Finding Structure in the Latent Space of Japanese Words for Improving Word Difficulty Labelling

Jordan Coil
Department of Computer Science
University of British Columbia
`jcoil93@students.cs.ubc.ca`

Ethan Thoma
Department of Computer Science
University of British Columbia
`ethoma@students.cs.ubc.ca`

Abstract

Categorization of language into difficulty levels is commonly done on full texts or sentences. Typically, these models utilize word difficulty as a factor. However, word difficulty labels are not exhaustive. Following research on word difficulty labelling, in this paper, we focus on modelling the latent space of Japanese words via multiple factors to generate meaningful subspaces for better accuracy in predicting word difficulty labels. Evaluation of our latent space representation using k-means clustering produced silhouette scores of 0.6785, 0.6305, and 0.2679 for dimension spaces of 2, 4, and 8 respectively. Qualitatively, there is no clear separability between clusters, and any plots that indicated high inter-cluster distance failed to separate word difficulty from previously labelled datasets.

1 Introduction

Difficulty categorization of a language's lexicon is an important, non-trivial task utilized in a wide array of fields. An intuitive example of how difficulty categorization can be useful is children's books: books aimed at children are typically easier to comprehend, in terms of lexical difficulty, than books for adults. A major area of focus for labelling is the categorization of materials for second language (L2) learners. L2 students typically start out with beginner materials, progress to intermediate, and then onto advanced materials. These materials, like textbooks for example, divide sentences, vocabulary, and grammar into relative difficulty categories. However, these categorizations cannot contain every word and grammar pattern in a language. This can be problematic for effective learning if the categorized corpus is small, as Nation [3] suggests that more than 90% of a text must be comprehensible in-order to make improvements in L2 acquisition. In this paper, we focus on the categorization of Japanese word difficulty for L2 learners.

A common approach to categorizing the lexical difficulty is focused on labelling entire sentences or texts. However, these methods rely on incomplete and/or estimated word difficulty labels and some grammatical information [4][2][6]. Instead, we focus on obtaining more accurate word difficulty labels which may contribute to improving sentence- or text-based categorization methods. One common factor in labeling word difficulty is frequency [2][6][1]. However, frequency is not the endgame for this task. Hashimoto and Egbert [1] tested 33 variables and their relation to word difficulty and found that 6 of the 33 variables, including frequency, were only able to account for 37.2% of word difficulty. In order to better predict word difficulty, more features are thus required. In our work, we utilize a variational auto-encoder (VAE) optimized via reconstruction error to find a latent representation of word features. We then use principal component analysis (PCA) and t-distributed stochastic neighbor embedding (t-SNE) to see if there are clusters related to difficulty in the latent representation. The factors we use are: frequency, kanji[1] grade level, word embeddings via word2vec, and word similarity for phonetic estimation via Japanese kana.[2]

The rest of the paper has a layout of the following: Section 2 is the related works section, Section 3 is on the collected data as well as our methodology for selecting relevant features, Section 4 discusses the results of the clustering model in terms of evaluation and performance, and Section 5 discusses the success of our approach and things to consider in future work.

2 Related Work

Most proposed methods for evaluating the difficulty of Japanese L2 acquisition is focused on sentence based difficulty classification. Almost all methods tend to utilize word and grammatical pattern difficulty as factor. A simple method, that was explored by Nishina and Yoshihashi [4] for sentence difficulty classification, labelled the difficulty of sentences according to the most difficult word or grammatical pattern present. In doing so, they could build a classified corpus of sentences for Japanese L2 learners. However, this approach has the drawback that it is dependent on the accuracy of the difficulty classification of both words and grammatical patterns. As such, this dependency motivates the problem of obtaining more accurate labelling. Furthermore, this simplified aggregation lacks nuance as sentences are partitioned by varying degrees of difficulty. For example, a relatively simple sentence with one difficult word which can be derived from context would be evaluated to be the same as a sentence with only difficult words and grammatical patterns.

