Predicting Word Pronunciation in Japanese

Jun Hatori[1,*] and Hisami Suzuki[2]

[1] Department of Computer Science, University of Tokyo
7-3-1 Hongo, Bunkyo, Tokyo, 113-0033 Japan
hatori@is.s.u-tokyo.ac.jp
[2] Microsoft Research
One Microsoft Way, Redmond, WA 98052, USA
hisamis@microsoft.com

Abstract. This paper addresses the problem of predicting the pronunciation of Japanese words, especially those that are newly created and therefore not in the dictionary. This is an important task for many applications including text-to-speech and text input method, and is also challenging, because Japanese kanji (ideographic) characters typically have multiple possible pronunciations. We approach this problem by considering it as a simplified machine translation/transliteration task, and propose a solution that takes advantage of the recent technologies developed for machine translation and transliteration research. More specifically, we divide the problem into two subtasks: (1) Discovering the pronunciation of new words or those words that are difficult to pronounce by mining unannotated text, much like the creation of a bilingual dictionary using the web; (2) Building a decoder for the task of pronunciation prediction, for which we apply the state-of-the-art discriminative substring-based approach. Our experimental results show that our classifier for validating the word-pronunciation pairs harvested from unannotated text achieves over 98% precision and recall. On the pronunciation prediction task of unseen words, our decoder achieves over 70% accuracy, which significantly improves over the previously proposed models.

Keywords: Japanese language, pronunciation prediction, substring-based transliteration, letter-to-phone.

1 Introduction

This paper explores the problem of assigning pronunciation to words, especially when they are new and therefore not in the dictionary. The task is naturally important for the text-to-speech application [27], and has been researched in that context as letter-to-phoneme conversion, which converts an orthographic character sequence into phonemes. In addition to speech applications, the task is also crucial for those languages that require pronunciation-to-character conversion to input text, such as Chinese and Japanese, where users generally type in the pronunciations of words,

[*] This work was conducted during the first author's internship at Microsoft Research.

A. Gelbukh (Ed.): CICLing 2011, Part II, LNCS 6609, pp. 477–492, 2011.

which are then converted into the desired character string via the software application called pinyin-to-character or kana-kanji conversion (e.g. [8] [9]).

Predicting the pronunciation of words is particularly challenging for Japanese. Japanese orthography employs four sets of characters: *hiragana* and *katakana*, which are syllabary systems thus phonemic; *kanji*, which is ideographic and represents morphemes, and Roman alphabet. Kanji characters typically have multiple possible pronunciations, making the prediction of their pronunciation difficult. In many cases, you need to know the word to know its pronunciation: after all, the pronunciation is an idiosyncratic property of the word. Therefore, one goal of this paper is to propose an effective method for exploring textual resources to learn the pronunciation of words. At the same time, we are also motivated to find out how predictable the pronunciations of kanji words are. Native speakers of the language can take an educated guess at predicting a pronunciation of an unseen word; can a machine replicate such sophisticated performance?

Our approach to the problem of pronunciation prediction therefore consists of two parts: we first try to model the intuition that a fluent speaker has on how to pronounce words by a statistical model via the task of *pronunciation modeling*; we then use the model to harvest word-pronunciation pairs from the web in the task of *pronunciation acquisition*. In this paper, the pronunciation modeling task is considered as a simplified machine translation (MT) task, i.e., a substring-based monotone translation, inspired by recent work on string transduction research. Our model, trained discriminatively using the features that proved useful in related tasks, outperforms a strong baseline as well as an average human performance, while making the types of errors that are considered acceptable by human. For the pronunciation acquisition task, we use a classifier to validate word-pronunciation pairs extracted automatically from text, exploiting the convention of Japanese text that the pronunciation is often inserted in parentheses immediately following the word with a difficult or unusual pronunciation. Our classifier achieves over 98% precision and recall when Wikipedia was used as the source corpus.

There are several contributions of this paper. We believe that this is the first work to address the problem of word pronunciation prediction for Japanese in a comprehensive manner. We apply the state-of-the-art technology developed for related problems to solve this problem, with modifications that are motivated by the specific problem at hand. The use of unannotated corpus for the extraction of pronunciation in Japanese is also novel and proved effective.

The rest of the paper is organized as follows. Section 2 gives some background, including the task description and related work. Section 3 introduces our approach to the pronunciation modeling task, along with experimental results. Section 4 deals with the task of pronunciation acquisition from corpora, which takes advantage of the prediction model described in Section 3. We conclude with comments on future work in Section 5.

