Phonetic to morphemic text conversion of a Very Low Resource Language using the ByT5-small Transformer Model

Nick Overacker
Information & Communication Engineering
Kitami Institute of Technology
Kitami, Japan
m3235380013@std.kitami-it.ac.jp

Michal Ptaszynski
Information & Communication Engineering
Kitami Institute of Technology
Kitami, Japan
ptaszynski@ieee.org

Karol Nowakowski
Media and Information Technology
Tohoku University of Community Service and Science
Sakata, Japan
karol@koeki-u.ac.jp

Kenyu Yamamaru
The Foundation for Ainu Culture
Shiraoi, Japan
yamamaru-kenyu95a@ff-ainu.or.jp

***Abstract*—The Ainu language, indigenous to the Japanese island of Hokkaido and its surrounding areas, is a critically endangered low resource language. It is often transcribed in either the Japanese katakana script, the Latin alphabet, or both. Transliteration from one script to the other is non-trivial, and conversion from phonetic to morphemic transcriptions is particularly difficult using a rules-based programming approach. We fine-tuned the ByT5-small transformer model on a dataset of Saru Ainu texts to automate the transliteration process. Our experiments evaluate multiple hyperparameter configurations and demonstrate the model's capability to accurately convert Ainu text from katakana to the Latin alphabet. Our best model achieved a characTER score of 0.0067, demonstrating high accuracy in transliteration and providing a valuable tool for Ainu language preservation. More broadly, this work demonstrates a useful method to effectively expand the corpora of low-resource languages which have been recorded in several different orthographies.**

I. Copyright Notice

II. Introduction

Hokkaido Ainu was described by UNESCO in 2010 as a critically endangered language "with fewer than a dozen rudimentary speakers" [1]. It is additionally a low-resource language that traditionally had no written form before the incorporation of Hokkaido into Japan, and transcriptions written by Ainu language researchers constitute a significant portion of existing Ainu text. In the face of this dire situation, there are active efforts to promote Ainu language education such as free in-person classes [2], weekly lessons broadcast by STV Radio [3] and the annual *Itak=an ro* Ainu Speech Contest [4].

Ainu is typically transcribed using the Latin alphabet, Japanese katakana, or both. In addition, there have been several different conventions for transcribing Ainu in both of these writing systems. Because of this, existing Ainu text is effectively fragmented into several distinct orthographies, each comprising only a portion of an already small corpus. This fragmentation presents challenges for creating consistent resources and limits the availability of a unified corpus, potentially impacting digital processing and preservation efforts.

Although there is still no standardized orthography for Ainu, there are now common practices that can be treated as *de facto* standards to be applied going forward. Text written in Latin characters is commonly understood to be a morphemic[1], whereas katakana is used to transcribe pronunciation.

An automated means to convert Ainu text from katakana to the Latin alphabet is desirable for many reasons. For example, such a tool could be used to expand the Ainu corpus for use in training Ainu-centered language models. In addition, non-Japanese

[1]I.e., spelling reflects lexical items rather than pronunciation. An example in English would be writing "you are" to transcribe the spoken word "you're".

TABLE I
EXAMPLES OF AMBIGUOUS TRANSCRIPTIONS OF AINU WORDS.

Katakana	Latin	Definition	Source
ヤイライケ	yayirayke	to thank	[7]
ヤイライケ	yayrayke	to commit suicide	[7]
カ	k=a	I sit	[8]
カ	ka	string	[8]

people who wish to study the Ainu language may find it more accessible with the help of such a tool.

We have fine-tuned several state-of-the-art Ainu transliteration models based on the ByT5-small model [5]. In this paper, we document the process we followed to find locally optimal values for weight decay and learning rate for this task, and we compare the results to existing Ainu character conversion software. Finally, we briefly describe the next steps we will pursue to further enable the development of improved Ainu language models.