Another, more-advanced method for sentence difficulty, was proposed by Mizuno et al. [2]. In their paper, they utilized a weighted score of varying factors: sentence length and difficulty of words, grammar, and kanji. For difficulty, words were assigned a difficulty level based on their corresponding level in the Japanese Language Proficiency Test (JLPT).[3] For words that did not appear in the JLPT, word difficulty was assigned by comparing the frequency of that word to the frequency of other JLPT words (frequency was measured from a collection of newspaper articles). For example, if a word, with an unknown difficulty level, had a frequency in the range of other words in the lowest level of the JLPT test, then that word would be labelled similarly.[4]

Further expanding on the factors used by Mizuno et. al, Ramkissoon [6] utilized a support vector machine (SVM) on unigram frequencies of JLPT levels of the words, grammatical patterns, and kanji to predict for full texts (inclusive of sentences). For unlisted words, words not in the JLPT, Ramkissoon used a regression model on the word's unigram frequency from the Balanced Corpus of Contemporary Written Japanese (BCCWJ), the mode of the kanji levels in the word, and the hardest kanji level in the word. This method produced 82.35% for adjacent-level accuracy.[5]

Given the success of Ramkissoon's model, a combination of unigram frequency and kanji level difficulty provides sufficient accuracy when aggregated over a whole text. However, many other factors contribute to the difficulty of a word as found by Hashimoto and Egbert [1]. In their paper, they explored the relationship of word difficulty acquisition by L2 learners to 33 other factors stratified by the L1 language of the participants that rated the difficulty of the word. In their exploration, they found that the factors of polysemy/homonymy (has many meanings or sounds), contextual distinctiveness (which contexts a word appears in), unigram frequency in a corpus, dispersion of use in two corpora, and neighborhood density (how similar words are to each other) accounted for only 37.2% of word difficulty. This study highlights the importance of including more factors for predicting labels of words as compared to sentences or texts as aggregating over larger bodies implicitly includes other factors referenced by Hashimoto and Egbert. As such, none of the described models above provide sufficient factors for estimating word difficulty alone.

[1]Japanese characters adapted from the Chinese writing system.

[2]Kana are the Japanese phonetic characters.

[3]Specifically, the 2008 JLPT public word-lists.

[4]Currently there are 5 levels of the JLPT, N5 (lowest) through N1 (highest). Prior to 2010, when the last public word-lists were available, there were only 4 levels.

[5]adjacent-level accuracy is defined as a correct classification where the predicted difficulty level is equal to or one level easier than the true level.

3 Methodology

This section introduces the datasets and methodologies employed. First, we collect the required datasets. Second, we generate a word2vec for capturing the word embeddings as word similarity can be used to approximate polysemy and neighborhood density. Third, we capture a bag-of-characters of the kana of the Japanese words. This helps to capture homonymy and neighborhood density. Finally, we generate a clustering model of the merged datasets from the previous steps.

3.1 Assumptions

There are multiple assumptions stemming both from our analysis of the sentence-based methods for categorizing difficulty as well as for our own approach to our datasets. We make the following assumptions:

- The sentence- and text-based methods implicitly model contextual distinctiveness, unigram frequency, and dispersion of use. Thus, in order for word-based methods to perform comparably, these features must be included.
- The selected features of frequency, kanji, grade level, word embeddings via word2vec, and word similarity for phonetic estimation via Japanese kana contain the information of polysemy/homonymy, contextual distinctiveness, unigram frequency in a corpus, dispersion of use in two corpora, and neighborhood density as described in the study by Hashimoto and Egbert [1].
- The JLPT standardized test levels have a meaningful relationship to word difficulty.
- The difficulty of a word is the same across cultural backgrounds or that the effect is negligible.
- Each word's difficulty is independent of every other word's difficulty given it's features.
- Difficulty category boundaries are relatively arbitrary. Thus, the categories will not have high separability and have dense boundaries.

These assumptions will allow us to build a model that can predict word difficulty.

