Sub-character Neural Language Modelling in Japanese

Viet Nguyen **Julian Brooke** **Timothy Baldwin**
School of Computing and Information Systems
The University of Melbourne

vn@student.unimelb.edu.au, jabrooke@unimelb.edu.au, tb@ldwin.net

Abstract

In East Asian languages such as Japanese and Chinese, the semantics of a character are (somewhat) reflected in its sub-character elements. This paper examines the effect of using sub-characters for language modeling in Japanese. This is achieved by decomposing characters according to a range of character decomposition datasets, and training a neural language model over variously decomposed character representations. Our results indicate that language modelling can be improved through the inclusion of sub-characters, though this result depends on a good choice of decomposition dataset and the appropriate granularity of decomposition.

1 Introduction

The Japanese language makes use of Chinese-derived ideographs ("kanji") which contain sub-character elements ("bushu") that to varying degrees reflect the semantics of the character. For example, the character 鯨 (*kujira* "whale") consists of two sub-characters: 魚 (*sakana* "fish") and 京 (*kyou* "capital city"). Similarly, the character 鰈 (*hirame* "flounder") consists of the sub-characters 魚 (*sakana* "fish") and 平 (*hira* "something broad and flat"). Here, the sub-character 魚 (*sakana* "fish") is a semantically significant element which appears in characters relating to marine life. Current Japanese language models do not capture sub-character information, and hence lack the ability to capture such generalisations.

A key limitation of word-based language modelling is the tendency to produce poor estimations for rare or OOV (out-of-vocabulary) words, and character-based language models have been shown to solve some of the sparsity problem in English by modeling how words are constructed (Graves, 2013). We take inspiration from this work, but observe for Japanese that since the kanji portion of the Japanese writing system contains thousands rather than dozens of characters, a character-based language model will still be susceptible to sparsity. Given that a large number of Japanese characters can be decomposed into sub-characters, we examine the question of whether sub-character language models can achieve similar gains in language model quality to character language models in English.

In this paper we train sub-character language models for Japanese based on decompositions available in several existing kanji datasets. Our results suggest that decomposing characters is of value, but that the results are sensitive to the nature and granularity of the decomposition.

2 Kanji Datasets

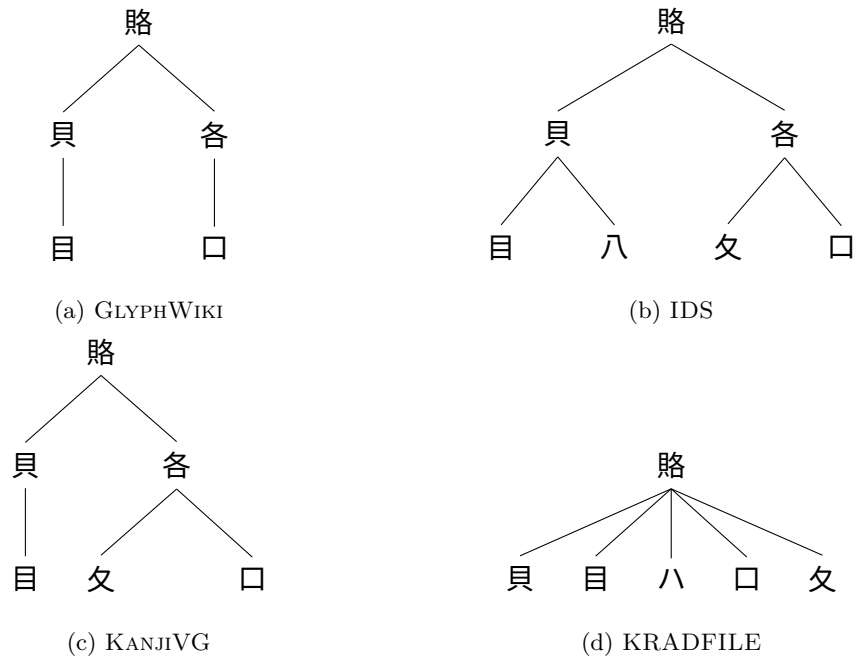
In order to investigate the usefulness of sub-character decomposition for language models, we need some way of deriving these bushu from kanji. Here, we consider four kanji datasets that provide decompositions for kanji characters: GlyphWiki, IDS, KanjiVG, and KRADFILE. An example kanji decomposition under the four datasets is provided in Figure 1.

