Discovery of Term Variation in Japanese Web Search Queries

Hisami Suzuki, Xiao Li, and Jianfeng Gao
Microsoft Research, Redmond
One Microsoft Way, Redmond, WA 98052 USA
{hisamis,xiaol,jfgao}@microsoft.com

Abstract

In this paper we address the problem of identifying a broad range of term variations in Japanese web search queries, where these variations pose a particularly thorny problem due to the multiple character types employed in its writing system. Our method extends the techniques proposed for English spelling correction of web queries to handle a wider range of term variants including spelling mistakes, valid alternative spellings using multiple character types, transliterations and abbreviations. The core of our method is a statistical model built on the MART algorithm (Friedman, 2001). We show that both string and semantic similarity features contribute to identifying term variation in web search queries; specifically, the semantic similarity features used in our system are learned by mining user session and click-through logs, and are useful not only as model features but also in generating term variation candidates efficiently. The proposed method achieves 70% precision on the term variation identification task with the recall slightly higher than 60%, reducing the error rate of a naïve baseline by 38%.

1 Introduction

Identification of term variations is fundamental to many NLP applications: words (or more generally, terms) are the building blocks of NLP applications, and any robust application must be able to handle variations in the surface representation of terms, be it a spelling mistake, valid spelling variation, or abbreviation. In search applications, term variations can be used for query expansion, which generates additional query terms for better matching with the terms in the document set. Identifying term variations is also useful in other scenarios where semantic equivalence of terms is sought, as it represents a very special case of paraphrase.

This paper addresses the problem of identifying term variations in Japanese, specifically for the purpose of query expansion in web search, which appends additional terms to the original query string for better retrieval quality. Query expansion has been shown to be effective in improving web search results in English, where different methods of generating the expansion terms have been attempted, including relevance feedback (e.g., Salton and Buckley, 1990), correction of spelling errors (e.g., Cucerzan and Brill, 2004), stemming or lemmatization (e.g., Frakes, 1992), use of manually- (e.g., Aitchison and Gilchrist, 1987) or automatically- (e.g., Rasmussen 1992) constructed thesauri, and Latent Semantic Indexing (e.g., Deerwester et al, 1990). Though many of these methods can be applied to Japanese query expansion, there are unique problems posed by Japanese search queries, the most challenging of which is that valid alternative spellings of a word are extremely common due to the multiple script types employed in the language. For example, the word for 'protein' can be spelled as たんぱくしつ, タンパク質, 蛋白質, たん白質 and so on, all pronounced *tanpakushitsu* but using combinations of different script types. We give a detailed description of the problem posed by the Japanese writing system in Section 2. Though there has been previous work on addressing specific subsets of spelling alterations within and across character types in Japanese, there has not been any comprehensive solution for the purpose of query expansion.

Our approach to Japanese query expansion is unique in that we address the problem comprehensively: our method works independently of the character types used, and targets a wide range of term variations that are both orthographically and semantically similar, including spelling errors, valid alternative spellings, transliterations and abbreviations. As described in Section 4, we define the problem of term variation identifica-

Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, pages 1484–1492,
Singapore, 6-7 August 2009.

tion as a binary classification task, and build two types of classifiers according to the maximum entropy model (Berger et al., 1996) and the MART algorithm (Friedman, 2001), where all term similarity metrics are incorporated as features and are jointly optimized. Another important contribution of our approach is that we derive our semantic similarity models by mining user query logs, which has been explored for the purposes of collecting related words (e.g., Jones et al., 2006a), improving search results ranking (e.g., Craswell and Szummer, 2007) and learning query intention (e.g., Li et al., 2008), but not for the task of collecting term variations. We show that our semantic similarity models are not only effective in the term variation identification task, but also for generating candidates of term variations much more efficiently than the standard method whose candidate generation is based on edit distance metrics.

2 Term Variations in Japanese

In this section we give a summary of the Japanese writing system and the problem it poses for identifying term variations, and define the problem we want to solve in this paper.