3.2 Datasets

We take a similar approach as Ramkissoon [6] when it came to collecting factors for modelling word difficulty. However, as discussed above, more factors are required. As such, we use a difficulty categorized word dataset, a corpus for unigram frequency and dispersion of use, and a word-list for kanji difficulty. It was shown by Hashimoto and Egbert [1] that more corpora would increase predictive power. We believe that this is due to it being a better model of contextual distinctiveness and dispersion of use. The respective datasets are as follows:

- Japanese Wikipedia articles containing around 3.4GBs of word data.[6] This is a large corpus used for generating the word2vec model.
  Note: The usage of this corpus as well as the hyperparameterization of the word2vec model derives from `https://github.com/philipperemy/japanese-words-to-vectors`.
- Pre-2010 JLPT public word-lists, containing words labelled by JLPT test level.[7] This dataset contains fields for each word's kanji, kana, and JLPT level. The levels are 1 through 4. 1 corresponds to the highest, or most difficult, test level. 4 corresponds to the the lowest, or easiest, test level. This word-list contains approximately 8,000 words.
  Note: The lists used are from 2007. Since 2010, the JLPT has changed to 5 levels and no longer releases word-lists to the public.
- Kanji grade level word-list.[8] This list contains 2136 kanji of grades 2 to 10. This dataset contains fields for the kanji and grade level.

[6] `https://dumps.wikimedia.org/jawiki/latest/jawiki-latest-pages-articles.xml.bz2`
[7] `http://www.thbz.org/kanjimots/jlpt.php3`
[8] `https://www.kanji-link.com/en/kanji/grade/`

- Netflix Japanese subtitle frequency dataset, containing approximately 120,000 words.[9] This dataset contains fields for each word's frequency count, cumulative frequency percentage, and part of speech.

3.3 Word2vec

To capture the features of polysemy and neighborhood density, we use word2vec trained on Japanese Wikipedia articles. Word2vec produces word embeddings that allows for simple computation of word similarity which is used to learn word associations. Using these embeddings, we can capture relevant features for word difficulty in its latent space. First, we tokenize the Japanese Wikipedia articles via the lattice-based MeCab[10] tokenizer. This is done as Japanese lacks spaces and word2vec requires the words to be easily separated (spaced). Then, we generated the word2vec model with continuous bag-of-words (CBOW) on the tokenized data via the Gensim[11] library.

3.4 Phonetics

Unlike English (and other Roman alphabet languages), Japanese kana are representative of the phonetics of the word. As such, to capture the phonetic data, we use a bag-of-characters model on the kana representation of each word and then transform it into a vector. In bag-of-characters, we treat a word as a bag and track the count of each of the characters in the bag. We believe this model is sufficient for capturing homonymy as words with the same kana (but different kanji) would have zero difference in their vectors. The same applies for words in which the order of kana is different but maintain the same multiplicity. Furthermore, these vectors also capture the length of kana which is another feature of word difficulty [1]. Finally, neighborhood density is also measured in terms of phonetic similarity which is encapsulated by these vectors.

3.5 Feature Merging

After generating both the word2vec model and the phonetic vectors, we then tidy and transform the JLPT word-lists and the kanji levels. For the word-lists, we first parse them all into one data-frame. Next, for the kanji levels, we load and normalize the levels from $[1, 10]$ to $[0, 1]$ to generate a second data-frame. Then we load the Netflix Japanese subtitle frequency dataset and filter words with Roman characters. This is done as sometimes Netflix uses English or Roman alphabet acronyms in their subtitles - especially for commonly used load words. We merge the first two data-frames from above with this frequency dataset. We do so by matching the kanji of the Netflix dataset with the kanji in the JLPT word-list (or the kana if there is no kanji). For the kanji levels, we set the word to the max kanji level of the word - a character-to-word variant of the word-to-sentence approach done by Nishina and Yoshihashi [4]. If none of the kanji in the word have a matching kanji level, it is set to -1. Finally, we add the phonetic vectors to by matching the kana of the vectors to the kana of the Netflix words.