2.1 GlyphWiki

GlyphWiki[1] is a community-driven wiki that stores information about kanji characters

[1] http://glyphwiki.org

Proceedings of the First Workshop on Subword and Character Level Models in NLP, pages 148–153,
Copenhagen, Denmark, September 7, 2017.



(a) GlyphWiki (b) IDS

(c) KanjiVG (d) KRADFILE

Figure 1: A visualisation of a full decomposition of the character 賂 (*mainai* "bribe") according to the four kanji datasets.

such as their decompositions into bushu, and their usage as bushu in other kanji. Decomposition information in GlyphWiki is collected by user contribution.

2.2 IDS

IDS[2] is based on an open source project which uses character processing methods to create descriptions of character components (Morioka and Wittern, 2002). The decompositions in IDS are generated automatically, using character topic maps to draw associations between kanji and their constituent elements.

In Figure 1b we see that IDS provides a more detailed breakdown than GlyphWiki, including two bushu that are not found in GlyphWiki. At the first level of decomposition, though, they are identical.

2.3 KanjiVG

KanjiVG[3] is a collection of images that provides information about kanji and their decompositions. Decomposition information for KanjiVG is derived from the analysis of strokes used to write kanji characters (Apel and Quint, 2004). Although KanjiVG does allow for the decomposition of characters down to the stroke level, we exclude all strokes in this research as we do not consider strokes to reflect semantic meaning.

Note in Figure 1c that the decomposition is different to IDS, because KanjiVG specifies that the bottom elements of the bushu character 貝 (*kai* "shell") are strokes rather than bushu, meaning they are excluded from decomposition.

2.4 KRADFILE

KRADFILE[4] represents a flat decomposition of kanji into their constituent bushu. One key aspect of KRADFILE that differs from the other datasets is the use of a relatively limited set of bushu. Additionally, unlike the other datasets, KRADFILE does not list bushu in an order consistent with their appearance in the kanji. Furthermore, KRADFILE provides a single exhaustive decomposition for all kanji characters and their bushu. Because of this, we consider KRADFILE to have only a single, indivisible layer of decomposition.

2.5 Dataset Comparison

Table 1 shows descriptive statistics for the four datasets and their decompositions.

[2] https://github.com/cjkvi/cjkvi-ids
[3] http://kanjivg.tagaini.net/
[4] http://users.monash.edu/~jwb/kradinf.html

	GlyphWiki	IDS	KanjiVG	KRADFILE
Characters	18761	20970	6744	12156
Unique bushu	2834	3104	1327	254
Average bushu per kanji	1.9	2.1	2.2	4.5
Average branching factor	1.5	2.1	1.9	4.5
Average depth	2.7	3.1	2.9	1.0

Table 1: Statistics for the four kanji datasets

The *Characters* row of the table describes the total number of characters in each dataset that have a decomposition. KanjiVG contains a much smaller number of kanji than the other two datasets, although note that in modern Japanese, kanji is generally restricted to 2,136 *jouyou* kanji (Bond and Baldwin, 2016). All four datasets have full coverage over our corpus.

Unique bushu describes the number of characters that have been used as bushu. In general, if a kanji character is found in the decomposition of another kanji character, then it is counted as a bushu. While most of the datasets use thousands of bushu, KRADFILE is notable in that it uses a much smaller bushu set. With respect to *Average bushu per kanji*, there is strong similarity between GlyphWiki, IDS and KanjiVG, but KRADFILE produces almost double the number of bushu because the decompositions are exhaustive.