2 Background

2.1 Pronunciation Prediction: Task Description

We define the task of pronunciation prediction as converting a string of orthographic characters representing a word (or a phrase corresponding to an entity) into a sequence of hiragana, which straightforwardly maps to pronunciation.[1] The problem is trivial if the word is spelled entirely in non-kanji characters, so we only target the cases where at least one character in the word is spelled in kanji. Let us take an example of the name of the recently appointed prime minister of Japan, Naoto Kan (菅直人). Our goal is to convert this string intoかんなおと, which is pronounced as [ka-N-na-o-to].[2] How ambiguous is this name to pronounce? According to the kanji pronunciation dictionary we have, the first character has three pronunciations, the second fourteen and the third twelve:[3] therefore, there are $3 \times 14 \times 12 = 504$ possible ways to pronounce this word. Naturally, some pronunciations are more common than others, especially given some contextual information. For example, 直人 is a common first name, pronounced as [nao-to] or [nao-hito] or maybe [tada-hito]; other pronunciations are highly unusual. Given that直人 is probably a first name, 菅 may be a last name, pronounced as [kan] or [suga], though it is fairly uncommon as a last name. Kanji characters typically have two types of pronunciations called *on-yomi* (literally 'sound pronuncation') and *kun-yomi* (literally 'meaning pronunciation'), corresponding to their origin (Chinese and Japanese, respectively), and they tend not to mix within a word, exemplified in 運転手 ([uN-teN-shu] 'driver', all on-yomi) vs. 手紙 ([te-gami] 'letter', all kun-yomi). Using these types of knowledge, one might guess that the name is reasonably pronounced as [kaN-nao-hito], [kaN-nao-to], [suga-tada-hito] and so forth. Eventually, the correct pronunciation can only be obtained by knowing the word, i.e., by identifying this string as a dictionary entry. The problems we try to solve in this paper is therefore twofold: one is to increase the dictionary coverage by learning word-pronunciation pairs automatically from text through *pronunciation acquisition*; secondly, for those words for which a dictionary entry is still missing, we would like to build a model to predict pronunciation that is not only highly accurate, but also makes reasonable mistakes when it fails – using the直人 example above, we hope to generate one of the three reasonable pronunciations. We focus on the task of predicting *word* pronunciation in this paper – selecting the right pronunciation for the words in a sentence is a related but independent task of *pronunciation disambiguation*, for which the pronunciation prediction task discussed in this paper will serve as an essential component.

2.2 Related Work

The task of pronunciation prediction is inspired by previous research on string transduction. The most directly relevant one is the work on letter-to-phoneme

[1] To be precise, additional operations are required to adjust the hiragana string for speech or text input applications, but we do not deal with this problem here.

[2] A hyphen is used to indicate a character boundary of the preceding string; [N] is used to indicate the pronunciation of the moraic nasal ん.

[3] This kanji pronunciation dictionary was available to us prior to the current research. It lists the pronunciations for about 6,000 kanji characters, with 2.5 pronunciations on average per character. The possible pronunciations for the three letters here are: 菅（すが,すげ,かん）, 直（ちょっ,すなお,す,ただし,ちょく,ね,ひた,ただ,のう,じき,なお,すぐ,じか,なおし）, 人（ひと,じん,と,り,たり,ど,にん,びと,うど,ぴと,うと,とな）.

conversion, where many approaches have been proposed for a variety of languages. The methods include joint n-gram models (e.g. [1] [2] [4]), discriminatively trained substring-based models (e.g. [11] [12]) which are themselves influenced by the phrasal statistical MT (SMT) models [15], and minimum description length-based methods [24]. The joint n-gram estimation method has also been applied to predicting pronunciation in Japanese (e.g., [21] [22]).

Similar techniques to the letter-to-phoneme task have also been applied to transliteration, which converts the words in one language into another that uses a different script, maintaining phonetic similarity. Early works on this task used the source-channel model based on one-to-one (or more) character alignment (e.g. [14]). Later they were extended to use many-to-many alignments using substring operations in the style of phrasal SMT (e.g. [28]), demonstrating improved accuracy over the character-based models. The components of the model proposed by [28] are themselves generative models, which can also be used in a SMT-style discriminative framework, where the weights on the component generative models are discriminatively trained. [5] proposed such a hybrid model, further improving the accuracy of transliteration. Joint n-gram models have also been applied to the task of transliteration (e.g. [17]).

In contrast to the wealth of literature in string transduction research, the task of pronunciation acquisition has attracted much less attention in the past. [10] describes a method in which they learn English pronunciations from the web using IPA (e.g., ‘beet /bit/’) and ad-hoc (e.g., ‘bruschetta (pronounced broo-SKET-uh)’) transcriptions by first extracting candidate pairs using a letter-to-phoneme model, which are then validated using SVM classifiers. Our approach is similar to theirs, with modifications in the method of generating candidates, to be explained in Section 4. [29] proposed a method to use the web for assigning word pronunciation in Japanese, but their focus is on disambiguating known word pronunciations rather than learning new word-pronunciation pairs. [16] and [26] discuss the methods of disambiguating new word pronunciation using speech data.

3 Substring-Based Pronunciation Prediction

This section describes our substring-based approach to pronunciation modeling. As mentioned above, the pronunciation of a kanji is dependent on those of the surrounding characters, which motivates a substring-based alignment and decoding over a character-based approach. We also assume that the task is basically monotone and without insertion/deletion, with kanji–hiragana alignments of 1–n (source–target, n ≥ 1) characters.[4] We adopt a discriminative learning framework that uses

[4] This is a slight oversimplification – we are aware of the cases where these assumptions do not hold. The monotonicity assumption breaks in the pronunciation of a kanji sequence that reflects the Chinese SVO word order, as in 不弓引 [yumi-hika-zu] (a place name, which originally means ‘not draw a bow’, in which the correct alignment is assumed to be 不-zu (not), 弓-yumi (bow) and 引-hika (draw). Also, hiragana insertions occurs quite commonly as in 一関 [ichi-no-seki] (a place name, meaning ‘first checkpoint’), where ‘no’ is a genitive marker inserted between the kanji characters.

component generative models as real-valued features, which is the standard method for statistical MT [23], and is reported to work comparably or better on a transliteration task than a discriminative model that uses sparse indicator features [5].

