Back-Transliteration of English loanwords in Japanese

Yuying Ren
The Graduate Center, City University of New York

Abstract

We propose methods for transliterating English loanwords in Japanese from their Japanese written form (katakana/romaji) to their original English written form. Our data is a Japanese-English loanwords dictionary that we have created ourselves. We employ two approaches: direct transliteration, which directly converts words from katakana to English, and indirect transliteration, which utilizes the English pronunciation as a means to convert katakana words into their corresponding English sound representations, which are subsequently converted into English words. Additionally, we compare the effectiveness of using katakana versus romaji as input characters. We develop 6 models of 2 types for our experiments: one with an English lexicon-filter, and the other without. For each type, we built 3 models, including a pair n-gram based on WFSTs and two sequence-to-sequence models leveraging LSTM and transformer. Our best performing model was the pair n-gram model with a lexicon-filter, directly transliterating from katakana to English.

1 Introduction

Loanwords have grown at a rapid pace in Japanese language since 1990s. English loanwords make up 8 percent of the Japanese vocabulary and 94 percent of all loanwords used in Japanese (Stanlaw, 2004). The excessive use of loanwords in mass media not only poses difficulties for Japanese to understand their own language (Irwin, 2011), but also creates callenges for English speakers to accurately back-transliterate the loanwords due to the significant differences in sound and written representation from their original forms.

Knight and Graehl (1998) utilized estimation-maximization to establish a mapping of the similarity between English and Japanese sounds. Among 38 phonemes they have examined, only 5 had a corresponding Japanese sound with a probability greater than .9, and 10 of them reached a probability of .8.

In terms of writing systems, Japanese has a relatively complex system. Japanese uses three sets of characters: hiragana, kanji, and katakana. Katakana is mainly used for writing foreign words and over 100 of these characters are in use. Moreover, unlike English, Japanese characters represent sounds syllabically instead of phonetically (DeFrancis, 1989). Romaji, another set of characters is informally used in Japanese, represents the Romanization of katakana. It was originally used to annotate the sounds of Japanese characters but has gained popularity as a means of typing Japanese using keyboards.

As described by Knight and Graehl (1998), the transliteration of English words to katakana is an information-losing operation from both the sound and writing system perspectives. For instance, because there is no distinction between sounds of / æ, ʌ / and / θ, s / in Japanese, the English words *bath* and *bus* are mapped to the same form in katakana: バス<ba-su>. In contrast, the word *camera* has two corresponding katakana forms: カメラ <ka-me-ra> or キャメラ <kya-me-ra>.

Due to the loss of information, back-transliteration becomes even more challenging. Nonetheless, in recent years, an increasing number of researchers have been employing NLP methods to address this issue. Many studies tackle the back-transliteration challenge as a Grapheme-to-Phoneme (G2P) problem (e.g., Jiampojamarn et al. 2010; Rosca and Breuel, 2016; Merhav and Ash, 2018), utilizing models such as WFST-based n-gram models or neural sequence-to-sequence models, which are commonly employed for the

Proceedings of the Workshop on Computation and Written Language (CAWL 2023), pages 43–49

G2P tasks (e.g., Novak et al. 2016; Gorman et al., 2020), to address the issue.

In this paper, we approach the back-transliteration problem by building models that investigate the impact of the sound information and the use of either katakana or romaji as the input for Japanese. In addition to utilizing the n-gram and sequence-to-sequence models typically employed in G2P tasks, we introduce a novel approach by incorporating an English lexicon-filter mechanism. We expect this technique to help us generate more relevant outputs.

2 Related Work

Knight and Graehl (1998) is one of earliest works on transliteration and back-transliteration of Japanese loanwords. They utilize WFSTs to build a modular system that transliterate the katakana words to their original English forms using the sound of English words. They test their system on two relatively small datasets: a content words dictionary with 1,449 katakana-English pairs and a name list with 100 pairs. The accuracy of their method outperforms that of human translators.

Yamashita et al. (2018) not only uses phonemes to map katakana to English but also directly uses characters. They employ a bidirectional recurrent neural network (RNN), trained on a katakana-English content words dictionary, and experiment with three test datasets: a content word list, a city name list, and a restaurant name list. They use 5 similarity algorithms evaluate their model and report the top-five precision of each measurement. Interestingly, they find that the direct mapping using characters performs better than using phonemes as medium for all their measurements.