2.1 The Japanese Writing System

There are four different character types that are used in Japanese text: *hiragana*, *katakana*, *kanji* and Roman alphabet. Hiragana and katakana are the two subtypes of *kana* characters, which are syllabic character sets, each with about 50 basic characters. There is a one-to-one correspondence between hiragana and katakana characters, and, as they are phonetic, they can be unambiguously converted into a sequence of Roman characters. For example, the word for 'mackerel' is spelled in hiragana as さば or in katakana as サバ, both of which can be transcribed in Roman characters as *saba*, which is how the word is pronounced. Kanji characters, on the other hand, are ideographic and therefore numerous – more than 5,000 are in common usage. One difficulty in handling Japanese kanji is that each character has multiple pronunciations, and the correct pronunciation is determined by the context in which the character is used. For instance, the character 行 is read as *kou* in the word 銀行 *ginkou* 'bank', *gyou* in 行 'column', and *i* or *okona* in 行った *itta* 'went' or *okonatta* 'done' depending on the context in which the word is used.[1] Proper name readings are particularly difficult to disambiguate, as their pronunciation cannot be inferred from the context (they tend to have the same grammatical function) or from the dictionary (they tend to be out-of-vocabulary). Therefore, in Japanese, computing a pronunciation-based edit distance metric is not straightforward, as it requires estimating the readings of kanji characters.

2.2 Term Variation by Character Type

Spelling variations are commonly observed both within and across character types in Japanese. Within a character type, the most prevalent is the variation observed in katakana words. Katakana is used to transliterate words from English and other foreign languages, and therefore reflects the variations in the sound adaptation from the source language. For example, the word 'spaghetti' is transliterated into six different forms (スパゲッティ *supagetti*, スパゲッティー *supagettii*, スパゲッテイ *supagettei*, スパゲティ *supageti*, スパゲティー *supagetii*, スパゲテイ *supagetei*) within a newspaper corpus (Masuyama et al., 2004).

Spelling variants are also prevalent across character types: in theory, a word can be spelled using any of the character types, as we have seen in the example for the word 'protein' in Section 1. Though there are certainly preferred character types for spelling each word, variations are still very common in Japanese text and search queries. Alterations are particularly common among hiragana, katakana and kanji (e.g. さば~サバ~ 鯖 *saba* 'mackerel'), and between katakana and Roman alphabet (e.g. フェデックス *fedekkusu* fedex). This latter case constitutes the problem of transliteration, which has been extensively studied in the context of machine translation (e.g. Knight and Graehl, 1998; Bilac and Tanaka, 2004; Brill et al., 2001).

2.3 Term Variation by Re-write Categories

Table 1 shows the re-write categories of related terms observed in web query logs, drawing on our own data analysis as well as on previous work such as Jones et al. (2006a) and Okazaki et al. (2008b). Categories 1 though 9 represent strictly synonymous relations; in addition, terms in Categories 1 through 5 are also similar orthographically or in pronunciation. Categories 10

[1] In a dictionary of 200K entries, we find that on average each kanji character has 2.5 readings, with three characters (直,生,空) with as many as 11 readings.

Categories	Example in English	Example in Japanese
1. Spelling mistake	aple ~ apple	グウグル *guuguru* ~ グーグル *gu-guru* 'google'
2. Spelling variant	color ~ colour	さば~サバ~鯖; スパゲティ~スパゲッティー (Cf. Sec.2.2)
3. Inflection	matrix ~ matrices	作る *tsukuru* 'make' ~ 作った *tsukutta* 'made'
4. Transliteration		フェデックス ~ fedex 'Fedex'
5. Abbreviation/ Acronym	macintosh ~ mac	世界銀行 *sekaiginkou* ~ 世銀 *segin* 'World Bank'; マクドナルド *makudonarudo* ~ マック *makku* 'McDonald's'
6. Alias	republican party ~ gop	フランス *furansu* ~ 仏 *futsu* 'France'
7. Translation	パキスタン大使館 *pakisutantaishikan* ~ Pakistan embassy	
8. Synonym	carcinoma ~ cancer	暦 *koyomi* ~ カレンダー *karendaa* 'calendar'
9. Abbreviation (user specific)	mini ~ mini cooper	クロネコヤマト *kuronekoyamato* ~ クロネコ *kuroneko* (name of a delivery service company)
10. Generalization	nike shoes ~ shoes	シビック 部品 *shibikku buhin* 'Civic parts' ~ 車 部品 *kuruma buhin* 'car parts'
11. Specification	ipod ~ ipod nano	東京駅 *toukyoueki* 'Tokyo station' ~ 東京駅時刻表 *toukyouekijikokuhyou* 'Tokyo station timetable'
12. Related	windows ~ microsoft	トヨタ *toyota* 'Toyota' ~ ホンダ *honda* 'Honda'