3.1 Model and Features

We adopt a linear model of pronunciation prediction: given the target character (hiragana) sequence t and the source (kanji) sequence s, we define features over s and t, $f_i(s,t)$ for $i = 1, \cdots, n$ The features are arbitrary functions that map $< s,t >$ to real values, and the model parameters are a vector of n feature weights, $\boldsymbol{\lambda} = (\lambda_1, \cdots, \lambda_n)$. The score of t with respect to s is given by

$$\mathrm{Score}(s,t,\boldsymbol{\lambda}) = \boldsymbol{\lambda} \cdot \mathbf{f}(s,t) = \sum_{i=1}^{n} \lambda_i f_i(s,t).$$

For the features, we use those that are motivated by MT and transliteration research: the translation probabilities in both directions, $\mathrm{P}(t|s)$ and $\mathrm{P}(s|t)$, the target character language model probability $\mathrm{P}(t)$, the operation count, which corresponds to the number of phrases in phrasal SMT, and the ratio of the source and target character length. Crucially, the estimation of the first three of these probabilities requires a set of training corpus with source and target alignment at the substring level. We take an unsupervised approach in generating such training data: we used an automatic word aligner developed for MT for obtaining these alignments, as detailed in Section 3.3 below.

3.2 Training and Decoding

For the training of the parameters of the linear model, we used averaged perceptron training. Let d stand for a *derivation* that describes a substring operation sequence converting s into t. Given a training corpus of such derivations $D = \{d_i, \cdots, d_n\}$ obtained from the substring-aligned text, the perceptron iterates the following two steps for each training sample $d_i \in D$:

$$\text{Decode: } d^* = \underset{d \in D(src(d_i))}{\operatorname{argmax}} \ \boldsymbol{\lambda} \cdot \mathbf{f}(d)$$

$$\text{Update: } \boldsymbol{\lambda} \leftarrow \boldsymbol{\lambda} + \mathbf{f}(d_i) - \boldsymbol{f}(d^*)\,,$$

where $D(src(d))$ are all possible derivations with the same source side as d. For decoding, we used a monotone phrasal decoder similar to the one employed in phrasal SMT [31], a stack decoder with the beam size of 20, which was set using a development data.

3.3 Experiments

Data and settings. As mentioned above, we need a parallel data of kanji words with their pronunciation in our approach. An obvious source of such data is a dictionary: we used UniDic [6], a resource available for research purposes, which is updated on the regular basis and includes 625K word forms as of the version 1.3.12 release

(July 2009). Since we focus on the prediction of new words which are mostly nouns, we used the noun (including proper noun) portion of the dictionary, containing 195K words in total.

Though UniDic is a lexical resource that is constantly refreshed, we also investigated into a dictionary-free approach, where we exploit a large body of unannotated text to collect words' pronunciation. Specifically, our approach takes advantage of the convention of Japanese text that the pronunciation of those words that are difficult or unusual to pronounce [5] are often indicated in parentheses immediately following the word in question, as shown in Figure 1.

新潟県（にいがたけん）は、本州日本海側に位置する
旧国名から**越佐**（えっさ）と表現することもある。
ふたご座流星群（ふたござりゅうせいぐん、学名 Geminids）は…
名取市立**館腰**（たてこし）小学校
一力亭（うどん）

Fig. 1. Examples of parenthetical pronunciation expression from Wikipedia. Strings in boldface indicate the words corresponding to the pronunciation in parentheses; the regular expression (described below) extracts the underlined substrings.

We used a simple regular expression-based pattern matching to extract word-pronunciation candidate pairs from Japanese Wikipedia. It extracts a substring of hiragana characters in a pair of parentheses, preceded by any character string bounded by a punctuation character or a beginning of a sentence. Additional heuristics consist of the constraints based on kana characters (i.e. no kana character is allowed in the word string unless it also appears in the pronunciation string.) and length ratio (e.g. the pronunciation string cannot be shorter than the word string.[6]). Note that the extraction method runs the risk of extracting too much pre-parenthetical material: as seen in the second to last example of Figure 1, たてこし indicates the pronunciation of only the last two characters (館腰). Another more substantial source of noise comes from the cases where the hiragana characters in parentheses do not indicate the pronunciation at all, as in the last example of Figure 1: 一力亭 [ichi-riki-tei] is a name of a restaurant, followed by the kind of food they serve (うどん [u-do-N] 'noodle') which happens to be written in hiragana. Though the extracted word-pronunciation data is therefore quite noisy, we will demonstrate that the use of this data greatly enhances the accuracy of the prediction. Note that in spite of the use of simple heuristics, the annotator found that more than 90% of extracted instances are valid word-pronunciation pairs (as mentioned in the last paragraph of Section 3.4), while the heuristics were weak enough to cover most pronunciation candidates in Wikipedia.