Merhav and Ash (2018)'s research primarily focusses on the transliteration of named entities from English to katakana. They employ a n-gram model with the `Phonetisaurus` library (Novak et al. 2016), an RNN model with the `seq2seq` library (Luong et al. 2017) and a transformer model with `tensor2tensor` library (Vaswani et al. 2018). They apply their transformer model to the back-transliteration task, which outperforms the other two models. They report the 1-best, 2-best, and 3-best word error rate (WER) for evaluation and the WER of the back-transliteration are averagely .2 points higher than the transliteration.

For our experiments with the no-lexicon-filter models, we adapt the implementation of the baselines in 2020 SIGMORPHON shared task[1] (Gorman et al., 2020), which consist of 3 models: a pair n-gram model built using OpenGrm toolkit (Roark et al. 2012, Gorman, 2016), and two sequence-to-sequence models with LSTM (Luong et al. 2015) and transformer (Vaswani et al. 2017) architectures that are built with the `fairseq` (Ott, et al. 2019) library.

3 Data

Our dataset is compiled from three dictionaries. First, we use the JMdict[2] (Breen, 1995), a product of the Japanese-Multilingual electronic dictionary project to extract the katakana-English word pairs. Second, we incorporate the CMUdict[3] (Weide, 2014), which provides the pronunciation of the English words. Finally, we utilize the Webster's Dictionary (Neilson and Knott, 1934) and the CMUdict to build the English lexicon-filter.

To construct our dataset, we first filter out non-loanwords such as onomatopoeias and then manually expand the abbreviated katakana words. For example, the word アメフト<a-me-fu-to> 'Ame foot', is extended to アメリカンフットボール<a-me-ri-ka-fu-to-bo-o-ru> 'American football'. Next, we utilize the CMUdict to map the sounds of English words and pair them with the corresponding katakana words. Finally, for the purpose of our experiment, we add a column of Romanized katakana words to our data by using a python library `romkan`[4]. A sample of our final dataset is shown in Table 1.

Our dataset consists 26,208 items, which we randomly divided into train, dev, and test sets. The proportions of the three sets are 80%, 10%, and 10%, respectively. It is noteworthy that 47.2% of our data consist of katakana words are mapped to multiple-word expressions in English, such as the example of American football mentioned above.

Finally, we merge the CMUdict (Weide, 2014) and Webster's dictionary (Neilson and Knott, 1934) to form an English wordlist with over 320k distinct words for building the lexicon-filter models.

[1] https://github.com/sigmorphon/2020.

[2] https://www.edrdg.org/jmdict/jmdict_whatsnew.html.

[3] http://www.speech.cs.cmu.edu/cgi-bin/cmudict.

[4] https://pypi.org/project/romkan.

katakana	romaji	English	CMU
スイーパー	suiipaa	sweeper	S W IY P ER
テニスエルボー	tenisueruboo	tennis elbow	T EH N AH S EH L B OW
...	...	...	...

Table 1. Samples of final dataset. The CMU column contains the English pronunciations.

4 Methods

4.1 Approaches

Both the direct and indirect transliteration approaches will explore the impact of using different Japanese characters: katakana and romaji as the inputs. In the direct approach, words are directly converted from their Japanese writing forms to their English forms. On the other hand, the indirect approach can be implemented in various ways. We trained models on katakana-phonemes and romaji-phonemes data, which are utilized to predict the possible pronunciations for English words in the first step. Subsequently, we train models using the phoneme-English data from the CMUdict (Weide, 2014) and employ them and the phoneme results from the first step to predict the final words in English. Figure 1 illustrates the difference between using katakana and romaji for indirect and direct approaches.

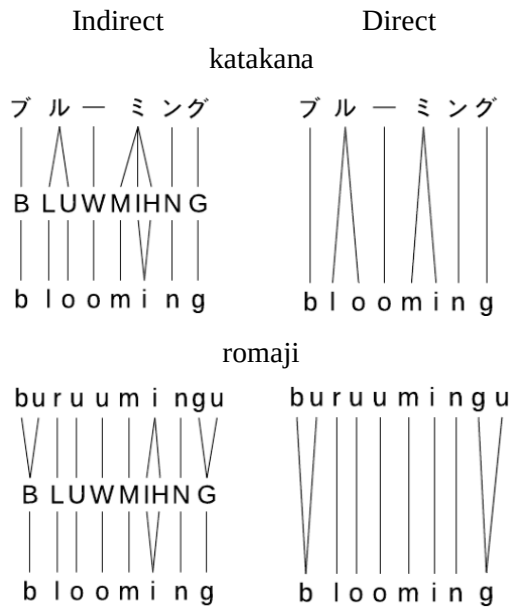

Figure 1. Demonstrations of difference between using katakana and romaji as the input for indirect (left) and direct (right) approaches.