Table 1: Categories of Related Words Found in Web Search Logs

through 12, on the other hand, specify non-synonymous relations.

Different sets out of these categories can be useful for different purposes. For example, Jones et al (2006a; 2006b) target all of these categories, as their goal is to collect related terms as broadly as possible for the application of sponsored search, i.e., mapping search queries to a small corpus of advertiser listings. Okazaki et al. (2008b) define their task narrowly, to focusing on spelling variants and inflection, as they aim at building lexical resources for the specific domain of medical text.

For web search, a conservative definition of the task as dealing only with spelling errors has been successful for English; a more general definition using related words for query expansion has been a mixed blessing as it compromises retrieval precision. A comprehensive review on this topic is provided by Baeza-Yates and Ribeiro-Neto (1999). In this paper, therefore, we adopt a working definition of the term variation identification task as including Categories 1 through 5, i.e., those that are synonymous and also similar in spelling or in pronunciation.[2] This definition is reasonably narrow so as to make automatic discovery of term variation pairs realistic, while covering all common cases of term variation in Japanese, including spelling variants and transliterations. It is also appropriate for the purpose of query expansion: because term variation defined in this manner is based on spelling or pronunciation similarity, their meaning and function tend to be completely equivalent, as opposed to Categories 6 through 9, where synonymy is more context- or user-dependent. This will ensure that the search results by query expansion will avoid the problem of compromised precision.

3 Related Work

In information retrieval, the problem of vocabulary mismatch between the query and the terms in the document has been addressed in many ways, as mentioned in Section 1, achieving varying degrees of success in the retrieval task. In particular, our work is closely related to research in spelling correction for English web queries (e.g., Cucerzan and Brill, 2004; Ahmad and Kondrak, 2005; Li et al., 2006; Chen et al., 2007). Among these, Li et al. (2006) and Chen et al. (2007) incorporate both string and semantic similarity in their discriminative models of spelling correction, similarly to our approach. In Li et al. (2006), semantic similarity was computed as distributional similarity of the terms using query strings in the log as context. Chen et al. (2007) point out that this method suffers from the data sparseness problem in that the statistics for rarer terms are unreliable, and propose using web search results as extended contextual information. Their method, however, is expensive as it requires web search results for each query-candidate pair, and also because their candidate set, generated using an edit distance function and phonetic similarity from query log data, is impractically large and must be pruned by using a language model. Our approach differs from these methods in that we exploit user query logs to derive semantic knowledge of terms, which is

[2] In reality, Category 3 (Inflection) is extremely rare in Japanese web queries, because nouns do not inflect in Japanese, and most queries are nominals.

used both for the purpose of generating a candidate set efficiently and as features in the term variation identification model.

Acquiring semantic knowledge from a large quantity of web query logs has become popular in recent years. Some use only query strings and their counts for learning word similarity (e.g., Sekine and Suzuki, 2007; Komachi and Suzuki, 2008), while others use additional information, such as the user session information (i.e., a set of queries issued by the same user within a time frame, e.g., Jones et al., 2006a) or the URLs clicked as a result of the query (e.g., Craswell and Szummer, 2007; Li et al., 2008). This additional data serves as an approximation to the meaning of the query; we use both user session and click-through data for discovering term variations.