[5] This contrasts with the dictionary, where the pronunciations of all words are found. As is explained below, we used Wikipedia, which is a cross between a dictionary and free text: pronunciations are always given in parentheses for each title word, in addition to the words that occur in the free text portion of the articles.

[6] This is because a kanji character normally corresponds to one or more hiragana characters. While we are aware of some exceptional cases in non-compositional pronunciation, as in 啄木鳥 [kera] 'woodpecker', they are negligibly rare (<10 cases in 195K nouns in UniDic.).

The parallel data extracted from Wikipedia in this manner as well as from the UniDic entries is then aligned at the substring level. Our method for this follows [5]: we use a phrase-based word aligner originally developed for MT, similar to the word aligner described in [32], by considering each character as a word. We also used hard substring length limits for the same purpose: 1 for the input and 4 for the output strings, reflecting the fact that word pronunciation is typically composed of the pronunciation of individual kanji characters.[7] The aligner generates only monotonic alignments, and does not allow alignments to a null symbol in either source or target side. The same restriction is applied during decoding as well.

We extracted a total of 463,507 word-pronunciation pairs from Japanese Wikipedia articles as of January 24, 2010. After removing duplicates, we reserved 5,000 pairs for development and testing (of which we used 200 for development and 2,000 for final evaluation), and used the rest for training, i.e., for generating training derivations upon which the features of the linear model were computed. The translation probabilities, $\mathrm{P}(s|t)$ and $\mathrm{P}(t|s)$, are estimated by maximum likelihood on the operations observed in the training corpus with one important modification: recall that these operations, estimated using the character aligner in an unsupervised manner, are minimal non-decomposable operations, and therefore does not capture any contextual information. In order to remedy this, we re-align the training data by using composed operations which are constructed from operation sequences attested in the training data to maximize $\mathrm{P}(s|t)$ and $\mathrm{P}(t|s)$, respectively, thereby removing the substring length limit employed in the character alignment phase.[8] Figure 2 shows an example of an alignment before and after the composition. This process offers an additional benefit of noise reduction of the training data, as we removed the operations that occurred less than C times (C is set using the development data, C=2 in our case), removing the training examples that are not reachable from the remaining operations. This reduced our data size for perceptron training to 427,644 pairs, a reduction of 6.7%. More detail on the relation between the data size and the accuracy of the prediction task is discussed in the next subsection. For the target character language

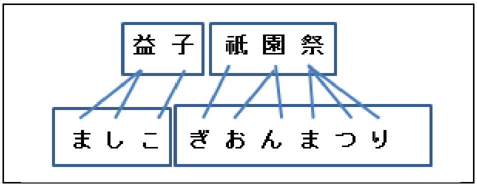

Fig. 2. Alignment before (=character level indicated by the lines) and after (=substring level by the boxes) composition for 益子祇園祭 ([mashi-ko-gi-on-matsuri], 'Mashiko Gion Festival')

[7] There are exceptions to this: occasionally, a pronunciation is assigned to a kanji string in a non-compositional manner (e.g., 今日 [kyou] 'today') . This is handled by the use of composed operations, to be explained below.

[8] We do however impose a limit on the length of composition (3 in our case). This two-path alignment approach also follows [5]. Since the phrase length limit for original operations are 1 for the source and 4 for the target, resulting composed operations can capture up to 3–12 (source–target) character alignments.

model, we used a 4-gram language model with Kneser-Ney smoothing and the BOS (beginning-of-string) and EOS (end-of-string) symbols, and trained it with the same training data as described above.

Baseline. We describe two baseline models that we used for comparison in the experiment. The first is KyTea, a publicly-available Japanese word segmentation and pronunciation prediction tool,[9] which achieves the state-of-the-art performance on the task of Japanese pronunciation prediction. According to [20] and the KyTea manual, the program first performs word segmentation, after which the pronunciation of each word is independently selected using a linear SVM classifier, choosing among the pronunciations that have appeared in the training data. When they encounter an unknown word, the output is the combination of the most frequent pronunciations of each kanji character. We ran KyTea (Version 0.11) with the default settings and with "the high-performance SVM model" available from the website, which is mainly trained on the Balanced Corpus of Contemporary Written Japanese (BCCWJ; [19]) and UniDic.

Our second baseline, the joint n-gram model, was proposed by [1], which has also been used for Japanese [21]. In this model, n-gram statistics are learned over the sequences of *pairs* of letters and phonemes, instead of the sequences of phonemes. While [21] used KyTea to extract word-pronunciation pairs from the annotated BCCWJ and newswire corpus to learn bigram statistics, we learned our n-gram statistics from the alignments obtained from the Wikipedia training set as described above. Note that even though we describe this approach as a baseline, it crucially relies on the paired substrings extracted in an unsupervised manner using the proposed approach. In that sense, the effectiveness of this baseline also incorporates a novel contribution of this work. We implemented the joint trigram model with Kneser-Ney smoothing, after adding the BOS (beginning-of-string) and EOS (end-of-string) symbols. The decoder performs the exact inference with Viterbi search over the probability space.

