalization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5532–5541, Online. Association for Computational Linguistics.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Taishi Ikeda, Hiroyuki Shindo, and Yuji Matsumoto. 2016. Japanese text normalization with encoder-decoder model. In *Proceedings of the 2nd Workshop on Noisy User-generated Text (WNUT)*, pages 129–137.

SB Intuitions. 2024a. Sarashina2.

SB Intuitions. 2024b. Sarashina2.2.

Nobuhiro Kaji and Masaru Kitsuregawa. 2014. Accurate word segmentation and pos tagging for Japanese microblogs: Corpus annotation and joint modeling with lexical normalization. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 99–109, Doha, Qatar. Association for Computational Linguistics.

Masahiro Kaneko and Naoaki Okazaki. 2023. Reducing sequence length by predicting edit spans with large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 10017–10029, Singapore. Association for Computational Linguistics.

Phillip Keung, Yichao Lu, György Szarvas, and Noah A. Smith. 2020. The multilingual Amazon reviews corpus. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4563–4568, Online. Association for Computational Linguistics.

Risa Kondo, Ayu Teramen, Reon Kajikawa, Koki Horiguchi, Tomoyuki Kajiwara, Takashi Ninomiya, Hideaki Hayashi, Yuta Nakashima, and Hajime Nagahara. 2025. Text normalization for Japanese sentiment analysis. In *Proceedings of the Tenth Workshop on Noisy and User-generated Text*, pages 149–157, Albuquerque, New Mexico, USA. Association for Computational Linguistics.

Taku Kudo, Kaoru Yamamoto, and Yuji Matsumoto. 2004. Applying conditional random fields to Japanese morphological analysis. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 230–237, Barcelona, Spain. Association for Computational Linguistics.

Olli Kuparinen, Aleksandra Miletić, and Yves Scherrer. 2023. Dialect-to-standard normalization: A large-scale multilingual evaluation. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 13814–13828, Singapore. Association for Computational Linguistics.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Peerat Limkonchotiwat, Wannaphong Phatthiyaphaibun, Raheem Sarwar, Ekapol Chuangsuwanich, and Sarana Nutanong. 2021. Handling cross- and out-of-domain samples in Thai word segmentation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1003–1016, Online. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized bert pretraining approach.

Ismini Lourentzou, Kabir Manghnani, and ChengXiang Zhai. 2019. Adapting sequence to sequence models for text normalization in social media. In *Proceedings of the international AAAI conference on web and social media*, volume 13, pages 335–345.

Massimo Lusetti, Tatyana Ruzsics, Anne Göhring, Tanja Samardžić, and Elisabeth Stark. 2018. Encoder-decoder methods for text normalization. In *Proceedings of the Fifth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial 2018)*, pages 18–28, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Kikuo Maekawa, Makoto Yamazaki, Toshinobu Ogiso, Takehiko Maruyama, Hideki Ogura, Wakako Kashino, Hanae Koiso, Masaya Yamaguchi, Makiro Tanaka, and Yasuharu Den. 2014. Balanced corpus of contemporary written Japanese. *Language Resources and Evaluation*, 48(2):345–371.

Benjamin Muller, Benoit Sagot, and Djamé Seddah. 2019. Enhancing BERT for lexical normalization. In *Proceedings of the 5th Workshop on Noisy User-generated Text (W-NUT 2019)*, pages 297–306, Hong Kong, China. Association for Computational Linguistics.

Thanh-Nhi Nguyen, Thanh-Phong Le, and Kiet Nguyen. 2024. ViLexNorm: A lexical normalization corpus for Vietnamese social media text. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1421–1437, St. Julian's, Malta. Association for Computational Linguistics.

Ayaha Osaki, Yoshiaki Kitagawa, and Mamoru Komachi. 2017. Text normalization by sequence labeling for Japanese Twitter documents (nihon-go Twitter bunsho-wo taishō-toshita kēretsu labeling-niyoru hyōki sēkika) [in Japanese]. *IPSJ SIG Technical Report*, 2017-NL-231(12):1–6.

Idhibhat Pankam, Peerat Limkonchotiwat, and Ekapol Chuangsuwanich. 2023. Two-stage thai misspelling correction based on pre-trained language models. In *2023 20th international joint conference on computer science and software engineering (JCSSE)*, pages 7–12. IEEE.

Kunat Pipatanakul, Potsawee Manakul, Natapong Nitarach, Warit Sirichotedumrong, Surapon Nonesung, Teetouch Jaknamon, Parinthapat Pengpun, Pittawat Taveekitworachai, Adisai Na-Thalang, Sittipong Sripaisarnmongkol, Krisanapong Jirayoot, and Kasima Tharnpipitchai. 2024. Typhoon 2: A family of open text and multimodal Thai large language models. *Preprint*, arXiv:2412.13702.

Barbara Plank, Kristian Nørgaard Jensen, and Rob van der Goot. 2020. DaN+: Danish nested named entities and lexical normalization. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6649–6662, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Maja Popović. 2015. chrF: character n-gram F-score for automatic MT evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395, Lisbon, Portugal. Association for Computational Linguistics.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels. Association for Computational Linguistics.

Tao Qian, Yue Zhang, Meishan Zhang, Yafeng Ren, and Donghong Ji. 2015. A transition-based model for joint segmentation, POS-tagging and normalization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1837–1846, Lisbon, Portugal. Association for Computational Linguistics.

Jipeng Qiang, Yun Li, Yi Zhu, Yunhao Yuan, Yang Shi, and Xindong Wu. 2021. LSBert: Lexical simplification based on BERT. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3064–3076.

Qwen Team. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text Transformer. *J. Mach. Learn. Res.*, 21(1).