III. Related Work

Cjyet Yo of Osaka University recently investigated the practical limits of a rules-based approach to Ainu language transliteration [6]. Yo found that it is practical to transliterate Ainu from morphemic Latin transcriptions to phonetic katakana transcriptions using a rules-based programming approach. However, phonetic katakana transcription of individual words can be semantically lossy compared to Latin transcriptions, resulting in ambiguity as is demonstrated in Table I.

Because of this ambiguity as well as other difficulties for a rules-based programming approach, a machine learning approach seems better suited for the problem of converting Ainu text from katakana to Latin script. As a baseline, we patched Yo's *ainconv* package[2] to be more compatible with our corpus and evaluated its performance on our test set. The results are shown in Table II.

Besides Yo's *ainconv* package, at time of submission there seems to only one other Ainu language transliteration tool[3], which was release by the Hokkaido University linguistics club *Huling* on November 1, 2024. It is also a rules-based program, but due to the time of its release we have not had the opportunity to rigorously compare it to Yo's package. However, it appears to be subject to similar limitations.

[2]The patched version of Yo's *ainconv* package may be accessed at via the corresponding author's GitHub repository at https://github.com/NickOveracker/ainconv-rs/.

[3]Accessible at https://hulinguistics.github.io/conv/ain

TABLE II
PERFORMANCE METRICS FOR THE *kana2latn* FUNCTION IN THE *ainuconv* PYTHON PACKAGE, EVALUATED OVER THE TEST SET.

characTER	CER	chrF
0.0814	0.0822	76.60

IV. Methodology

A. Data Preparation

We augmented the unified Ainu corpus prepared by Nowakowski, et al. [9] with all Ainu text contained in the 2013 edition of the *New Express* Ainu textbook [10]. The augmented dataset was then pre-processed according to the following steps, in order:

1) **Normalize Latinized punctuation and remove accent marks from Latin letters.** This step consists of pattern replacements, such as replacing Japanese full-width punctuation characters (e.g., '、') with Latin half-width characters (e.g., ',').

2) **Remove all entries that do not have both Latin and katakana transcriptions.** Training a transliteration model requires both scripts to be present. After removing entries with only one script, only Saru dialect data remained.

3) **Remove identical entries.** We judged two entries to be identical if and only if both have identical normalized Latin transcriptions and identical katakana transcriptions. We did not consider other data columns such as the Japanese translation. Removing identical entries helps to prevent the same sentence pairs from appearing in more than one of the split datasets (i.e., train, validation, and test sets). In addition, we believe that this may help ensure that the model weights are not biased towards any particular subset of data.

4) **Remove near-identical entries.** We defined sentences with identical katakana transcriptions but different Latin transcriptions as "near-identical". We removed near-identical entries that had obviously incorrect Latin transcriptions (for example, those including extraneous characters not present in the katakana text such as '@', and those that added or deleted words). Entries which were not obviously incorrect were left intact. Only 22 near-identical pairs were left after this step.

5) **Replace all instances of the consonant cluster 'np' with 'mp' in the Latinized**

TABLE III
TRAIN/VALID/TEST SPLIT SHARES OF THE CORPUS.

	Training	Validation	Test
Entries	49380	6172	6173
Percentage	80%	10%	10%

entries.
We took this step in order to match the orthography conventions in the *New Express* Ainu textbook [10].

6) **Remove spacing between personal affix markers and the words to which they are affixed.**
For example, change '*ku= ipe*' to '*ku=ipe*'.

7) **Randomly split the data into training (80%), validation (10%), and test sets (10%), as shown in Table III.**
The same split was used for each experiment.

8) **Shuffle the training and validation data.**
This was done at loading time in the fine-tuning program with a fixed random seed value.

Each data entry consists of a katakana string between 1 and 97 characters long (mean length: 19.36) and a matching Latinized string between 1 and 110 characters long (mean length: 27.93).

B. Model and Tokenizer

We selected ByT5-small, a "pre-trained byte-level Transformer [model] based on the T5 architecture" [5], as the base model for these experiments. A byte-level model is attractive for the task of Ainu transliteration because there is no need to expand its vocabulary when fine-tuning on unseen languages. A single byte can only take on 256 possible values, none of which are out-of-vocabulary for the ByT5 model family.