3.6 Final Clustering

For modeling the features, we use an autoencoder neural network model to transform the features into varying-sized latent spaces. An autoencoder works by shrinking the input space down to the size of the bottleneck layer - refereed to as the encoder stage. After the bottleneck layer, the network grows the space back to the same size - refereed to as the decoder stage. Thus, the encoder stage can be used for transforming our features into a latent space of our choice. Specifically, we use a simple model of 2 hidden layers in the encoder and 2 hidden layers in the decoder. We utilize AdamW optimizer which is Adam boosting with proper weight decay implementation and use the MSE reconstruction error as the loss function. We tried 3 different sizes for the bottleneck layer: 2, 4, and 8. This is done as the multiplicity of difficulty levels has no clear factor of separation as it is a subjective measurement.

The choice of clustering over classification via JLPT levels is due two reasons: one, there are approximately 8,000 labelled words out of the estimated 500,000 Japanese words and two, the 4 levels of JLPT are arbitrary and inconsistent. For the first point, we believe trying to model a highly

[9] `https://github.com/chriskempson/japanese-subtitles-word-frequency-list`

[10] `https://taku910.github.io/mecab/`

[11] `https://radimrehurek.com/gensim/`

accurate model would be infeasible as we could only check for accuracy on the training and testing of 1.6% of words. For the second point, the arbitrary nature of these subdivisions is apparent as it is a subjective evaluation by L2 learners and teachers. In terms of inconsistency, there is a tendency for JLPT labels to differ for the same word depending on the organization and people creating said labels. This is why Ramkissoon [6] presented both accuracy and adjacent-level accuracy for his models as there is an assumption that miss-labelled data would be more likely to appear in adjacent labels. For these points, we believe that it is better suited to instead cluster the latent space and draw comparisons between labelled words and their clusters.

For the displaying of the latent space, we utilize both principal component analysis (PCA) and t-distributed stochastic neighbourhood embedding (t-SNE). Note, for a bottleneck size of 2, dimensionality reduction is not required. PCA works by finding the eigenvectors of the features that explains the most variance of the dataset. Furthermore, PCA maps the global structures of a dataset and tends to lose local structures. On the other hand, t-SNE is much better for mapping local structures. The t-SNE model is an unsupervised model that optimizes for the Kullback-Leibler divergence between conditional probabilities. Both methods work for dimensionality reduction but there are some things to note: one, PCA has a tendency to provide better visualizations for word2vec but is less suited for other models than t-SNE and two, t-SNE is a better suited for the crowding problem.[12] The crowding problem has importance in our exploration. Since we believe that these boundaries of difficulty are subjective, there is the implication that cluster boundaries will be unclear and populated. As such, the crowding problem will be more severe.

4 Analysis

After training the auto encoder, we plotted the latent representation of the labelled samples to qualitatively access if there are structures or clusterings present. We plotted the 2d latent representations directly as a scatter-plot, but in order to plot higher dimensional latent representations, we plotted the results of PCA and t-SNE. We utilized both methods as we have no assumptions of the scale of the structures in the data. When looking at the scatter-plots from Figure 1, Figure 2, and Figure 3, there does appear to be some structure to the data. Namely, the lowest difficulty samples seem to concentrate around a specific point. However, there is small inter-cluster difference between JLPT levels as there are many samples with differing difficulty levels also in this cluster. In the plot for the 2d latent space, there does appear to be small separate clusters, but when taking into account the JLPT labels in those clusters, it is clear these clusters are not representative of JLPT levels.