Itsumi Saito, Kyosuke Nishida, Kugatsu Sadamitsu, Kuniko Saito, and Junji Tomita. 2017. Automatically extracting variant-normalization pairs for Japanese text normalization. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 937–946, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Itsumi Saito, Kugatsu Sadamitsu, Hisako Asano, and Yoshihiro Matsuo. 2014. Morphological analysis for Japanese noisy text based on character-level and word-level normalization. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1773–1782, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.

David Samuel and Milan Straka. 2021. ÚFAL at MultiLexNorm 2021: Improving multilingual lexical normalization by fine-tuning ByT5. In *Proceedings of the Seventh Workshop on Noisy User-generated Text (W-NUT 2021)*, pages 483–492, Online. Association for Computational Linguistics.

Ryohei Sasano, Sadao Kurohashi, and Manabu Okumura. 2013. A simple approach to unknown word processing in Japanese morphological analysis. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 162–170, Nagoya, Japan. Asian Federation of Natural Language Processing.

Binbin Shen, Jie Wang, Meng Meng, and Yujun Wang. 2024. TNFormer: Single-pass multilingual text normalization with a transformer decoder model. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 11671–11675. IEEE.

Ning Shi, Bradley Hauer, and Grzegorz Kondrak. 2024. Lexical substitution as causal language modeling. In *Proceedings of the 13th Joint Conference on Lexical and Computational Semantics (*SEM 2024)*, pages 120–132, Mexico City, Mexico. Association for Computational Linguistics.

Makoto Shing, Kou Misaki, Han Bao, Sho Yokoi, and Takuya Akiba. 2025. TAID: Temporally adaptive interpolated distillation for efficient knowledge transfer in language models. *Preprint*, arXiv:2501.16937.

Yu Tanaka, Yugo Murawaki, Daisuke Kawahara, and Sadao Kurohashi. 2020. Building a Japanese typo dataset from Wikipedia's revision history. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 230–236, Online. Association for Computational Linguistics.

Nobuhiro Ueda, Kazumasa Omura, Takashi Kodama, Hirokazu Kiyomaru, Yugo Murawaki, Daisuke Kawahara, and Sadao Kurohashi. 2023. KWJA: A unified Japanese analyzer based on foundation models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pages 538–548, Toronto, Canada. Association for Computational Linguistics.

Rob van der Goot, Alan Ramponi, Tommaso Caselli, Michele Cafagna, and Lorenzo De Mattei. 2020. Norm it! lexical normalization for Italian and its downstream effects for dependency parsing. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 6272–6278, Marseille, France. European Language Resources Association.

Rob van der Goot, Alan Ramponi, Arkaitz Zubiaga, Barbara Plank, Benjamin Muller, Iñaki San Vicente Roncal, Nikola Ljubešić, Özlem Çetinoğlu, Rahmad Mahendra, Talha Çolakoğlu, Timothy Baldwin, Tommaso Caselli, and Wladimir Sidorenko. 2021. MultiLexNorm: A shared task on multilingual lexical normalization. In *Proceedings of the Seventh Workshop on Noisy User-generated Text (W-NUT 2021)*, pages 493–509, Online. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Aobo Wang, Min-Yen Kan, Daniel Andrade, Takashi Onishi, and Kai Ishikawa. 2013. Chinese informal word normalization: an experimental study. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 127–135, Nagoya, Japan. Asian Federation of Natural Language Processing.

Benjamin Warner, Antoine Chaffin, Benjamin Clavié, Orion Weller, Oskar Hallström, Said Taghadouini, Alexis Gallagher, Raja Biswas, Faisal Ladhak, Tom Aarsen, Nathan Cooper, Griffin Adams, Jeremy Howard, and Iacopo Poli. 2024. Smarter, better, faster, longer: A modern bidirectional encoder for fast, memory efficient, and long context finetuning and inference. *Preprint*, arXiv:2412.13663.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mT5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498, Online. Association for Computational Linguistics.

Koichi Yasuoka. 2023. Sequence-labeling roberta model for dependency-parsing in classical chinese and its application to vietnamese and thai. In *2023 8th International Conference on Business and Industrial Research (ICBIR)*, pages 169–173. IEEE.

Sumeth Yuenyong, Kobkrit Viriyayudhakorn, Apivadee Piyatumrong, and Jillaphat Jaroenkantasima. 2025. OpenThaiGPT 1.5: A Thai-centric open source large language model. *Preprint*, arXiv:2411.07238.

Hao Zhang, Richard Sproat, Axel H. Ng, Felix Stahlberg, Xiaochang Peng, Kyle Gorman, and Brian Roark. 2019. Neural models of text normalization for speech applications. *Computational Linguistics*, 45(2):293–337.

Wenxuan Zhang, Hou Pong Chan, Yiran Zhao, Mahani Aljunied, Jianyu Wang, Chaoqun Liu, Yue Deng, Zhiqiang Hu, Weiwen Xu, Yew Ken Chia, Xin Li, and Lidong Bing. 2024a. SeaLLMs 3: Open foundation and chat multilingual large language models for southeast Asian languages. *Preprint*, arXiv:2407.19672.

Yang Zhang, Travis M. Bartley, Mariana Graterol-Fuenmayor, Vitaly Lavrukhin, Evelina Bakhturina, and Boris Ginsburg. 2024b. A chat about boring problems: Studying GPT-based text normalization. *Preprint*, arXiv:2309.13426.

A JMLN Dataset

A.1 Data Statistics

The detailed dataset statistics are shown in Table 8 and Table 9.