3.4 Results and Discussion

Table 1 shows the comparison of the proposed method against the baseline models in terms of the whole word accuracy on the pronunciation prediction task, evaluated on the 2,000 Wikipedia test pairs. All models other than KyTea were trained on the combination of Wikipedia-derived and UniDic pairs. As is observed from the table,

Table 1. Word-level accuracy (in %) of pronunciation prediction on Wikipedia test data

Model	Accuracy
KyTea	57.8
Joint Bigram	66.4
Joint Trigram	70.0
Proposed	71.7

[9] http://www.phontron.com/kytea/

the proposed method outperforms all the baseline models, with a 1.7% improvement over the joint trigram model, which is statistically significant with the significance level of $p < 0.01$ by the McNemar's test. Though falling short of the best model, the joint n-gram models are quite competitive, suggesting that the proposed method of unsupervised training data generation using word alignment techniques is beneficial for the task. The advantages of using an MT-inspired framework are therefore twofold: word alignment techniques for training data generation, and the linear combination of relevant feature functions for the best accuracy on the prediction task. KyTea performs poorly on this data set, suggesting that using a unigram model trained on manually created resources does not work well on the words that appear in Wikipedia.

It is noteworthy that the joint trigram and proposed models both outperformed the average human performance[10] (~65%) on the same data set. Since the current experimental setting allows only one pronunciation to be correct disregarding the context in which the word is used, it is possible that many errors are actually not errors but are acceptable in other contexts. An error analysis on the output of the proposed model confirms this speculation: about half of the errors were judged acceptable or correct upon human verification. Considering this fact, the performance of the proposed model is considered to be approaching the upper bound of this task; hence, the improvement of 1.7% is quite meaningful.

Table 2. Accuracy (in %) of pronunciation prediction models evaluated on the Wikipedia test set with respect to various training data sets

	Joint Trigram	Proposed
UniDic	46.9	47.5
Wikipedia	68.5	70.8
Wikipedia+UniDic	70.0	71.7

Table 2 shows the accuracy of the joint trigram and proposed models as a function of different training data source. The models trained with UniDic performed poorly, with the accuracy lower than 50%. This suggests that the alignments learned solely from a static dictionary resource are insufficient to predict the pronunciation of new words in Wikipedia. The use of Wikipedia-derived instances as training data improved the accuracy dramatically, achieving more than 20% improvement over the models trained on UniDic. Combining the two resources further improved the accuracy. The proposed model consistently outperforms the joint trigram baseline in all training data settings.

We also examined how incorrect alignments in the training data affect the pronunciation prediction performance. We did this by setting a cutoff threshold of the alignment scores, and removing those alignments with the scores lower than the

[10] The human performance is measured as follows: the source strings are presented to two native speakers with no context, and they are asked to assign guessed pronunciation. They both had graduate-level education.

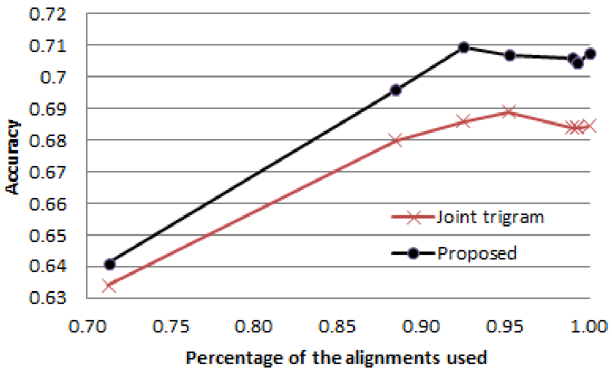

Fig. 3. Accuracy of pronunciation prediction models on the test set w.r.t. training data noise filtering

threshold from the training set. The alignment scores are output by the character aligner discussed above as the log probability of the alignment, and are normalized by the length of the source (i.e. kanji) sequence to avoid the preference for shorter sequences. Figure 3 shows the accuracy of the models with respect to the cutoff threshold. The x-axis corresponds to the percentage of the instances used in model training, while the y-axis indicates the word-level accuracy. The nodes in the graph correspond to the score threshold of −1, −2, −3, −4, −6, −8, −10, and −Infinity. The best performance is achieved when 90–95% of the training data is used, which is consistent with the observation that out of 100 words for which we manually verified the alignments, 11 instances contained alignments that appeared improper, and 7 instances were not word-pronunciation pairs though the aligner forced an alignment, which means that these instances are noise. Comparing the proposed vs. joint trigram-based methods, the former appears slightly more robust to noise: though it is rather difficult to see in the graph, the trigram model gained the maximum of 0.45% improvement by the noise filtering, while the proposed model gained the maximum of only 0.2% improvement. This may be attributed to the fact that the proposed model incorporates a noise filtering mechanism by way of minimum operation counts threshold C, as mentioned above.