Average branching factor describes the average number of bushu found through exhaustively decomposing over every kanji and its bushu (an example of exhaustive decomposition — which we call "deep decomposition" — can be seen in Figure 2). Because KRADFILE provides a single layer of decomposition that is complete and indivisible, we cannot decompose each bushu any further. Therefore, KRADFILE has an average depth of 1.

3 Experimental Setup

For our experiments, we use version 1.5 of the NAIST text corpus (Iida et al., 2007), a collection of Japanese newspaper articles which is widely used in Japanese NLP research (Imamura et al., 2009; Sasano and Kurohashi, 2011). The corpus consists of roughly 1.7 million character tokens, of which roughly 42% are kanji. To build and test our models we use 5-fold cross-validation.

Our language models are standard neural network models, implemented in Tensorflow; they consist of a embedding layer, with embeddings for each character (including kanji, bushu, and other elements of the Japanese writing systems) which are learned during training, a standard unidirectional LSTM (Sundermeyer et al., 2012), and a layer which maps the output of the LSTM to a vector representing the probability of the next character; the hidden (embedding) size of the LSTM for our experiments is 128. We train the language model by minimizing the cross-entropy between the output probabilities and the one-hot vector corresponding to the correct answer, using the Adam optimizer with a batch size of 128 and a learning rate of 0.002.

In addition to the four kanji datasets, we consider two kinds of decomposition: shallow and deep. Shallow decomposition refers to using only the first layer of decomposition of a kanji character, whereas deep expansion refers to an exhaustive decomposition of the kanji and all of its bushu. We use these two methods to explore whether semantic information is reflected in deeper levels of decomposition. In general, we aim to compare the performance of a language model based on the way kanji are decomposed and the depth of their decompositions.

Figure 2 includes examples of both shallow and deep decompositions, for kanji including 賂 (*mainai* "bribe") from Figure 1b. Decomposition is done in a left-to-right in-order traversal.

It is possible for multiple kanji to share the same bushu. For example, according to KanjiVG, the characters 由 (*yoshi* "cause/reason"), 甲 (*kou* "carapace/shell"), and 申 (*saru* "monkey") are decomposed into 田 (*ta*

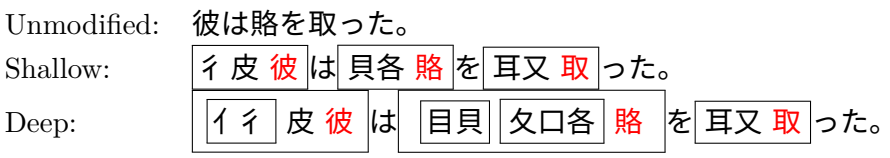


Figure 2: Examples of shallow and deep decomposition using KanjiVG, with original characters highlighted in red. Boxes denote characters that have been decomposed.

k	Baseline	GlyphWiki		IDS		KanjiVG		KRADFILE
		Shallow	Deep	Shallow	Deep	Shallow	Deep	
1	77.76	34.31	40.68	37.11	47.75	40.04	47.04	160.94
2	38.17	33.34	37.52	33.75	48.54	35.91	45.20	89.17
3	59.41	39.72	49.02	44.73	63.69	45.95	57.79	99.79
4	70.40	50.70	61.68	50.79	62.00	50.79	64.68	125.67
5	60.05	50.83	52.65	52.12	68.58	53.03	71.00	155.00
Average	61.16	41.78	48.31	43.70	58.11	44.74	57.14	126.11

Table 2: Language model perplexity based on the different decompositions

"rice field") and 丨 (*tatebou* "vertical line"). Because of this, using just the decomposition of a character can actually lead to a loss of information. Thus, we postpend all decomposition sequences with the original kanji character to preserve the mapping of bushu to its original kanji.