ID	Name	#Sent	#Word	#Norm
01	BJ-OC	1,441	28,631	928
02	BJ-OY	1,785	29,092	1,445
03	RC-BLG	2,312	37,135	766
04	RC-REV	1,541	31,283	310
05	RK-ICB	1,251	21,548	251
06	RK-TRV	1,610	29,240	289
07	RK-RCP	2,479	29,834	1,104
08	AM	1,769	28,055	477
09	NC-VID	918	11,908	262
10	NC-PED	947	14,858	387
11	TW	1078	14,701	858
12	JW	540	18,817	503
13	NU	507	9,176	509
14	SK	684	11,935	286
	Total	18,862	316,216	8,375

Table 8: Statistics of the JMLN Cur-sets.

ID	Name	#Sent	#Word	#Nrom
01	BJ-OC	200	3,981	33
02	BJ-OY	201	3,821	56
03	RC-BLG	200	2,903	57
04	RC-REV	200	3,872	27
05	RK-ICB	200	2,942	16
06	RK-TRV	200	3,139	18
07	RK-RCP	200	2,763	25
08	AM	200	2,932	12
09	NC-VID	150	2,418	23
10	NC-PED	182	2,672	46
11	TW	207	3,127	99
13	NU	200	2,940	82
14	SK	200	2,182	16
	Total	2,540	39,692	510

Table 9: Statistics of the JMLN Rand-sets.

A.2 Data Sources and Licenses

To construct our JMLN dataset, we used following datasets and text sources as shown in Table 10:

- (01–02) BCCWJ (Maekawa et al., 2014): available under a usage contract;[13]
- (03–04) Recruit Dataset: available under a usage contract;[14]
- (05–07) Rakuten Dataset: available under a usage contract;[15]
- (08) Multilingual Amazon Reviews Corpus (Keung et al., 2020): previously available under a proprietary license (now unavailable);[16]
- (09–10) Niconico Dataset: available under specific terms of use;[17]
- (11) Twitter (now X) posts: obtained via the Twitter streaming API (copyright retained by each post's author);
- (12) Japanese Wikipedia Typo Dataset (Tanaka et al., 2020): available under CC-BY-SA 3.0 license;[18]
- (13) Nagoya University Conversation Corpus (Fujimura et al., 2012): available under CC BY-NC-ND 4.0 license;[19]
- (14) Japanese and Chinese Skype Conversation Corpus: available under specific terms of use.[20]

[13]https://clrd.ninjal.ac.jp/bccwj/en/index.html
[14]https://www.nii.ac.jp/dsc/idr/recruit/
[15]https://alaginrc.nict.go.jp/rakuten-dataset.html
[16]https://registry.opendata.aws/amazon-reviews-ml/
[17]https://www.nii.ac.jp/dsc/idr/nico/
[18]https://nlp.ist.i.kyoto-u.ac.jp/EN/?JWTD
[19]https://mmsrv.ninjal.ac.jp/nucc/
[20]http://nakamata.info/database/

A.3 Word Category Definition

We extended the UGT-specific Japanese word categories defined by Higashiyama et al. (2021b) and assigned each word in the annotation sentences to every category that it matches. As shown in Table 11, the categories are divided into vocabulary types and variant-form types, with the latter applied non-standard word forms.

A.4 Standard/Non-Sandard From Definition

As stated in Higashiyama et al. (2021b), "there are no trivial criteria to determine which variant forms of a word are standard forms" (and non-standard forms) "because most Japanese words can be written in multiple ways." Thus, we followed their definition on standard and non-standard forms. In brief, the definitions can be summarized as follows: standard forms are those variants whose relative frequencies in the reference corpus exceed a set threshold, while non-standard forms are identified per variant category based on falling below category-specific frequency thresholds or exhibiting distinctive orthographic features. Example non-standard forms for each category is shown in Table 11. For more detailed definitions, see §4.2 of their paper.

ID	Data name	Source dataset	Text register	Year
01	BJ-OC	BCCWJ: Yahoo! Chiebukuro	Q&A posts/responces	2004–2005
02	BJ-OY	BCCWJ: Yahoo! Blog	Blog posts	2008–2009
03	RC-BLG	Recruit: beauty salon blogs	Blog posts	2012–2014
04	RC-REV	Recruit: beauty salon reviews	Reviews (beauty salon)	2012–2014
05	RK-ICB	Rakuten Ichiba	Reviews (EC site)	2019
06	RK-TRV	Rakuten Travel	Reviews (hotel site)	2017–2019
07	RK-RCP	Rakuten Recipe	Recipes	2017
08	AM	Multilingual Amazon Reviews Corpus	Reviews (EC site)	2000–2015
09	NC-VID	Niconico Dataset: Video meta data	Video descriptions	2018
10	NC-PED	Niconico Dataset: Forum data	Forum posts/replies	2008–2014
11	TW	Twitter	Social media posts	2020–2022
12	JW	Japanese Wikipedia Typo Dataset	Encyclopedia edit history	–2021
13	NU	Nagoya University Conversation Corpus	Conv. transcriptions	2001–2003
14	SK	Skype Conversation Corpus	Conv. transcriptions	2012

Table 10: Data sources of JMLN. The Year column indicates the years of original text publication.

		Example	Standard forms	Translation
Vocabulary type	Neologisms/Slang	コピペ	–	copy and paste
	Proper names	ドラクエ	–	Dragon Quest
	Onomatopoeia	キラキラ	–	glitter
	Interjections	おお	–	oops
	Dialect words	ほんま	–	truly
	Foreign words	ＥＡＳＹ	–	easy
	Ancient words*	[行く]べし	–	should [go]
	Character endings*	[行く]にゃ	–	†
	Blend words*	おはこんばんにちは	–	‡
	Emoticons/AA	（＾－＾）	–	
Variant-form type	Character type variants	カワイイ	かわいい,可愛い	cute
	Alternative representations	大きぃ	大きい	big
	Sound change variants	おいしーい	おいしい,美味しい	tasty
	Typographical errors	つたい	つらい,辛い	tough

Table 11: Word categories extended from Higashiyama et al. (2021b). New categories are marked with "*." "[]" indicates the context. †"[行く] にゃ" is a kitten-style sentence ending and might be expressed as "[I go,] meow" or "[Goin']nya." ‡"おはこんばんにちは" is a coined blend of "good morning," "good afternoon," and "good evening," and might be expressed as "Good morn-noon-evening."