Furthermore, the Ainu language allows the construction of morphemically dense words; a written word might have as many morphemes as there are letters in the Latin transcription[4], or even more morphemes than there are characters in the katakana transcription[5]. A byte-based model helps to accommodate these by ensuring a high token-density per word[6].

[4]E.g., *aep*, "food" [8], glossed as *a-e-p*, "4th-person transitive subject-eat-thing" [11].

[5]E.g., the word ケ ("I eat", Latinized as *k=e*) is written with a single katakana character but is glossed with two morphemes as "first-person transitive subject=eat" [8].

[6]In fact, three bytes are required to encode a single katakana character in UTF-8.

TABLE IV
TRAINING ARGUMENTS FOR ALL EXPERIMENTS.

Hyperparameter	Setting
learning_rate	See Table V
weight_decay	See Table V
lr_scheduler_type	See Table V
evaluation_strategy	'epoch'
save_strategy	'epoch'
per_device_train_batch_size	4
per_device_eval_batch_size	4
save_total_limit	1
load_best_model_at_end	True
num_train_epochs	10
predict_with_generate	True
generation_max_length[5]	516
All other parameters	default

We initialized ByT5-small with a language modeling head using the *T5ForConditionalGeneration* class in the Hugging Face Transformers library [12]. Although ByT5 does not require a tokenizer for regular use, the Hugging Face model card [13] recommends using a tokenizer for padding during batched training. As such, we using the Transformers *AutoTokenizer* class as proposed in the model card.

C. Fine-Tuning

We produced a total of 34 fine-tuned models using the Trainer API from version 4.40.2 of the Hugging Face Transformers library [12] and Python version 3.11.4. We padded or truncated every input sequence to 516[7] bytes, and set the fixed training hyperparameters via the Transformers *Seq2SeqTrainingArguments* class as shown in Tables IV and V.

As noted in the footnotes for Table V, some pairs of schedulers are effectively identical due to unmodified default hyperparameters. As a result, although eight scheduler types are listed, there were effectively only five distinct types in our experiments. However, we have included the results of each experiment in the table because the evaluation metrics differed between most of these pairs. This was possible because the 'full_determinism' attribute for the Transformers *Seq2SeqTrainer* class was left at its default value of *False*.

[7]We intended to set the input length to 512, a power of 2. However, it was set to 516 by mistake and left at that value for consistency across all experiments.

TABLE V
RESULTANT CHARACTER (CTER) [14], CER [15], AND CHRF [16] TEST SET METRICS AS A FUNCTION OF LEARNING RATE (LR), WEIGHT DECAY (WD), AND SCHEDULER TYPE (ST)