4 Word Pronunciation Acquisition Task

This section describes our approach to harvesting word-pronunciation pairs from a large corpus with minimal supervision. We formulate this problem as a binary classification task: each candidate pair is determined to be a word-pronunciation pair or not, using a discriminatively trained classifier. Using the word-pronunciation candidate pairs in Figure 1 as an example, our goal is to classify the first four examples as positive and the last example as negative. A special treatment is required for the second to last example: recall that in this example, 名取市立館腰（たてこし），

only the last two characters (館腰) of the extracted word string corresponds to the pronunciation in hiragana in parentheses. For the modeling task in Section 3, we ignored this problem and treated these cases simply as noise. For the pronunciation acquisition task, we generate additional candidate pairs from these cases, so that from the string above, we generate the following pairs:

〈腰, たてこし〉，〈館腰, たてこし〉，〈立館腰, たてこし〉，〈市立館腰, たてこし〉

These expanded candidate pairs share the word strings to the right, their length bounded maximally by the number of hiragana characters in the pronunciation (assuming one-to-one or -many mapping between kanji and hiragana), and minimally by the length of hiragana pronunciation string divided by 3 (assuming that *on average*, a kanji character maps to up to 3 hiragana characters at the word level). Each of these candidate pairs are then submitted to the classifier to be validated as a desired word-pronunciation pair or not.

The task of pronunciation acquisition formulated in this manner, along with the sub-problem of the boundary detection, can be viewed as a very similar task to bilingual dictionary creation for Chinese MT (e.g. [3] [18]). The goal of this line of work is to exploit parenthetical expressions in the web text to extract Chinese-English phrase translation candidates, which are then validated using a classifier. Because Chinese text is similar to Japanese in that the word boundaries are not marked explicitly using white spaces, the same boundary detection problem exists as in '我的磁石(magnet)', in which only the underlined part corresponds to the translation of 'magnet'. Despite these similarities, the process of aligning the input and output sequences and using the alignment for the validation task is much more complex in the bilingual dictionary creation task, as it cannot be reformulated as a monotone substring mapping problem, and requires additional steps to generate word translation pairs from extracted string pairs.

Coming back to our pronunciation acquisition task, given that the orthography-pronunciation mapping is basically monotone without insertion or deletion, one can think of a very simple method for validating word-pronunciation candidate pairs which does not use a classifier. A finite-state acceptor can be used to search for a path through the hiragana pronunciation string, from right to left, emitting the corresponding kanji sequence using a fixed kanji pronunciation dictionary. When a valid path is found at the end of the pronunciation string, it returns success, with the emitted kanji character sequence, performing the left boundary detection as the same time. This method is expected to have near 100% precision, as the pronunciations for each kanji character is already validated in the dictionary. The recall, however, suffers from an incomplete coverage of the pronunciation dictionary. The pronunciation of kanji characters in Japanese reflects the effect of various morpho-phonological processes [30] as exemplified in Figure 4. These sound changes are reflected in the pronunciation (hiragana) orthography, but are often missing from the dictionary. We compare the performance of this baseline against the proposed method below.

[Rendaku (sequential voicing)]
神（かみ） kami ‘god’ + 棚（たな） tana ‘shelf’ → 神棚（かみだな） kami-dana, ‘altar’

[Renjo (liaison)]
反（はん）han ‘counter’+ 応（おう）ou ‘response’→ 反応（はんのう）han-nou, ‘reaction’

[Vowel alteration]
雨（あめ） ame, ‘rain’ + たれ tare, ‘drop’ → 雨だれ（あまだれ） ama-dare, ‘raindrop’

[Onbin (historical alterations around vowels)]
月（げつ）getsu ‘moon’+ 光（こう） kou ‘light’ → 月光（げっこう）geQkou ‘moonlight’ (Q indicates germination)

Fig. 4. Examples of morpho-phonological alterations in Japanese. Underlined characters indicate the changes in hiragana pronunciation strings.

4.1 Model and Features

As we saw in Section 3.3, not all pairs extracted from Wikipedia are valid word-pronunciation pairs. Also, there exists the boundary detection problem from the extracted instances. Thus, the problem we need to solve is twofold: to determine if a set of instances generated from a single original string contains a true word-pronunciation pair, and if it does, to determine which generated instance is the correct pair (i.e. boundary detection). One approach to this dual problem is to construct a ranker, which ranks each instance according to the likelihood of being a correct word-pronunciation pair. This is the approach employed by [3]. However, in our case, our preliminary experiments showed that the ranker approach is not necessary and a simple binary classifier which treats all instances separately will do, as the average number of distinct candidate pairs generated from a single extracted string is not very large (2.3 per extracted instance on average in our training data).

With this observation and the computational cost taken into consideration, we propose to use a binary classifier with MART (Multiple Additive Regression Trees; [7]), a widely used classification framework based on additive trees. The MART classifier was shown to outperform an averaged perceptron classifier with a substantial margin in our preliminary experiment on the development set.

Table 3 shows the list of features we used in the MART classifier. *LR* is the length ratio of pronunciation sequence to the kanji sequence. Since a kanji sequence is in

Table 3. Features used in the binary classifier

Feature	Description
LR	Length ratio (hiragana / kanji)
Align	Log of alignment score
Dist	Minimum value of phonologically-motivated edit distance to 20-best outputs of the transducer in Section 3
MT	Features from the transducer described in Section 3.1: p(*s*|*t*), p(*t*|*s*), p(*t*), operation count, phrase count, word count

principle never longer than the corresponding pronunciation sequence, [11] and one kanji character usually corresponds to one to three kana characters, this feature is expected to remove the pairs with obviously non-standard length ratio. *Align* is the log of the alignment score. *Dist* is the smallest edit distance to the 20-best pronunciation prediction results of the transducer described in Section 3. The process of generating this feature is as follows: first, the pronunciation candidate and the transducer outputs are transformed into Roman alphabets (e.g. 'かいぎ' ⇒ 'kaigi'), and the Levenshtein distances between the candidate and transducer outputs are calculated. The model uses the minimum of the edit distance values as the feature value. Crucially, we define some special zero-cost character replacement rules to address the Japanese morpho-phonological transformations as described in Figure 4, trying to capture as many pronunciation variants as possible. The followings are the rules we used; most of these alterations are described in [30].