Data	Number				F_1 agreement score						
	Sent	Word	Cate	VForm	Word	POS	Lem	Cate	Cate^b	VForm	VForm^b
01 BJ-OC	20	327/325	30/30	19/18	99.08	98.77	98.16	83.33	83.33	86.49	86.49
02 BJ-OY	20	359/358	52/48	17/15	99.58	96.79	95.40	86.00	94.00	75.00	81.25
03 RC-BLG	32	1424/1424	133/124	45/43	99.86	99.02	98.81	81.71	91.83	63.64	84.09
04 RC-REV	41	2802/2802	129/137	22/22	100.0	99.04	98.75	96.24	96.24	100.0	100.0
05 RK-ICB	20	600/601	27/27	15/14	99.75	98.58	98.58	85.19	96.30	82.76	89.66
06 RK-TRV	25	748/748	23/29	11/14	100.0	99.20	98.93	84.62	84.62	80.00	80.00
07 RK-RCP	22	442/442	43/39	26/20	99.32	98.64	97.96	87.80	92.68	82.61	82.61
08 AM	20	675/675	85/86	32/34	100.0	99.56	95.56	91.23	99.42	90.91	96.97
09 NC-VID	20	353/358	55/50	20/10	97.33	95.08	93.67	70.48	83.81	46.67	53.33
10 NC-PED	20	443/446	63/63	24/24	98.31	97.19	94.26	77.78	85.71	62.50	66.67
Total	240	8173/8179	640/633	231/214	99.66	98.65	97.82	85.78	92.22	76.85	83.60

Table 12: Statistics on inter-annotator agreement. The "Number" columns show counts of annotated sentences (Sent), words, word categories (Cate), variant forms (VForm) for each annotator (formatted as "value1/value2"). The "F_1 agreement score" columns report F_1 scores for word segmentation (Seg), parts-of-speech (POS), lemmas (Lem), Cate, and VForm. Binary agreement on the presence or absence of a category/variant-form assignment is indicated by "*b*."

A.5 Standard Form Annotation

The annotation process first identifies word boundaries and then assigns each word to (a category and) a standard form ID, as in the example below (meaning "Super tired today."); for simplicity, the example displays the standard-form string instead of the ID. This approach makes it possible to obtain explicit mappings between non-standard words and their standard forms.

Original text	今日はスゴクツカレタ〜
Segmented text	今日 | は | スゴク | ツカレタ〜
Standard forms	– | – | すごく | 疲れた

However, when only sentence-level normalization is provided, aligning standard and non-standard forms is not straightforward. In our example, the contiguous span mapping from "スゴクツカレタ〜" to "すごく疲れた" ("super tired") is easily identified, but determining precise word-level alignments remains challenging.

A.6 Inter-Annotator Agreement

Table 12 shows the detailed statistics on inter-annotator agreement. Sentences in domains 01–07 were annotated by Annotators B and C, and those in domains 08–10 were by B and D. Notably, after we fixed annotation disagreements for these sentences through discussions, the sentences were integrated into our final dataset.

B Encoder-based Approach Variants

In addition to the FULL-SEG approach, we introduce two variant approaches based on the encoder-base detect&infill method: PART-SEG and FULL-SEG-POS. Experimental results on these variants are reported in D.2.

Full/Partial Word Segmentation In the boundary prediction subtask, we employ full or partial word segmentation (FULL-SEG and PART-SEG), depending on whether the training sentences are annotated with full word boundaries or only with non-standard word boundaries. Unlike the former case described in §5, in the latter case boundary tags for tokens within standard words should be assigned `O`. For example, the tag sequence should be [`B`, `I`, `I`, `E`, `O`, `O`, `O`] for the input text in Figure 1.

Word Feature Prediction In the detection step, we can optionally employ multi-task learning for word feature prediction. Specifically, we adopt part-of-speech (POS) tag prediction for each token using an additional linear layer if POS annotation is available. We refer to this approach as FULL-SEG-POS.

C Detailed Experimental Settings

C.1 Definition of Evaluation Metrics

Assume each input sentence $\boldsymbol{x}$ has a sequence of gold normalization instances $\mathcal{N}_{\boldsymbol{x}} = \{(b_i, e_i, S_i)\}_i$, where each instance (b_i, e_i, S_i) consists of a span (b_i, e_i) and a set $S_i = \{s_{i,k}\}_{k=1}^{K_i}$ of one or more standard forms for the corresponding non-standard word. A system is required to output a set of predicted normalization instances $\hat{\mathcal{N}}_{\boldsymbol{x}} = \{(\hat{b}_j, \hat{e}_j, \hat{s}_j)\}_j$, where each $\hat{s}_j$ is a single predicted standard form.

We count a predicted instance as a TP if its span $(\hat{b}_j, \hat{e}_j)$ matches the span (b_i, e_i) of a gold instance and its predicted form $\hat{s}_j$ belongs to the corresponding gold set S_i over all test sentences. Precision P, recall R, and the $F_{0.5}$ score over the test set are then defined as follows:

$$\begin{aligned} P &= \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}}, \\ R &= \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}, \\ F_{0.5} &= (1+0.5^2)\frac{PR}{0.5^2P+R}. \end{aligned}$$

C.2 Pretrained Models

We list all pre-trained models used in our experiments in Table 19. We selected these models based on their strong performance on general benchmarks; however, for Thai encoder-only and encoder-decoder models, few alternative candidates were available.