LR	WD	ST	cTER	CER	chrF
$48 * 10^{-6}$	0.01	cosine w/ restarts[8]	0.0067	0.0063	98.59
$48 * 10^{-6}$	0.1	polynomial[9]	0.0067	0.0063	98.63
$48 * 10^{-6}$	0.01	linear	0.0068	0.0066	98.58
$48 * 10^{-6}$	0.1	cosine w/ restarts[6]	0.0068	0.0065	98.56
$52 * 10^{-6}$	0.01	reduce on plateau	0.0068	0.0065	98.55
$48 * 10^{-6}$	0.1	linear	0.0068	0.0065	98.58
$48 * 10^{-6}$	0.01	constant	0.0069	0.0067	98.53
$48 * 10^{-6}$	0.1	constant	0.0069	0.0067	98.54
$48 * 10^{-6}$	0.01	const. w/ warmup[10]	0.0069	0.0065	98.56
$48 * 10^{-6}$	0.01	reduce on plateau[11]	0.0069	0.0065	98.56
$48 * 10^{-6}$	0.01	polynomial[7]	0.0070	0.0067	98.53
$56 * 10^{-6}$	0.01	linear	0.0070	0.0067	98.51
$52 * 10^{-6}$	0.01	linear	0.0070	0.0068	98.53
$56 * 10^{-6}$	0.1	reduce on plateau	0.0070	0.0069	98.50
$48 * 10^{-6}$	10^{-4}	linear	0.0071	0.0066	98.55
$48 * 10^{-6}$	10^{-5}	linear	0.0071	0.0066	98.55
$56 * 10^{-6}$	0.01	reduce on plateau	0.0071	0.0067	98.56
$48 * 10^{-6}$	0.1	inverse sqrt	0.0071	0.0066	98.53
$52 * 10^{-6}$	0.1	linear	0.0072	0.0072	98.43
$48 * 10^{-6}$	0.001	linear	0.0072	0.0070	98.51
$48 * 10^{-6}$	0.1	const. w/ warmup[8]	0.0073	0.0070	98.39
$48 * 10^{-6}$	0.1	reduce on plateau[9]	0.0073	0.0070	98.39
$44 * 10^{-6}$	0.01	linear	0.0073	0.0071	98.46
$48 * 10^{-6}$	0.3	linear	0.0073	0.0070	98.43
$36 * 10^{-6}$	0.01	linear	0.0074	0.0072	98.42
$48 * 10^{-6}$	0.01	inverse sqrt	0.0074	0.0071	98.44
$48 * 10^{-6}$	0.5	linear	0.0074	0.0074	98.40
$40 * 10^{-6}$	0.01	linear	0.0075	0.0072	98.44
$52 * 10^{-6}$	0.1	reduce on plateau	0.0075	0.0072	98.43
$48 * 10^{-6}$	0.01	cosine	0.0076	0.0073	98.40
$48 * 10^{-6}$	0.1	cosine	0.0077	0.0076	98.36
$44 * 10^{-6}$	0.1	linear	0.0079	0.0080	98.24
$28 * 10^{-6}$	0.01	linear	0.0083	0.0081	98.25
$24 * 10^{-6}$	0.01	linear	0.0090	0.0088	98.05
$32 * 10^{-6}$	0.01	linear	0.0091	0.0092	97.87
$20 * 10^{-6}$	0.01	linear	0.0095	0.0093	97.92
$16 * 10^{-6}$	0.01	linear	0.0112	0.0112	97.46
$12 * 10^{-6}$	0.01	linear	0.0121	0.0120	97.20
$8 * 10^{-6}$	0.01	linear	0.0141	0.0143	96.56
$4 * 10^{-6}$	0.01	linear	0.0214	0.0215	94.90

D. Evaluation

We selected characTER as the primary metric to compare the 40 fine-tuned models to each other. The characTER metric "is specified as the minimum number of character edits required to adjust a hypothesis, so that it absolutely matches the reference, normalized by the length of the hypothesis sentence" [14]. This is illustrated by the equation below. In addition to characTER, we present CER [15] and chrF [16], which are also character-based metrics, for each model in Table V. We selected these metrics because the task of transliteration is inherently a character-based problem, and as such these provide a more meaningful metric than other metrics such as BLEU which are based on words or larger subword units. Each of these metrics are evaluated over the test set.

$$characTER = \frac{shift\ cost\ +\ edit\ distance}{\#characters\ in\ hypothesis\ sentence}$$

The equation above is from the original characTER paper [14], in which each term is described more thoroughly than space permits for the current publication.

During training, the best epoch is chosen by the Transformers *Seq2SeqTrainer* based on validation set loss at the end of each epoch.

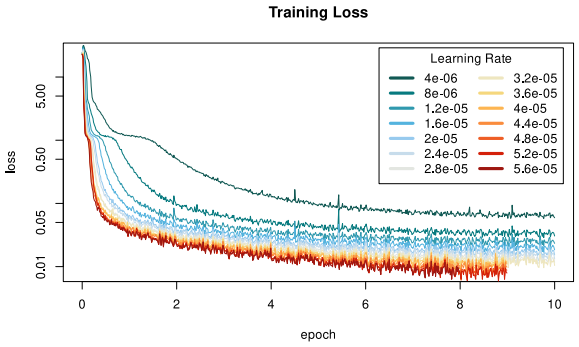

Fig. 1. Training loss for the initial learning rate sweep.