- Rendaku: [k,t,s,sh] ⇔ [g, d, z, j]
- Vowel change: [e] ⇔ [a], [i] ⇔ [o], [o] ⇔ [a]
- On-bin: [mu] ⇒ [m, n], [tta] ⇒ [ta]
- Chinese dialect variation:[12] [k,s,t,n,h,m,r] ei ⇔ [k,sh,ch,n,h,m,r] ou/you

Although these rules are specific to Japanese pronunciations, the framework based on the extended edit distance with phonologically-motivated rules is generally applicable to any language with phonological transformations. Finally, *MT* indicates the features from the transduction model described in Section 3. They consist of the translation probabilities $\mathrm{P}(s|t)$ and $\mathrm{P}(t|s)$, target language model probability $\mathrm{P}(t)$, operation count, word count, and phrase count, as described in Section 3.

4.2 Experiments

Data and settings. We use the same Wikipedia data sets used in Section 3. From the training portion, we extracted 3,000 instances, and expanded them to generate the instances differing in the left boundary of the word, based on the heuristics described above. This resulted in 6,872 instances in total. Roughly 90% of the original instances before expansion were positive instances; after expansion, the number of negative instances increases, making the training instances more balanced. We then manually labeled each instance as positive or negative, indicating whether the pair is the correct word-pronunciation pair or not.

4.3 Results and Discussion

Table 4 shows the comparison of the models with various feature sets. The baseline is the finite-state acceptor method with a fixed dictionary, applied to the paired data

[11] Again, there are some exceptions to this such as <似而非, えせ> [ese]'pseudo'. However, they are very rare: we only found 14 cases in the 195K entries of UniDic nouns.

[12] This reflects the variations in the original Chinese pronunciations. For example, 清 has two common on-yomi pronunciations: [sei], which was imported around 7–8c from Tang dynasty, and [shou], imported earlier around 5–6c.

Table 4. Performance (in %) of yomi classification evaluated on labeled Wikipedia pairs, with five-fold cross validation

	Prec.	Recall	F1
Baseline	99.8	80.8	89.3
MART(LR)	56.8	62.9	59.7
MART(LR+Align)	94.3	90.9	92.5
MART(LR+Align+Dist)	97.8	96.2	97.0
MART(LR+Align+Dist+MT)	98.5	98.0	98.2

after candidate expansion. We used the dictionary mentioned in Footnote 4 for this purpose, which includes around 15K distinct kanji-pronunciation pairs. As expected, this baseline achieves a very high precision of 99.8%, but the recall is only around 80%, showing that the kanji pronunciation dictionary we used, though reasonably large, does not have satisfactory coverage for this task. The model with the LR feature performed poorly, with approximately 60% precision and recall, which is to be expected, as this feature by itself is a weak feature. The LR+Align model performs quite reasonably, with both the precision and recall over 90%. Adding the linguistically motivated edit distance features (LR+Align+Dist+MT) achieves a nice performance gain, with the F1 score achieving 97%. Finally, with all the features, we achieve the best performance of both precision and recall exceeding 98%.

Using this classifier, we were able to obtain ~420K word-pronunciation pairs from Wikipedia with 98.5% precision. To our knowledge, this is the largest Japanese word-pronunciation lexicon that is automatically generated.

5 Conclusion and Future Work

We have presented our approach to the task of Japanese pronunciation prediction. We have shown that the proposed pronunciation prediction model achieves the performance that is coming close to the level of human performance, and also that the model can be used effectively to harvest word-pronunciation pairs from unannotated text. We believe that the accuracy we achieve is sufficiently high to be used in realistic applications, such as text-to-speech and text input method. Measuring the contribution of this research in such application scenarios is one direction of future research.

We also plan to apply the proposed pronunciation acquisition technique to a much larger web-scale corpus. This step will be important for acquiring the pronunciation of the words other than nouns, as the Wikipedia data we used was dominated by noun-pronunciation pairs. The pronunciation extraction of the parts-of-speech that inflect (verbs and adjectives) are expected to be more challenging, as the parenthetical pronunciation aids are inserted within a word, rather than at the word boundary, as in 醒（さ）めた [sa-me-ta] ‘awake’. Although newly created words tend to be nouns, predicting the pronunciation of non-nouns will be important when we use the methods proposed in this paper for the task of predicting pronunciations at the sentence level.