C.3 Model Hyperparameters

The hyperparameter values used in the experiments are listed in Table 13.

We conducted hyperparameter search for some important parameters within our computational budgets. Based on $\mathrm{F}_{0.5}$ score for the JMLN dev set, we chose the best value of learning rate from the search space of {1e-5, 2e-5, 3e-5, 4e-5, 5e-5} for encoder-based models using RoBERTa-large, chose the best value of learning rate from {1e-4, 2e-4, 3e-4, 4e-4, 5e-4} and that of beam search width during inference from {1, 2, 3} for encoder-decoder and decoder-based models using T5-large and Sarashina2-7b, respectively. We also chose the best value of LoRA rank from {4, 8, 16} for decoder-only models using Sarashina2-7b.

Hyperparameter	Value
Training epochs	ja: 30; th: 20
Batch size	ja: 16; th: 32
Learning rate	3e-5
Learning rate scheduler	linear
Warmup ratio	0.1
Gradient norm clipping threshold	1.0
Optimizer	AdamW
Training epochs	ja: 30; th: 15
Batch size	ja: 32; th: {4, 16}
Learning rate	2e-4
Learning rate scheduler	constant
Warmup ratio	0.1
Gradient norm clipping threshold	1.0
Optimizer	AdaFactor
Beam width for inference	2
Training epochs	10
Batch size	8
Learning rate	2e-4
Learning rate scheduler	cosine
Warmup ratio	0.03
Weight decay	0.001
Gradient norm clipping threshold	0.3
Optimizer	paged_adamw_32bit
bf16	True
LoRA rank	8
LoRA alpha	16
LoRA dropout	0.05
LoRA target modules	all linear layers
Quantization bit	{none, 4bit}
Beam width for inference	1

Table 13: Hyperparameter settings for encoder-only (top), encoder-decoder (middle), and decoder-only models (bottom). Batch sizes of 4 and 16 were used for the Thai mT5-large model and the other Thai encoder-decoder models. For models with 7B parameters or more, 4-bit quantization were applied during fine-tuning.

C.4 Prompts for Decoder-only Models

We used the instruction prompts in Table 3 as `{inst}` in the full prompt text: "Instruction: \n`{inst}`\n\nInput: \n`{src}`\n\nOutput: \n". These are common to both Japanese and Thai models, and the `{lang}` placeholder is specified by either language name.

C.5 Computational Budget for Fine-tuning

In our experiments, we used NVIDIA V100 GPUs with 32GiB memory, A100 GPUs with 80GiB memory, and H200 GPUs with 140GiB memory. In total, the models were fine-tuned for 42 GPU hours on V100s, 2700 GPU hours on A100s, and 141 GPU hours on H200s. Encoder-only and encoder-decoder models were fine-tuned on a single GPU, whereas decoder-only models were fine-tuned using eight GPUs.

Instruction text
If no informal `{lang}` word forms exist in the input text, output the text as is. Otherwise, identify informal word forms and normalize them into their corresponding standard forms. Provide the full normalized text where the original word forms are replaced with the standard forms.
If no informal `{lang}` word forms exist in the input text, output the text as is. Otherwise, identify informal word forms and normalize them into their corresponding standard forms. Provide the full normalized text, embedding the original and normalized word forms in the format "[[before>>after]]". Ensure that the concatenated string of the text outside the brackets and the "before" parts is identical to the input text.
If no informal `{lang}` word forms exist in the input text, output exactly "NONE". Otherwise, identify every informal word form and normalize it into its corresponding standard form. For each occurrence, output a record in the format "before>>after>>count". Here, count is the count of how many times the identical original string has already appeared earlier in the input text. If multiple informal forms are found, output each record in the order they occur and separate them with "||".

Figure 3: Instruction prompts for decoder-only models with PLAIN (top), STRUCT (middle), and SPAN approaches (bottom).

C.6 Throughput Calculation Setting

In the experiments reported in §7.3, we used the following settings. For each model checkpoint (from one of two runs), we conducted a single warm-up inference followed by three inference passes over all 3,786 JMLN test sentences (a total of 11.1k characters), which were sorted by increasing token count according to its tokenizer. These evaluations were run at multiple batch sizes; we selected the batch size that yielded the highest throughput (shown in Table 14) and reported the mean throughput of the three runs. Models were cast to `float16` on the V100 and to `bfloat16` on the H100. All inference was performed using the Hugging Face Transformers (Wolf et al., 2020) library.

Model	V100	H200	V100	H200
	FULL-SEG			
BERT-large	64	128	–	–
RoBERTa-large	64	128	–	–
DeBERTa-large	64	128	–	–
	STRUCT		SPAN	
T5-large	256	256	256	256
Sarashina2.2-1B	128	256	256	256
Sarashina2.2-3B	128	128	64	128
Sarashina2-7B	64	128	32	128

Table 14: Batch size yielding the best throughput for each method.

C.7 Word Coverage Indicators

For analyses in experiments reported in §7.5 and §7.6, we introduced three word coverage-based indicators. Below, we provide further explanations on them.

The Surf-Outside-Train rate is defined as "the proportion of non-standard surface tokens in the test set that are not found among the training set's non-standard surfaces." Assume that the non-standard surface tokens appearing in the entire training set are ["u", "u", "r", "thx"], while those appearing in the entire test set are ["u", "r", "cuz"]. In this case, the rate is $1/3 = 0.33$.