We evaluate based on perplexity, normalizing the product of the probability of the (sub-)characters in our test set by the character length of the corpus (lower perplexity is better). Because decomposing characters affects the superficial length of the corpus, however, we note that in the cases of decomposition we are normalizing using the original (undecomposed) corpus length in all cases, and not the decomposed token length. This reflects the fact that by adding decompositions of a character we are not really adding new text to the corpus. Both regular and decomposed language models in fact predict exactly the original contents of the corpus, but for the decomposed models the uncertainty associated with each kanji is distributed among the predictions of its bushu (and the postpended kanji), and can be retrieved simply by multiplying all the individual probabilities together.

4 Results

Table 2 shows the perplexity for each of the sub-character language models, for the two possible decomposition depths, as compared to a baseline where no decomposition occurs. We report perplexity for each fold of our 5-fold cross-validation, as well as the average.

First, we note that while most of the sub-character language models showed some improvement over the undecomposed baseline, KRADFILE performed substantially worse, with a mean perplexity score almost twice as high as that of the baseline. One potential problem with KRADFILE is that it provides only deep, exhaustive decompostions. Other limitations of using KRADFILE for language modelling are the lack of order in how bushu are arranged, and the fact that the bushu are limited to a specific set of characters. We can conclude that it is not a useful dataset for this purpose.

The best-performing dataset was GlyphWiki, with shallow decomposition. Not only was this configuration markedly better than the baseline on average, it also beat every other option on every fold of our cross-validation. The results for KanjiVG and IDS were similar, but slightly worse. Interestingly, based on the statistics in Table 1, GlyphWiki is the most conservative of the datasets in terms of the average number of decomposed bushu. We also found that the best results all involved shallow decomposition, which may reflect the fact that the most semantically-

salient bushu tend to appear at the first level of composition; this result was also consistent across folds for IDS and KanjiVG. Taken together, these results indicate that some decomposition is useful for building Japanese language models, but too much decomposition is not advisable.

5 Related Work

Working at the character level has proven useful in language modelling in English, as well as related applications such as building word representations (Graves, 2013; Ling et al., 2015). With regards to ideographic languages, there is work in information retrieval that has considered the appropriate representation for indexing; the focus has typically been word versus character (Kwok, 1997; Baldwin, 2009), but Fujii and Croft (1993) considered (though ultimately rejected) sub-character based indexing. In terms of investigations of the usefulness of sub-character representations for neural network models in ideographic languages, relevant work includes recent papers that use sub-character information to assist in the training of character embeddings for Chinese (Sun et al., 2014; Li et al., 2015; Yin et al., 2016) or build sub-character embeddings directly (Shi et al., 2015), demonstrating that sub-character information is useful for representing semantics in Chinese. However, our work differs not only in language and task, but also in our use of decomposition, since the work done in Chinese has primarily focused on a single semantically relevant sub-character (known as the radical), despite the fact that other sub-characters do provide additional semantic information in some characters.

6 Conclusion and Future Work

In this paper we have explored the idea of decomposing Japanese kanji to improve language modeling using neural network models. Our results indicate that it is possible to improve the predictive power of a language model using decomposition, as measured by perplexity, but the effectiveness of this does depend on the properties of the kanji database: whereas GlyphWiki is a useful resource for our purpose, KRADFILE is clearly not.

With respect to future work, we have thus far explored only a subset of the options with regards to the decomposition and ordering of sub-characters, and we would also like to consider more sophisticated models which integrate the structure of the kanji instead of flattening it, and applying our sub-character modeling to other sequential tasks such as part-of-speech tagging. Given the correspondence between kanji and Chinese characters, a comparison of the two languages with regards to the usefulness of decomposition would be worth exploring. We are also interested in performing an intrinsic evaluation of the character- and bushu-level embeddings learned through language modelling, e.g. relative to character-level similarity datasets such as that of Yencken and Baldwin (2006).