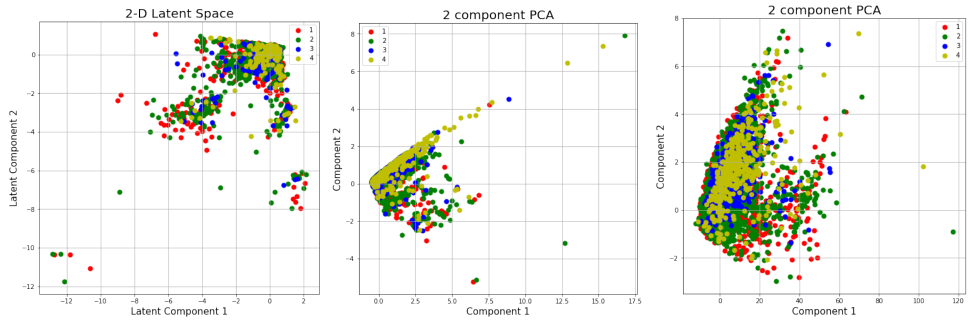

Figure 1: Latent space representations of words with labelled difficulty. From left to right, plot of the 2d latent space, plot of PCA components from a 4d latent space, and plot of PCA components from a 8d latent space.

[12]The crowding problem is when points in higher dimensions overlap in lower dimensions.

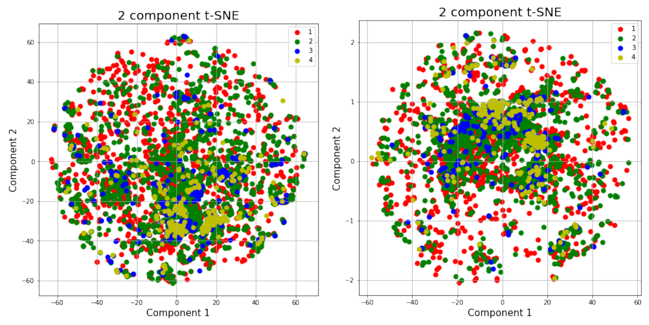

Figure 2: Latent space representations of words with labelled difficulty. Left: plot of the t-SNE components from a 4d latent space, and plot of the t-SNE components from a 8d latent space.

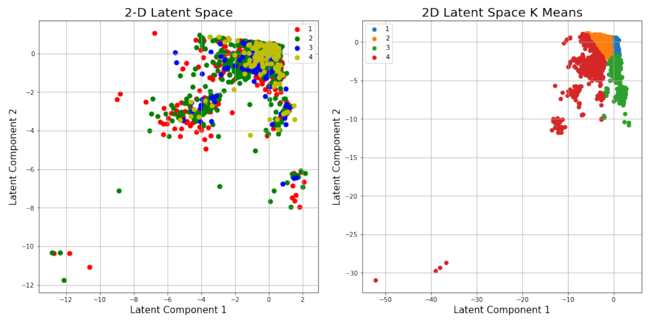

Figure 3: Left: Scatter-plot of 2d latent representations of words with labelled difficulty. Right: Scatter-plot k-means on 2d latent representations of all samples. Comparing these plots makes it apparent that the JLPT difficulty labels don't map onto the clusters in latent space

In order to see if there was some meaningful clustering present in the latent representations, we ran k-means on each of the 2d, 4d, and 8d representations - setting the number of clusters set to 4 (akin to JLPT label multiplicity). We then computed a silhouette score on the results to measure the validity of the clusters. From Table 1, we can see that the 2d and 4d k-means clusters have the highest silhouette score. This means that these clusters have lower intra-cluster distance and higher inter-cluster distance. As such, it indicates some structure to the data. However it is hard to say whether or not this structure is representational of word difficulty. When we compare the plots of the labelled examples in 2d latent space to the k-means plot of all examples in 2d latent space, it seems that the clusters do not match the dispersion of the labelled examples. Thus, despite 2d k-means clusters having the highest silhouette score, it is likely the structure in the latent space does not represent word difficulty.