The Surf-In-Train rate is defined as "the proportion of original surface forms for examples that matched any non-standard forms in the training set." Assume the same training set above and non-standard surface examples of interest are ["u", "r", "cuz"], the rate is $2/3 = 0.67$

The Norm-In-Lex rate is defined as "the proportion of predicted normalized forms appearing in the UniDic lexicon." Assume the set of normalized strings produced by the model is {"you", "are", "becaus"}, and an external lexicon contains only "you" and "are" from that set; the rate is $2/3 = 0.67$.

D Detailed Experimental Results

D.1 Accuracy Across Model Series

For the experiment reported in Table3 (§7.1), we extracted results for different model size within each model series and plotted them in Figure 4. Performance improved with increasing model size across model series.

D.2 Comparison of Encoder Approaches

As a preliminary experiment, we compared variants of encoder-based methods—PART-SEG, FULL-SEG, and FULL-SEG-POS, introduced in Appendix B—on the JMLN dev set. We also report the F1 scores for word segmentation, POS tagging, and length prediction (Len^p: tokens in positive non-standard words; Len^n: other tokens), as well as additional normalization metrics: sentence-level exact match accuracy (S-Acc^p: accuracy for sentences containing at least one positive non-standard word; S-Acc^n: accuracy for other sentences) and the chrF score (Popović, 2015) implemented in sacreBLEU (Post, 2018).[21] Table 16 shows the

[21]We calculated chrF scores with the default options (signature: `nrefs:1|case:mixed|eff:yes|nc:6|nw:0|space:no|version:2.5.1`).

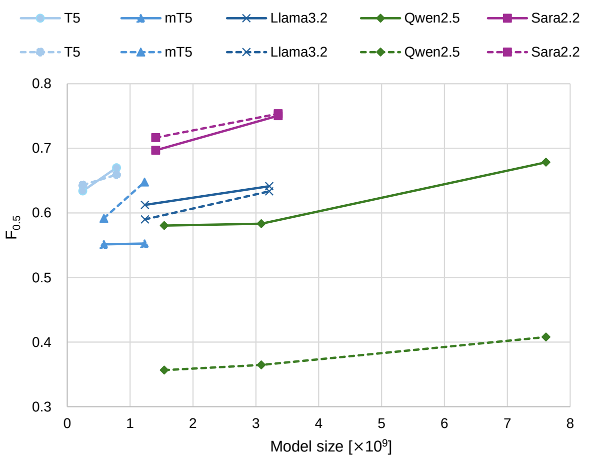

Figure 4: Plot of $F_{0.5}$ scores in Table3 for each model series—T5, mT5, Llama-3.2, Qwen2.5, and Sarashina2.2. The scores for the STRUCT and SPAN approaches are shown with solid and dotted lines, respectively.

results.

The encoder-only models (3 backbone models×2 model size) with the FULL-SEG approach obtained +0.01–0.05 $F_{0.5}$ gains from the PART-SEG counterpart, indicating the importance of learning full word segmentation tasks. Adding POS tagging task showed no clear improvements.

D.3 Comparison of Generative Approaches

As a preliminary experiment, we compared the three generative approaches—PLAIN, STRUCT, and SPAN— using T5, Sarashina2/2.2, and Swallow models on the JMLN dev set. The STRUCT and SPAN approaches outperformed or matched the PLAIN approach for T5 and Sarashina2/2.2 models (e.g., -0.009 to +0.034 S-Acc^p points), but underperformed for Swallow (e.g., -0.076 S-Acc^p points).

D.4 Results for Each Category

Table 15 shows the JMLN test performance (recall) of the three models evaluated in §7.5 for each domain. Consistent with the domain-specific evaluation results in §7.5, Sarashina achieved the highest recall across all categories. All three models showed the same recall order across the four categories, and the consistently lowest recall of the fourth category, i.e., typos, once again highlights the difficulty of correcting them.

D.5 Prediction Examples

Table 18 shows model outputs on the JMLN dev set for the three models evaluated in §7.5; in addition

Category	#	DeBERTa	T5	Sara2.2
Char type var.	534	0.395	0.375	**0.563**
Alter rep.	271	0.550	0.500	**0.703**
Sound change	794	0.796	0.739	**0.812**
Typos	178	0.087	0.076	**0.219**

Table 15: JMLN test recall of representative models (DeBERTa-large, T5-large, and Sarashina2.2-3b) for each category.

to error type classification (TP, FP, and FN) based on the gold standard evaluation, the first author also assessed the validity of each output (✓: valid, △: questionable, ✗: invalid).

In example (a), all three methods produced semantically valid normalized strings at the phrase level; however, only DeBERTa was classified as a TP, as it correctly matched the gold span. Similarly, in examples (b) and (c), Sarashina produced plausible normalized strings, but due to mismatches with the gold annotation, the outputs were classified as FPs. In example (d), Sarashina normalized the input to a synonym (i.e., “darling” to “husband”); while the result was semantically appropriate, it falls outside the scope of the task, which requires normalization to variant surface forms of the same word. In examples (e) and (f), Sarashina produced normalized strings that were semantically unrelated to the original input. DeBERTa generated invalid, non-word outputs in examples (b) and (d), while T5 did so in example (c). Example (g) shows that all three models produced different but erroneous outputs, suggesting that normalization becomes particularly challenging when multiple non-standard words appear in sequence, making boundary and word identification more difficult.