References

Ulrich Apel and Julien Quint. 2004. Building a graphetic dictionary for Japanese kanji: character look up based on brush strokes or stroke groups, and the display of kanji as path data. In *Proceedings of the Workshop on Enhancing and Using Electronic Dictionaries*. Geneva, Switzerland, pages 36–39.

Timothy Baldwin. 2009. The hare and the tortoise: speed and accuracy in translation retrieval. *Machine Translation* 23(4):195–240.

Francis Bond and Timothy Baldwin. 2016. Introduction to Japanese computational linguistics. In Francis Bond, Timothy Baldwin, Kentaro Inui, Shun Ishizaki, Hiroshi Nakagawa, and Akira Shimazu, editors, *Readings in Japanese Natural Language Processing*, CSLI Publications, Stanford, USA, pages 1–28.

Hideo Fujii and W. Bruce Croft. 1993. A comparison of indexing techniques for Japanese text retrieval. In *Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. Pittsburgh, USA, pages 237–246.

Alex Graves. 2013. Generating sequences with recurrent neural networks. *CoRR* abs/1308.0850.

Ryu Iida, Mamoru Komachi, Kentaro Inui, and Yuji Matsumoto. 2007. Annotating a Japanese text corpus with predicate-argument and coreference relations. In *Proceedings of the Linguistic Annotation Workshop*. Prague, Czech Republic, pages 132–139.

Kenji Imamura, Kuniko Saito, and Tomoko Izumi. 2009. Discriminative approach to predicate-argument structure analysis with zero-anaphora

resolution. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*. Singapore, pages 85–88.

Kui-Lam Kwok. 1997. Comparing representations in Chinese information retrieval. In *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. Philadelphia, USA, pages 34–41.

Yanran Li, Wenjie Li, Fei Sun, and Sujian Li. 2015. Component-enhanced Chinese character embeddings. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Lisbon, Portugal, pages 829–834.

Wang Ling, Chris Dyer, Alan W Black, Isabel Trancoso, Ramon Fermandez, Silvio Amir, Luis Marujo, and Tiago Luis. 2015. Finding function in form: Compositional character models for open vocabulary word representation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Lisbon, Portugal, pages 1520–1530.

Tomohiko Morioka and Christian Wittern. 2002. *Moji Dētabēsu ni Motoduku Moji Object Gijutsu no Kōchiku*. In *Proceedings of IPSJ 2002*. (in Japanese).

Ryohei Sasano and Sadao Kurohashi. 2011. A discriminative approach to Japanese zero anaphora resolution with large-scale lexicalized case frames. In *Proceedings of the 5th International Joint Conference on Natural Language Processing (IJCNLP 2011)*. Chiang Mai, Thailand, pages 758–766.

Xinlei Shi, Junjie Zhai, Xudong Yang, Zehua Xie, and Chao Liu. 2015. Radical embedding: Delving deeper to Chinese radicals. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*. Beijing, China, pages 594–598.

Yaming Sun, Lei Lin, Nan Yang, Zhenzhou Ji, and Xiaolong Wang. 2014. Radical-enhanced Chinese character embedding. In *Proceedings of the 21st International Conference on Neural Information Processing (ICONIP 2014)*. Kuching, Malaysia, pages 279–286.

Martin Sundermeyer, Ralf Schlüter, and Hermann Ney. 2012. LSTM neural networks for language modeling. In *Proceedings of the 13th Annual Conference of the International Speech Communication Association (Interspeech 2012)*. Portland, USA, pages 194–197.

Lars Yencken and Timothy Baldwin. 2006. Modelling the orthographic neighbourhood for Japanese kanji. In *Proceedings of the 21st International Conference on the Computer Processing of Oriental Languages (ICCPOL 2006)*. Singapore, pages 321–332.

Rongchao Yin, Quan Wang, Rui Li, Peng Li, and Bin Wang. 2016. Multi-granularity Chinese word embedding. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Austin, USA, pages 981–986.