Kana2Kanji Report

Brandon Benoit (brandonbenoit@u.boisestate.edu),
Will Lawrence (willlawrence1@u.boisestate.edu)

April 2023

Abstract

Japanese is a complex language that can be inherently difficult to navigate. We present a tool to mitigate such challenges by converting Japanese kana to its respective kanji. Primarily using a text-to-text language generation model, we are able to fine-tune an existing model to aid educators and learners of the Japanese language.

1 Introduction

The Japanese language has, in short, three alphabets. Hiragana, which is used for native Japanese words, katakana, which takes from other languages, and kanji, which are Chinese characters that adapt to Japanese phonetics. Kanji function as symbols and can have a set of hiragana (kana) to denote its phonetics. In addition to this, a kanji's phonetics may change based on what neighbors it. Thus, the purpose of this assignment is to take Japanese hiragana and automatically convert to Japanese kanji to handle the ambiguity.

Since Japanese sentences can be very complex, this sort of conversion can be very useful in educational settings. If a student does not know a certain kanji, no alphabet exists to help them phonetically get through the reading. So, as

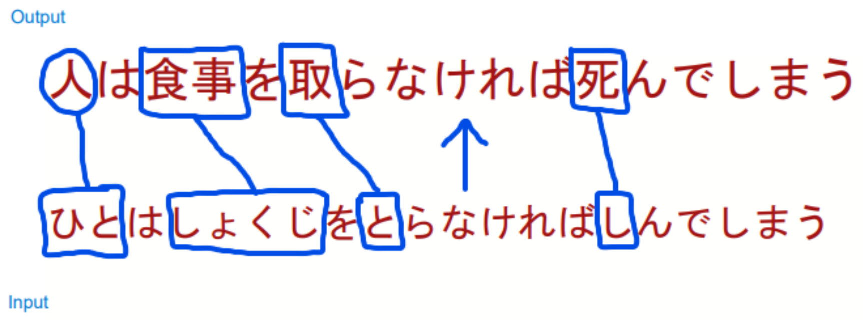
Output
人は食事を取らなければ死んでしまう
ひとはしょくじをとらなければしんでしまう
Input

Figure 1: Kana input to kanji example

students learn new kanji, they could automatically adjust sentences to match their vocabulary ensuring they only face sentences they should be able to read. This is additionally relevant to the Japanese-Language Proficiency Test (JLPT) which is the Japanese standard of language proficiency evaluation. It's levels range from N1-N5 (difficult-easy) where each level has a list of associated kanji. Students learning N4, for example, would be able to automatically attach N4 and N5 level kanji to their sentences. If they cannot read certain sentences, this would indicate what kanji or vocabulary to study.

Many uses may be found with this tool, but this is just a start to working with the Japanese language and machine learning.

For this project, we attempted a few different methods such as using a tagger to nearly compose correct kanji conversions, masking with Bert, English to Japanese translate to practice with Google's T5 text-to-text model, and utilizing Dr. Kennington's sample code also using the text-to-text model. For the purpose of this report, any resources or lessons will be generic unless in direct relation to the final T5 model method.

2 Background

Many models and tokenizers are trained on English, and other languages are underdeveloped in comparison. For example, on Huggingface alone, English boasts its participation in 3,132 data sets. Meanwhile, Japanese is found in 163. This can add some additional complications when working with Japanese. The added complexity of the Japanese language affects the dataset as well. In English, the word "horse" is only "horse". In contrast, in Japanese, "horse" is "うま (uma)" which can also be represented with its kanji, "馬 (uma)". For one word in English, we have two representations in Japanese. The shortcomings of apt Japanese data sets becomes more glaring since the inclusion of these two (or more) representations may be lost.

We investigated the cl-tohoku/bert-base-japanese model which is a pre-trained Japanese BERT model. In additon, we also used Google's T5 (specifically, mt5-small) .[1]and sonoisa/t5-base-japanese-v1.1.[2]

3 Data

A similar drawback to limited models, data sets lack the kanji and only kana variations of Japanese sentences. So, for initial testing, we used universal dependencies's ja-gsd subset. [3] This dataset features several Japanese sentences with various forms of information including their tokens, lemmas, pos, and more. For the translation task, we used opus100's en-ja subset which has English sentences associated with Japanese sentences. Later, we created a small list of sentences featuring their kanji and kana only variations.