Backbone	Approach	Seg	POS	Len^p	Len^n	Norm					
		F$_1$	F$_1$	F$_1$	F$_1$	P	R	F$_{0.5}$	S-Acc^p	S-Acc^n	chrF
Leave-As-Is		–	–	–	–	–	0	–	0	1	95.29
BERT-B	None	–	–	0.644	–	0.676	0.510	0.635	0.487	0.982	96.69
	Seg	0.980	–	0.656	0.978	0.721	0.546	0.678	0.518	0.979	96.82
	Seg-POS	0.978	0.963	0.636	0.976	0.649	0.532	0.621	0.508	0.970	96.56
RoBERTa-B	None	–	–	0.630	–	0.662	0.515	0.626	0.488	0.976	97.04
	Seg	0.982	–	0.669	0.980	0.726	0.537	0.678	0.502	0.981	97.22
	S&P	0.982	0.971	0.677	0.981	0.713	0.566	0.678	0.525	0.973	97.27
DeBERTa-B	None	–	–	0.637	–	0.678	0.516	0.638	0.489	0.977	97.12
	Seg	0.982	–	0.656	0.980	0.745	0.516	0.685	0.477	0.989	97.16
	S&P	0.983	0.972	0.680	0.982	0.716	0.566	0.680	0.528	0.974	97.26
BERT-L	None	–	–	0.682	–	0.717	0.570	0.682	0.542	0.979	96.92
	Seg	0.984	–	0.704	0.983	0.769	0.588	0.725	0.549	0.979	97.01
	S&P	0.980	0.967	0.691	0.979	0.756	0.576	0.712	0.534	0.984	97.04
RoBERTa-L	None	–	–	0.671	–	0.706	0.557	0.670	0.515	0.980	97.27
	Seg	0.983	–	0.701	0.982	0.753	0.571	0.708	0.542	0.980	97.34
	S&P	0.985	0.974	0.713	0.983	0.765	0.580	0.719	0.551	0.984	97.49
DeBERTa-L	None	–	–	0.685	–	0.763	0.556	0.710	0.523	0.985	97.34
	Seg	0.986	–	0.718	0.984	0.763	0.593	0.722	0.556	0.980	97.54
	S&P	0.985	0.975	0.727	0.983	0.788	0.592	0.739	0.548	0.984	97.50

Table 16: Detailed results of encoder-only models on the JMLN dev sets.

Backbone	Approach	P	R	F$_{0.5}$	S-Acc^p	S-Acc^n	chrF
–	Leave-As-Is	–	0	–	0	1	95.29
T5-base	Plain	–	–	–	0.497	0.958	96.54
	Struct	0.727	0.491	0.663	0.465	0.983	96.94
	Span	0.707	0.527	0.662	0.509	0.974	97.01
T5-large	Plain	–	–	–	0.535	0.950	96.44
	Struct	0.769	0.558	0.714	0.531	0.984	97.31
	Span	0.751	0.568	0.705	0.566	0.975	97.28
Sarashina2.2-1B	Plain	–	–	–	0.602	0.985	97.92
	Struct	0.750	0.635	0.724	0.602	0.972	97.72
	Span	0.770	0.609	0.730	0.593	0.975	97.59
Sarashina2.2-3B	Plain	–	–	–	0.656	0.980	98.17
	Struct	0.792	0.682	0.766	0.651	0.972	97.94
	Span	0.792	0.677	0.767	0.653	0.979	98.11
Sarashina2-7B	Plain	–	–	–	0.646	0.982	98.08
	Struct	0.766	0.670	0.745	0.658	0.971	97.99
	Span	0.767	0.662	0.743	0.637	0.965	97.74
Llama-3.1-Swallow-8B	Plain	–	–	–	0.638	0.983	98.03
	Struct	0.771	0.609	0.732	0.584	0.980	97.68
	Span	0.755	0.602	0.718	0.562	0.974	97.50

Table 17: Detailed results of encoder-decoder and decoder-only models on the JMLN dev sets.

(a)	RK-ICB: (···) 効果はわからない～～$_{\text{Pos:Unk}}$。 (Not sure about the effect.)				
	Gold	ない～～→{ない}	(not)		
	DeBERTa-L	ない～～→ない$_{\text{InL}}$	(not)	TP	✓
	T5-L	わからない～～→わからない$_{\text{OOL}}$	(not sure)	FP&FN	✓
	Sarashina2.2-3B	わからない～～→わからない$_{\text{OOL}}$	(not sure)	FP&FN	✓
(b)	BJ-OC: かいわれ大根って、(···) 水で洗ったらそのままだべれる$_{\text{Pos:Unk}}$の? (Can you just was daikon radish sprouts with water and eat it as is?)				
	Gold	だべれる→{食べれる}	(can eat)		
	DeBERTa-L	だべれる→食れる$_{\text{OOL}}$	(∅)	FP&FN	✗
	T5-L	–		FN	✗
	Sarashina2.2-3B	だべれる→食べられる$_{\text{OOL}}$	(can eat)	FP&FN	✓
(c)	BJ-OC: (···) 近鉄の「まわりゃんせ」がおトク$_{\text{Neg:Unk}}$じゃないでしょうか。 (Isn't Kintetsu Railway's "Mawaryanse" pass a good deal?)				
	Gold	–			
	DeBERTa-L	–		–	–
	T5-L	トク→とこう$_{\text{InL}}$	(∅)	FP	✗
	Sarashina2.2-3B	おトク→お得$_{\text{OOL}}$	(good deal)	FP	✓
(d)	BJ-OY: (···) だぁりん$_{\text{Pos:Unk}}$がおごってくれました。 (My darling treated me.)				
	Gold	だぁりん→{ダーリン}	(darling)		
	DeBERTa-L	だぁりん→だじりん$_{\text{OOL}}$	(∅)	FP&FN	✗
	T5-L	–		FN	✗
	Sarashina2.2-3B	だぁりん→旦那$_{\text{InL}}$	(husband)	FP&FN	△
(e)	BJ-OY: (···) ウォルトンでカンツリ$_{\text{Neg:Unk}}$と利根川水系でバスをやんべ。 (I'm going kan-tsuri (= fishing at a managed sport) at Walton and bass fishing in Tone River system.)				
	Gold	–			
	DeBERTa-L	–		–	–
	T5-L	–		–	–
	Sarashina2.2-3B	カンツリ→キャッチ&リリース$_{\text{OOL}}$	(catch & release)	FP	✗
(f)	NU: (···) 最高気温25度とかっしょう$_{\text{Pos:Unk}}$? (The high is like 25°C or something, right?)				
	Gold	っしょう→{でしょう}	(right?)		
	DeBERTa-L	–		FN	✗
	T5-L	–		FN	✗
	Sarashina2.2-3B	かっしょう→夏$_{\text{InL}}$	(summer)	FP&FN	✗
(g)	TW: あ～あまぁ～た$_{\text{Pos:Unk}}$初音ミクがトレンド入りしてるよ (···) (Ugh, Hatsune Miku is trending again.)				
	Gold	まぁ～た→{また}	(again)		
	DeBERTa-L	あまぁ～→あまっ$_{\text{InL}}$	(extra)	FP&FN	✗
	T5-L	まぁ～た→まあ$_{\text{InL}}$	(well...)	FP&FN	✗
	Sarashina2.2-3B	あまぁ～た→あまった$_{\text{OOL}}$	(extra)	FP&FN	✗