4.2 Models

We create 2 types of models: models with an English lexicon-filter, and models without an English lexicon-filter. Within each type, we have three models, namely a pair n-gram model, and two neural sequence-to-sequence models with LSTM and transformer (Gorman et al., 2020).

The architecture of the pair n-gram model is similar to the architecture of Phonetisaurus toolkit (Novak, 2016), but it is implemented with the libraries of Pynini (Gorman, 2016), Baum-Welch, and NGram (Roark et al. 2012). We use this model to train an aligner based on WFSTs that maps katakana or romaji characters to English characters. Next, the alignments are then used to compute a higher-order n-gram model (we set the order to 8), which is converted to a final WFST. The resulting WFST can take the input words written either in katakana or romaji, and produce a weighted lattice of possible English words, the prediction is made by selecting the shortest path/s through the lattice.

The neural network models are implemented using the `fairseq` (Ott, et al. 2019) library. The LSTM-based model contains a bidirectional LSTM encoder with a single layer and a unidirectional LSTM decoder with a single layer (Luong et al., 2015). The transformer-based model (Vaswani et al. 2017) contains 4 encoder and 4 decoder layers, and both tuned using Wu et al. (2020)'s pre-layer normalization method. The two models share most of the training hyperparameters, such as the Adam optimizer (Kingma and Ba, 2015), and label-smoothed cross-entropy for regularization. We tune the models on the development dataset, with different learning rates of .001 and .005, batch sizes of 128, 256, 512, and embedding layer dimensions of 128, 256, and hidden layer units of 512, 1024. We perform early stopping in a similar way to Gorman et al. (2020), that we save every 5 checkpoints, and use the checkpoint that reached

Models		Indirect		Direct	
		katakana	romaji	katakana	romaji
LSTM	No lexicon-filter	38.38	37.96	33.12	33.42
	Lexicon-filter	34.41	33.93	26.33	25.91
Transformer	No lexicon-filter	46.70	48.95	34.30	35.41
	Lexicon-filter	41.17	43.99	24.95	25.60
Pair n-gram	No lexicon-filter	34.38	35.83	34.26	35.02
	Lexicon-filter	**30.64**	31.59	**23.01**	25.41

Table 2. WER results for all experiments.

the lowest word error rate (WER) on the development set to predict on the test set.

The implementation of the lexicon filter differs between the pair n-gram model and the sequence-to-sequence models. In the case of the pair n-gram model, we construct a FST utilizing the English wordlist data and then compose it with the output lattices. This process enables us to eliminate any predictions that are not present in the English wordlist. On the other hand, for the neural network models, we extract the top-5 predictions and use the English wordlist to filter out the predictions that include non-existent words. In all models, we retain the original outputs if all the hypotheses contain non-existent word.

5 Evaluation

We evaluate our models by reporting the word error rate (WER), which represents the percentage of predicted words that differ from the target words. A lower WER value indicates a better performance. We consider the multi-word expressions as a single entity during our evaluation, meaning that any incorrect prediction of a word in the expression results in the entire prediction being considered incorrect. Additionally, as the selection of random seed values can non-trivially affect the model's performance (Reimers and Gurevych, 2017), we opted to train each of our models with five different random seeds and present the median value of the resulting five WERs.

6 Results

Table 2 displays the results of all experiments, which demonstrate that the indirect approach performs worse than the direct approach. This finding is consistent with the results reported by Yamashita et al. (2018). However, the difference in WERs between using katakana and romaji as the input are insignificant for either approach.

In addition, it is worth noting that while the transformer models with lexicon-filter have shown better performance compared to the LSTM-based models in the direct transliteration experiments, the pair n-gram model surpassed them by a reduction of 2 to 3 points in terms of WER. This is noteworthy as transformer has generally outperformed other two models in previous G2P tasks, as well as the named entity recognition task by Merhav and Ash (2018).

Finally, the results demonstrate a significant reduction in WER for models with lexicon-filters compared to those without. Particularly for experiments with the direct approach, show a reduction of average 10 points for all models. The pair n-gram model with lexicon-filter that directly transliterate katakana to English proved to be the most robust, which achieves a WER of 23.01.