[1]`https://huggingface.co/google/mt5-small`
[2]`https://huggingface.co/sonoisa/t5-base-japanese-v1.1`
[3]`https://huggingface.co/datasets/universal_dependencies`

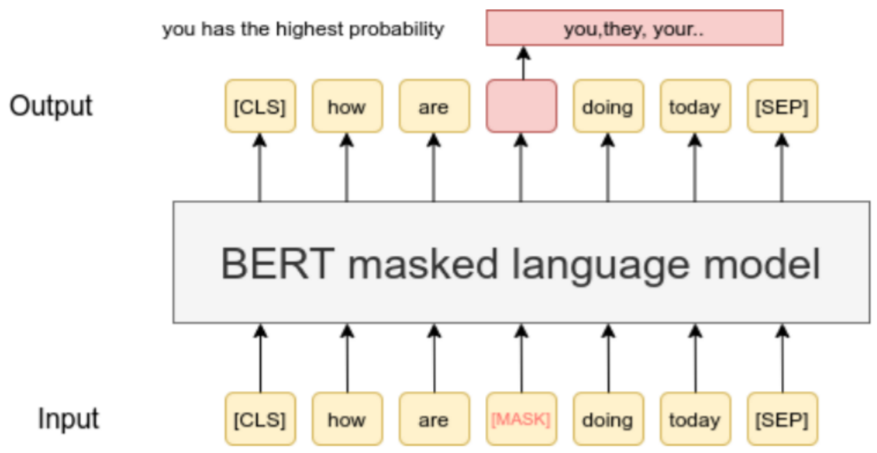

Figure 2: Masked Language Model

4 Method and Approach

Approach number one featured the fugashi tagger that allowed us to take our input from the sentence level to word level. From here we could take the words and match them with the best possible kanji. We can do all of this from the fugashi tagger as the result of its parsing gives us an associated kanji character.

Approach number two focused around a masked language model using BERT. This model masks some inputs and tries to predict the masked values. This method seemed logical since we could mask kana and get the kanji output. This, however, did not yield any results.

Approach number three was to re-contextualize the problem. Logically, converting kana to kanji looks like a translation problem. After following a demo to train English to French, we converted that to translate English to Japanese. This idea presented itself late in our process so this approach bore no fruit and was underdeveloped.

Approach number four presented itself late in the process as well. Dr. Kennington gave us some code that should function similar to what we needed. Approach three and four used the same method. This became mostly ”plug and play” allowing for us to attach our own dataset to the model and run it fairly easily. Despite all of the help, our model did not yield fruit here either. With less than two days to produce, we were unable to make the model output our target. The hyperparams stayed mostly consistent. The batch size changed but rested at 32 (and therefore the logging steps changed as well). Learning rate stayed consistent at 0.000056, and number of epochs totaled 3.

5 Evaluation

Our problem remains unsolved, but we made progress towards our goal. The results of our various models are interesting. The fugashi tagger had the highest ”accuracy” where we only needed to rely on whether the tagger parses the input correctly. This had to be manually calculated by comparing kanji outputs. Out of the 15 we tested, they all had more correct kanji printed than incorrect. We found that accuracy increased along with simplicity. As a solution to our problem, this worked fairly well. This did not come close to our goal nor is a machine learning task, but this performed best.

Next, the results of our T5 model utilizing Dr. Kennington’s code gave us output, but not anything close to our expectation. Using the given model, our rouge score was averaging 88, but we got no input. If we did get an input it was either a particle or a Japanese period. When replaced with sonoisa/t5-base-japanese-v1.1 pre-trained model, we got a worse rouge score of about 9 on average, but our output was a sentence. It just happened to be the same sentence as the input. At our lowest data set volume, we had the word ”りす” at the beginning of the output. As we added more data to the test set, we got an output that was the input sentence with the word ”こころ” at the end. Despite this model not giving us an output remotely close to our target, it was interesting to see the relative impact of various pre-trained models. The final test had a peak rouge1 score of 15.

6 Implications

This task could have major use in education. As an educator, instructors would be able to create any sentence without concerning themselves with kanji that the class does not know. A student, or an individual self studying, could filter their own knowledge base of kanji and prioritize their learning material. The stop and go of looking up words can interrupt productive learning. With resources like jisho.org readily available, this tool may not be so useful. Despite students being able to look up any unfamiliar word, an advantage of prioritizing on the lesson material include eliminating distractions. Motivated students would not be disabled from further research in words they are unfamilar with.

7 Conclusion

Though our model failed to meet our expectations and goal for this project, a semi-accurate system given kana as an input will print an output that features kanji. Another model, though inaccurate, takes a text-to-text generation model goes through the training steps and outputs something.

This project facilitated use of multiple natural language processing methods from analyzing a problem located in the field. Text-to-text generation and text translation were just a few of the attempted method that provided a pivotal learning experience.

Given more time or more apt preparation, we would like to try adjusting the trainer model more to start outputting kanji or attempt a new model with a completely different approach.