Table 18: Example original text (fragment) and corresponding model outputs. For each row of gold-standard and model output, columns 2–5 indicate: (1) original string→normalized string, (2) gloss or the normalized string, (3) error type, (4) manual validity judgement by the first author (✓: valid, △: questionable, ✗: invalid). Non-standard words in the original text are underlined; if positive instances, they are marked with a subscript Pos : Unk, and if negative, with Neg : Unk (Unk indicates that the word did not appear in the training set). Predicted normalized strings are marked with subscripts InL or OOL, indicating whter the form is included or not included in the UniDic lexicon.

ID	Pretrained Model	Lang	Exp		Size	Hugging Face ID	License
			ja	th			
01	BERT-base	ja	✓		91M	tohoku-nlp/bert-base-japanese-char-v3	Apache 2.0
02	BERT-large	ja	✓		310M	tohoku-nlp/bert-large-japanese-char-v2	Apache 2.0
03	RoBERTa-base	ja	✓		100M	ku-nlp/roberta-base-japanese-char-wwm	CC BY-SA 4.0
04	RoBERTa-large	ja	✓		320M	ku-nlp/roberta-large-japanese-char-wwm	CC BY-SA 4.0
05	RoBERTa-base	th		✓	88M	KoichiYasuoka/roberta-base-thai-char	Apache 2.0
06	DeBERTa-base	ja	✓		100M	ku-nlp/deberta-v2-base-japanese-char-wwm	CC BY-SA 4.0
07	DeBERTa-large	ja	✓		330M	ku-nlp/deberta-v2-large-japanese-char-wwm	CC BY-SA 4.0
08	T5-base	ja	✓		250M	retrieva-jp/t5-base-long	CC BY-SA 4.0
09	T5-large	ja	✓		780M	retrieva-jp/t5-large-long	CC BY-SA 4.0
10	T5-base	th		✓	250M	kobkrit/thai-t5-base	N/A
11	mT5-base	M	✓	✓	580M	google/mt5-base	Apache 2.0
12	mT5-large	M	✓	✓	1.2B	google/mt5-large	Apache 2.0
13	Llama-3.2-1B	M	✓	✓	1.2B	meta-llama/Llama-3.2-1B	Llama 3.2
14	Llama-3.2-3B	M	✓	✓	3.2B	meta-llama/Llama-3.2-3B	Llama 3.2
15	Llama-3.1-8B	M	✓	✓	8.0B	meta-llama/Llama-3.1-8B	Llama 3.1
16	Swallow-8B	M→ja	✓		8.0B	tokyotech-llm/Llama-3.1-Swallow-8B-v0.2	Llama 3.1
17	Qwen2.5-1.5B	M	✓	✓	1.5B	Qwen/Qwen2.5-1.5B	Apache 2.0
18	Qwen2.5-3B	M	✓	✓	3.1B	Qwen/Qwen2.5-3B	Qwen Research
19	Qwen2.5-7B	M	✓	✓	7.6B	Qwen/Qwen2.5-7B	Apache 2.0
20	TinySwallow-1.5B	M→ja	✓		1.5B	SakanaAI/TinySwallow-1.5B	Apache 2.0
21	Sarashina2.2-1B	ja&en	✓		1.4B	sbintuitions/sarashina2.2-1b	MIT
22	Sarashina2.2-3B	ja&en	✓		3.4B	sbintuitions/sarashina2.2-3b	MIT
23	Sarashina2-7B	ja&en	✓		7.3B	sbintuitions/sarashina2-7b	MIT
24	Typhoon2-1B	M→th		✓	1.2B	scb10x/llama3.2-typhoon2-1b	Llama 3.2
25	Typhoon2-3b	M→th		✓	3.2B	scb10x/llama3.2-typhoon2-3b	Llama 3.2
26	Typhoon2-8B	M→th		✓	8.0B	scb10x/llama3.1-typhoon2-8b	Llama 3.1
27	ThaiGPT1.5-7B	M→th		✓	7.6B	openthaigpt/openthaigpt1.5-7b-instruct	Qwen
28	SeaLLMs-v3-1.5B	M→M		✓	1.5B	SeaLLMs/SeaLLMs-v3-1.5B	SeaLLMs
29	SeaLLMs-v3-7B	M→M		✓	7.6B	SeaLLMs/SeaLLMs-v3-7B	SeaLLMs

Table 19: Backbone models used in Japanese and Thai experiments ("Exp"="ja" and "th"). The "Lang" column indicates the languages that the model mainly trained on ("M" indicates a multilingual model, and "M→*" indicates a continually pre-trained model from a multilingual model).