Table 1: K-means Silhouette Scores

Latent Dimensions	Silhouette Score
2	0.6785
4	0.6305
8	0.2679

5 Discussion and future work

Conclusion In our paper, it is qualitatively apparent that the latent space does not represent difficulty - at least relative to JLPT levels. What does this imply? Perhaps some of the assumptions that we made are too strong. For instance, the process of selecting words for the JLPT tests is an unknown factor. As such, it is possible that features we captured, such as the frequency, are not good predictors of

JLPT difficulty. Furthermore, JLPT in itself may be a misleading factor for word difficulty. Although commonly used by Japanese companies to access Japanese L2 learners ability, it is possible that these labels are biased. There may be meaningful subspaces of the latent space that correlate to word difficulty, however, given the findings by Hashimoto and Egbert [1], our expectations of these subspaces is low.

Assumption of separability From the plotted data, it is interesting to note that there was no separability in the latent space. This exploration gives some evidence that on the assumption mentioned before of relatively arbitrary boundaries. Furthermore, this helps motivate the claims of Hashimoto and Egbert [1] that the factors we included should only account for about a third of difficulty. We also researched different optimizations in order to encourage separability in the latent space such as those described in the article by Platform.ai [5]. However, all these approaches failed to produce better results and only increased training time - furthering the belief that there exists no inherent separability.

Feature drawbacks For our selected features, there are a few drawbacks. First, an important factor in word difficulty is dispersion of use in a corpus [1]. Including more corpora would help factors like unigram frequency, dispersion of use, and contextual distinctiveness approach their population parameters. However, we opted to utilize just the Netflix Japanese subtitle frequency dataset. This has inherent bias towards spoken forms of language as well as words and phrases often used in video-graphic media. Since our other features come from datasets based on text-based media, this may impede our models overall ability to find meaningful structure. Second, using word2vec has its own inherent problems. Word2vec models similarity between words and as no incentive for separability either. This is why it is uncommon to perform clustering directly to word2vec vectors as density-based cluster models tend to fail (the more common approach being k-means on aggregated vectors). One proposed method of dealing with this problem is to take the sum of cosine similarity between an unlabelled word and every JLPT labelled word, weighted by JLPT label.

Future work For future work, we aim to include many more corpora to better approximate the population distributions of unigram frequency, dispersion of use, and contextual distinctiveness. Furthermore, we would also study different methods of aggregation for approaching the word2vec and phonetic vectors. This would allow for more accurate representations of polysemy/homonymy and neighborhood density. Finally, for more robust JLPT labelling, we would employ the method used by Ramkissoon [6] of labelling a word by the mode of multiple JLPT sources.[13]

References

[1] B. J. Hashimoto and J. Egbert. More than frequency? exploring predictors of word difficulty for second language learners. *Language Learning*, 69(4):839–872, 2019. doi: https://doi.org/10.1111/lang.12353. URL `https://onlinelibrary.wiley.com/doi/abs/10.1111/lang.12353`.

[2] J. Mizuno, H. Oyama, T. Kobayashi, K. Sakata, N. Evans, Y. Taniguchi, and Y. Matsumoto. Human-computer interaction system using web news. Master's thesis, Nara Institute of Science and Technology, 2008.

[3] I. S. P. Nation. *Learning Vocabulary in Another Language*. Cambridge Applied Linguistics. Cambridge University Press, 2 edition, 2013. doi: 10.1017/CBO9781139858656.

[4] K. Nishina and K. Yoshihashi. Japanese composition support system displaying co-occurrences and example sentences. In *Proc. of the International Symp. on Large-Scale Knowledge Resources*, pages 119–122, Mar. 2007.

[5] Platform.ai. The silhouette loss function: Metric learning with a cluster validity index, 2020. URL `https://www.platform.ai/post/the-silhouette-loss-function-metric-learning-with-a-cluster-validity-index`.

[6] N. Ramkissoon. Japanese reading difficulty classification for non-native language learners. unpublished, 2020. URL `http://dev-blog-materials.s3.amazonaws.com/j_text_classification_research_paper.pdf`.

[13] Specifically, the paper used 3 sources and if at least 2 sources shared a label, set it to it.