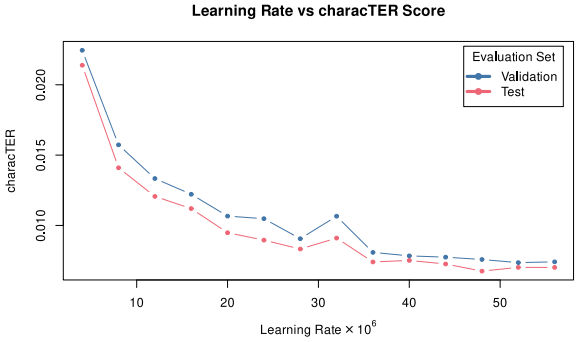

Fig. 2. The final characTER score as a function of learning rate.

V. RESULTS

Figure 1 shows the training loss for the initial learning rate sweep. Improvements diminished for learning rates above $3.6 * 10^{-5}$, after which the best epoch for each learning rate was the ninth or earlier. Figure 2 shows that the best-performing model from this sweep was produced with a learning rate of $4.8 * 10^{-5}$. The full results of each experiment are

[8]The 'num cycles' hyperparameter was left at its default value of 1, so the cosine w/ restarts scheduler was nearly identical to cosine.

[9]The 'power' hyperparameter was left at its default value of 1, so the polynomial scheduler was nearly identical to the linear scheduler.

[10]The 'warmup steps' hyperparameter was left at its default value of 0, so the const. w/ warmup scheduler was nearly identical to constant.

[11]reduce on plateau produced results exactly identical to const. w/ warmup because the learning rate was never actually reduced.

presented in Table V, sorted in descending order of performance.

VI. Discussion and Next Steps

It is first important to understand the scope of this research. There are many dialects of Ainu, and the scope of this work is limited to the Saru dialect. All Ainu dialects are low-resource, but among them Saru has relatively many available audio recordings and parallel transcriptions of those recordings that can be used in machine learning. Additionally, a significant portion of the available parallel transcriptions used in training consist of oral literature, which can differ in many ways from colloquial Ainu speech. It should not yet be assumed that any results attained for the Saru dialect will readily transfer to other dialects, nor that our models are equally performant for both literary and colloquial Saru text.

Bearing this scope mind, even the most poorly performing models we produced in this research are accurate enough for practical use according to each of the evaluation metrics. For comparison, we have evaluated the *ainconv* Rust package developed by Cjyet Yo of Osaka University to explore the limits of rules-based approaches to Ainu transliteration [6]. The results are presented in Table II.

The results show a significant improvement over a rules-based approach for Ainu transliteration, although there is still a higher error rate than would be expected of an expert human transliterator. We believe that higher accuracy can still be attained through further preprocessing of the corpus. Even after the normalization step, there are a number of conventions that are inconsistent throughout the corpus, such as whether or not the personal affixes *a* and *i* are marked (i.e., *a*= and *i*=).

Although performance is lower than an expert human, we believe that this tool and its future iterations can greatly reduce the workload of experts transliterating Ainu text from katakana to Latin script. Additionally, this will contribute to the expansion of the Latinized Ainu corpus for use and application in future NLP research.

At time of publication, we are reviewing all "error" output, i.e. outputs which do not match the original human transcriptions within the corpus. With the help of Ainu language experts including co-author Yamamaru, we will classify every "error" as either a valid or invalid transcription. Because there are some known errors in the digitized corpus, we will likewise classify the corresponding original transcriptions as either valid or invalid. This will help identify patterns within the invalid outputs and inform future improvements to the corpus and trained models.