7 Discussion

Different designs in the indirect approach can yield different outcomes. Our chosen design for the experiment inherently introduces noise to the models during the conversion of phonemes to English words. Table 3 displays the WER results of experiments where words in katakana or romaji are converted into their corresponding English phonemes. The relatively high scores implies that the phonemes utilized for predicting English words, generated from the Japanese data, can significantly differ from the data used to train the phoneme-English conversion models. Yamashita et al. (2018) compared this design with an alternative approach where both Japanese and English words were converted to phonemes, and the similarity between the results was measured, which yielded better results in their study.

	katakana	romaji
LSTM	34.41	**33.65**
Transformer	42.92	46.62
Pair n-gram	**32.32**	34.26

Table 3. WER results of experiments converting Japanese loanwords to their corresponding English phonemes.

In order to assess the impact of different input characters and model architectures on solving the back-transliteration problem, we perform McNemar's tests[5] (Gillick and Cox, 1989) on the results obtained from the corresponding experiments. The null hypothesis in McNemar's test states that the two hypotheses exhibit equal accuracy and performance. In our case, we failed to reject the null hypothesis when comparing the use katakana and romaji as input characters, as well as when comparing models with lexicon-filter in the direct transliteration approach. However, it is interesting to find that pair n-gram model has surpassed the sequence-to-sequence models. Merhav and Ash (2018) has surprisingly found that their transformer model, which is typically known for its ability to handle long term dependencies, outperformed their WFST-based n-gram model in their named entity transliteration task with relatively small input sizes. Based on this finding, we divide our results into two categories: small and large input size, using the median as the threshold. We then compare the WER of the transformer and pair n-gram models within each category. We observe that while the two models exhibited similar WER on the small input size data, there was a significant difference on the large input size data, where the pair n-gram model outperformed the transformer model with a WER that was approximately 10 percent lower.

Upon analyzing the errors in our predictions, we examine the results generated by the pair n-gram model with the lexicon filter that used katakana as input. We identify two major types of the errors: the spelling errors and the word delimiter errors. Some of the spelling errors are attributed to the phonetic distinctions between Japanese and English, as discussed previously in this paper. For instance, the word *lighter* is predicted as *writer* due to the lack of distinguish between the sounds of / l, ɹ / in Japanese. Other spelling errors can arise from the English homophones such as the target-hypothesis pair of *site* and *sight*.

Word delimiter errors occur when the predicted words are correct, but the position of the whitespace is incorrect, such as the pairs of *fireman* and *fire man* or *homegrown terror* and *home grown terror*. These errors account for 10% of the total errors and accepting them could reduce the WER by 2 to 3 points.

8 Conclusion

Our study investigated the factors affecting the back-transliteration of English loanwords in Japanese. Specifically, we constructed models to compare the use of characters for direct transliteration versus the use of sounds as a medium, as well as the use of katakana versus romaji as input sources. We built 6 models with 2 types: models with lexicon-filter and those without lexicon-filter. For each type, we built two neural sequence-to-sequence models as well as a pair n-gram model. Our results revealed that models with lexicon-filter exhibited significant improvement in performance, with an average reduction of 10 points in WER. The most robust model we achieved was the pair n-gram model with lexicon-filter for the katakana-to-English transliteration, which produced a WER of 23.01. There are some areas for potential improvement in the future, such as integrating spelling correction models and incorporating word frequency computation. Moreover, we envision the integration of our model to address other challenges, including machine translation and entity matching.

Limitations

There remains a problem we have yet to address: the abbreviation of loanwords in Japanese. Japanese often abbreviates multi-word expressions after transliterating them into katakana. For example, スマートホン <su-ma-a-to-ho-n> 'smart phone' becomes スマホ <su-ma-ho>. For our method, we manually extended these abbreviated words to their full forms, but automating this process would be preferable due to the prevalence of these words in Japanese. However, back-

[5] We adapt Gorman and Bedrick's implementation of the test: https://github.com/kylebgorman/SOTA-taggers.

transliterating them presents challenges as they deviate further from their original English forms.

We designed our method to specifically focus on back-transliterating of content words, unlike many other studies that focused on the name entities data. This is because the loanwords of content words are prevalent in Japanese. However, names are also challenging as they are in other languages. Previous studies have suggested that a more sophisticated method may be necessary for back-transliterating names.

Acknowledgment

I would like to express my sincere gratitude to Dr. Kyle Gorman, who has generously shared his invaluable knowledge with me throughout this project. His expertise and insights have been instrumental in shaping the direction and improving the outcomes of this research.

Ethics Statement

The data used in this study are collected from sources that are publicly available and has been used in accordance with the terms of the respective licenses.

References

Jim Breen. 1995. Building an Electronic Japanese-English dictionary. In Japanese Studies Association of Australia Conference. Citeseer.