References

1. Bisani, M., Ney, H.: Investigations on joint-multigram models for grapheme-to-phoneme conversion. In: The Proceedings of the International Conference on Spoken Language Processing (2002)
2. Bisani, M., Ney, H.: Joint-sequence models for grapheme-to-phoneme conversion. Speech Communication 50(5), 434–451 (2008)
3. Cao, G., Gao, J., Nie, J.-Y.: A system to mine large-scale bilingual dictionaries from monolingual web pages. In: MT Summit XI (2007)
4. Chen, S.F.: Conditional and joint models for grapheme-to-phoneme conversion. In: The Proceedings of the European Conference on Speech Communication and Technology (2003)
5. Cherry, C., Suzuki, H.: Discriminative Substring Decoding for Transliteration. In: EMNLP (2009)
6. Den, Y., Ogiso, T., Ogura, H., Yamada, A., Minematsu, N., Uchimoto, K., Koiso, H.: The development of an electronic dictionary for morphological analysis and its application to Japanese corpus linguistics. Japanese linguistics 22, 101–122 (2007) (in Japanese)
7. Friedman, J.H.: Greedy function approximation: a gradient boosting machine. Ann. Statist. 29(5), 1189–1232 (2001)
8. Gao, J., Goodman, J., Li, M., Lee, K.-F.: Toward a unified approach to statistical language modeling for Chinese. ACM Transactions on Asian Language Information Processing 1(1), 3–33 (2002a)
9. Gao, J., Suzuki, H., Wen, Y.: Exploiting headword dependency and predictive clustering for language modeling. In: EMNLP 2002 (2002b)
10. Ghoshal, A., Jansche, M., Khudanpur, S., Riley, M., Ulinski, M.: Web-derived Pronunciations. In: ICASSP (2009)
11. Jiampojamarn, S., Kondrak, G., Sherif, T.: Applying many-to-many alignments and hidden markov models to letter-to-phoneme conversion. In: HLT-NAACL (2007)
12. Jiampojamarn, S., Cherry, C., Kondrak, G.: Joint Processing and Discriminative Training for Letter-to-Phoneme Conversion. In: ACL (2008)
13. Jiampojamarn, S., Cherry, C., Kondrak, G.: Integrating Joint n-gram Features into a Discriminative Training Framework. In: The Proceedings of NAACL (2010)
14. Knight, K., Graehl, J.: Machine Transliteration. Computational Linguistics 24(4) (1998)
15. Koehn, P., Och, F., Marcu, D.: Statistical phrase-based translation. In: NAACL (2003)
16. Kurata, G., Mori, S., Suitoh, N., Nishimura., M.: Unsupervised lexicon acquisition from speech and text. In: The Proceedings of ICASSP (2007)
17. Li, H., Zhang, M., Su, J.: A joint source-channel model for machine transliteration. In: ACL (2004)
18. Lin, D., Zhao, S., Durme, B.V., Pasca, M.: Mining parenthetical translations from the web by word alignment. In: ACL 2008 (2008)
19. Maekawa, K.: Compilation of the KOTONOHA-BCCWJ Corpus. Nihongo no kenkyu (Studies in Japanese) 4(1), 82–95 (2008) (in Japanese)
20. Mori, S., Neubig, G.: Automatically improving language processing accuracy by using kana-kanji conversion logs. In: The Proc. of the 16th Annual Meeting of the Association for NLP (2010) (in Japanese)
21. Mori, S., Sasada, T., Neubig, G.: Language Model Estimation from a Stochastically Tagged Corpus. Technical Report, SIG, Information Processing Society of Japan (2010) (in Japanese)

22. Nagano, T., Mori, S., Nishimura, M.: An n-gram-based approach to phoneme and accent estimation for TTS. Transactions of Information Processing Society of Japan 47(6), 1793–1801 (2006) (in Japanese)
23. Och, F.J.: Minimum Error Rate Training for Statistical Machine Translation. In: ACL (2003)
24. Reddy, S., Goldsmith, J.: An MDL-based approach to extracting subword units for grapheme-to-phoneme conversion. In: NAACL (2010)
25. Sasada, T., Mori, S., Kawahara, T.: Extracting word-pronunciation pairs from comparable set of text and speech. In: The Proceedings of the 9th Annual Conference of the International Speech Communication Association (2008)
26. Sasada, T., Mori, S., Kawahara, T.: Domain adaptation of statistical kana-kanji conversion system by automatic acquisition of contextual information with unknown words. In: The Proceedings of the 15th Annual Meeting of the Association for NLP (2009) (in Japanese)
27. Schroeter, J., Conkie, A., Syrdal, A., Beutnagel, M., Jilka, M., Strom, V., Kim, Y.-J., Kang, H.-G., Kapilow, D.: A perspective on the next challenges for TTS research. In: The Proceedings of the IEEE 2002 Workshop on Speech Synthesis (2002)
28. Sherif, T., Kondrak, G.: Substring-based transliteration. In: ACL (2007)
29. Sumita, E., Sugaya, F.: Word Pronunciation Disambiguation using the Web. In: NAACL (2006)
30. Vance, T.J.: An introduction to Japanese phonology. State University of New York Press (1987)
31. Zens, R., Ney, H.: Improvements in Phrase-Based Statistical Machine Translation. In: HLT-NAACL (2004)
32. Zhang, H., Quirk, C., Moore, R.C., Gildea, D.: Bayesian learning of non-compositional phrases with synchronous parsing. In: ACL (2008)