VII. Model Repositories

The top three performing models (ranked by characTER scores as shown in Table V) may be found at the following Hugging Face repositories:

- TwentyNine/byt5-small-ainu-latinizer-cos_w_restarts
- TwentyNine/byt5-small-ainu-latinizer-polynomial
- TwentyNine/byt5-small-ainu-latinizer-linear

All three may be tested directly in the browser via the following Hugging Face Space:

- https://huggingface.co/spaces/TwentyNine/byt5-ain-kana-latin-converter

VIII. Acknowledgments

The corresponding author extends gratitude to his coauthors, whose guidance and advice have been and continue to be essential to every stage of research. He thanks Drs. Ryoko Okamura of Bowling Green State University and Yukie Saito of Utah Tech University, and all of his former instructors at Shinshu University, without whose guidance and instruction he would not have entered this field of research. Finally, he thanks Dr. Anna Bugaeva of Tokyo University of Science for recommending his current advisor.

References

[1] UNESCO, ed., *Atlas of the world's languages in danger*. Memory of Peoples Series, UNESCO, 3 ed., Feb. 2010.

[2] The Foundation for Ainu Culture, "アイヌ語入門講座 (Beginner Ainu Courses)." Available at https://www.ff-ainu.or.jp/web/session/files/r06_02shiraoi.pdf.

[3] STV Radio, "アイヌ語ラジオ講座 (Ainu Language Radio Course)." Available at https://www.stv.jp/radio/ainugo/index.html.

[4] The Foundation for Ainu Culture, "アイヌ語弁論大会〜イタカンロー〜(ainu language speech contest itak=anro)." Available at https://www.ff-ainu.or.jp/web/learn/language/itakanro/.

[5] L. Xue, A. Barua, N. Constant, R. Al-Rfou, S. Narang, M. Kale, A. Roberts, and C. Raffel, "ByT5: Towards a token-free future with pre-trained byte-to-byte models," *Transactions of the Association for Computational Linguistics*, vol. 10, pp. 291–306, 2022.

[6] C. Yo, "Pushing the boundaries of rule-based processing for ainu texts," Tech. Rep. 9, Osaka University Graduate School of Humanities, May 2024.

[7] S. Tamura, アイヌ語沙流方言辞典 *(Dictionary of the Saru Dialect of the Ainu Language)*. 株式会社草風館 (Soufuukan K.K.), 1996.

[8] National Institute for Japanese Language and Linguistics (NINJAL), "A topical dictionary of conversational ainu." Available at https://ainu.ninjal.ac.jp/topic/, 2015. [Software].

[9] K. Nowakowski, M. Ptaszynski, and F. Masui, "A proposal for a unified corpus of the ainu language," Tech. Rep. 2, Kitami Institute of Technology, Kitami Institute of Technology, Kitami Institute of Technology, sep 2018.

[10] H. Nakagawa, ニューエクスプレスアイヌ語 *(New Express Ainu Language)*. 株式会社白水社 (Hakusuisha Publishing Co., Ltd.), Nov. 2013.

[11] National Institute for Japanese Language and Linguistics (NINJAL), "A glossed audio corpus of ainu folklore." Available at https://ainu.ninjal.ac.jp/folklore/, 2016-2024. [Software].

[12] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, and A. M. Rush, "Transformers: State-of-the-art natural language processing," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, (Online), pp. 38–45, Association for Computational Linguistics, Oct. 2020.

[13] Google, "byt5-small." Available at https://huggingface.co/google/byt5-small, 2015.

[14] W. Wang, J.-T. Peter, H. Rosendahl, and H. Ney, "CharacTer: Translation edit rate on character level," in *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, (Berlin, Germany), pp. 505–510, Association for Computational Linguistics, Aug. 2016.

[15] A. Morris, V. Maier, and P. Green, "From wer and ril to mer and wil: improved evaluation measures for connected speech recognition.," 01 2004.

[16] M. Popović, "chrF: character n-gram F-score for automatic MT evaluation," in *Proceedings of the Tenth Workshop on Statistical Machine Translation*, (Lisbon, Portugal), pp. 392–395, Association for Computational Linguistics, Sept. 2015.