Brian Roark, Richard Sproat, Cyril Allauzen, Michael Riley, Jeffrey Sorensen, and Terry Tai. 2012. The OpenGrm open-source finite-state grammar software libraries. In Proceedings of the ACL 2012 System Demonstrations, pages 61–66, Jeju Island, Korea. Association for Computational Linguistics.

DeFrancis, J. (1989). *Visible speech: The diverse oneness of writing systems*. University of Hawaii Press.

Gillick, L., & Cox, S.J. (1989). Some statistical issues in the comparison of speech recognition algorithms. International Conference on Acoustics, Speech, and Signal Processing, 532-535 vol.1.

Irwin, Mark. (2011). Loanwords in Japanese. Amsterdam; Philadelphia: John Benjamins Pub. Co.

Jackson L. Lee, Lucas F.E. Ashby, M. Elizabeth Garza, Yeonju Lee-Sikka, Sean Miller, Alan Wong, Arya D. McCarthy, and Kyle Gorman. 2020. Massively Multilingual Pronunciation Modeling with WikiPron. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 4223–4228, Marseille, France. European Language Resources Association.

Kevin Knight and Jonathan Graehl. 1997. Machine Transliteration. In 35th Annual Meeting of the Association for Computational Linguistics and 8th Conference of the European Chapter of the Association for Computational Linguistics, pages 128–135, Madrid, Spain. Association for Computational Linguistics.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Kyle Gorman. 2016. Pynini: A Python library for weighted finite-state grammar compilation. In *Proceedings of the SIGFSM Workshop on Statistical NLP and Weighted Automata*, pages 75–80, Berlin, Germany. Association for Computational Linguistics.

Kyle Gorman, Lucas F.E. Ashby, Aaron Goyzueta, Arya McCarthy, Shijie Wu, and Daniel You. 2020. The SIGMORPHON 2020 Shared Task on Multilingual Grapheme-to-Phoneme Conversion. In Proceedings of the 17th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology, pages 40–50, Online. Association for Computational Linguistics.

Michiharu Yamashita, Hideki Awashima, and Hidekazu Oiwa. 2018. A Comparison of Entity Matching Methods between English and Japanese Katakana. In Proceedings of the Fifteenth Workshop on Computational Research in Phonetics, Phonology, and Morphology, pages 84–92, Brussels, Belgium. Association for Computational Linguistics.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A Fast, Extensible Toolkit for Sequence Modeling. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations), pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Novak, J. R., Minematsu, N., & Hirose, K. (2016). Phonetisaurus: Exploring grapheme-to-phoneme conversion with joint n-gram models in the WFST framework. Natural Language Engineering, 22(6), 907-938.

Nils Reimers and Iryna Gurevych. 2017. Reporting Score Distributions Makes a Difference: Performance Study of LSTM-networks for Sequence Tagging. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pages 338–348, Copenhagen, Denmark. Association for Computational Linguistics.

Rosca, M., & Breuel, T. (2016). Sequence-to-sequence neural network models for transliteration. arXiv preprint arXiv:1610.09565.

Shijie Wu, Ryan Cotterell, and Mans Hulden. 2020. Applying the transformer to character-level transduction. arXiv:2005.10213.

Sittichai Jiampojamarn, Kenneth Dwyer, Shane Bergsma, Aditya Bhargava, Qing Dou, Mi-Young Kim, and Grzegorz Kondrak. 2010. Transliteration Generation and Mining with Limited Training Resources. In *Proceedings of the 2010 Named Entities Workshop*, pages 39–47, Uppsala, Sweden. Association for Computational Linguistics.

Stanlaw, J. (2004). Japanese English: Language and Culture Contact. Hong Kong University Press.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective Approaches to Attention-based Neural Machine Translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal. Association for Computational Linguistics.

Vaswani, A., Bengio, S., Brevdo, E., Chollet, F., Gomez, A. N., Gouws, S., ... & Uszkoreit, J. (2018). Tensor2tensor for neural machine translation. *arXiv preprint arXiv:1803.07416*.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. Advances in neural information processing systems, 30.

Weide, R. (2014). The CMU pronunciation dictionary, release 0.7b.

Neilson, W.A. and Knott, T.A (editors). 1934. *Webster's New International Dictionary*. 2nd edition. G. & C. Merriam.

Yuval Merhav and Stephen Ash. 2018. Design Challenges in Named Entity Transliteration. In Proceedings of the 27th International Conference on Computational Linguistics, pages 630–640, Santa Fe, New Mexico, USA. Association for Computational Linguistics.