Our work also draws on some previous work on string transformation, including spelling normalization and transliteration. In addition to the simple Levenshtein distance, we also use generalized string-to-string edit distance (Brill and Moore, 2000), which we trained on aligned katakana-English word pairs in the same manner as Brill et al. (2001). As mentioned in Section 2.2, our work also tries to address the individual problems targeted by such component technologies as Japanese katakana variation, English-to-katakana transliteration and katakana-to-English back-transliteration in a unified framework.

4 Discriminative Model of Identifying Term Variation

Recent work in spelling correction (Ahmed and Kondrak, 2005; Li et al., 2006; Chen et al., 2007) and normalization (Okazaki et al., 2008b) formulates the task in a discriminative framework:

$$c^* = \text{argmax}_{c \in \text{gen}(q)} P(c|q)$$

This model consists of two components: gen(q) generates a list of candidates $C(q)$ for an input query q, which are then ranked by the ranking function $P(c|q)$. In previous work, gen(q) is typically generated by using an edit distance function or using a discriminative model trained for its own purpose (Okazaki et al., 2008b), often in combination with a pre-complied lexicon. In the current work, we generate the list of candidates by learning pairs of queries and their re-write candidates automatically from query session and click logs, which is far more robust and efficient than using edit distance functions. We describe our candidate generation method in detail in Section 5.1.

Unlike the spelling correction and normalization tasks, our goal is to *identify* term variations, i.e., to determine whether each query-candidate pair (q,c) constitutes a term variation or not. We formulate this problem as a binary classification task. There are various choices of classifiers for such a task: we chose to build two types of classifiers that make a binary decision based on the probability distribution $P(c|q)$ over a set of feature functions $f_i(q,c)$. In maximum entropy framework, this is defined as:

$$P(c|q) = \frac{\exp \sum_{i=1}^{K} \lambda_i f_i(c,q)}{\sum_c \exp \sum_{i=1}^{K} \lambda_i f_i(c,q)}$$

where $\lambda_1,\ldots, \lambda_k$ are the feature weights. The optimal set of feature weights λ^* is computed by maximizing the log-likelihood of the training data. We used stochastic gradient descent for training the model with a Gaussian prior.

The second classifier is built on MART (Friedman, 2001), which is a boosting algorithm. At each boosting iteration, MART builds a regression tree to model the functional gradient of the cost function (which is cross entropy in our case), evaluated on all training samples. MART has three main parameters: M, the total number of boosting iterations, L, the number of leaf nodes for each regression tree, and v, the learning rate. The optimal values of these parameters can be chosen based on performance on a validation set. In our experiments, we found that the performance of the algorithm is relatively insensitive to these parameters as long as they are in a reasonable range: given the training set of a few thousand samples or more, as in our experiments, M=100, L=15, and v=0.1 usually give good performance. Smaller trees and shrinkage may be used if the training data set is smaller.

The classifiers output a binary decision according to $P(c|q)$: positive when $P(c|q) > 0.5$ and negative otherwise.

5 Experiments

5.1 Candidate Generation

We used a set of Japanese query logs collected over one year period in 2007 and 2008. More specifically, we used two different extracts of log data for generating term variation candidate pairs:

Query session data. From raw query logs, we extracted pairs of queries q_1 and q_2 such that they are (i) issued by the same user; (ii) q_2 follows within 3 minutes of issuing q_1; and (iii) q_2 generated at least one click of a URL on the result

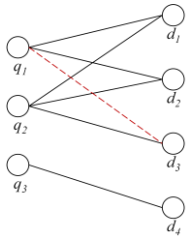

Figure 1. Random Walk Algorithm

page while q_1 did not result in any click. We then scored each query pair (q_1,q_2) in this subset using the log-likelihood ratio (LLR, Dunning, 1993) between q_1 and q_2, which measures the mutual dependence within the context of web search queries (Jones et al., 2006a). After applying an LLR threshold (LLR > 15) and a count cutoff (we used only the top 15 candidate q_2 according to the LLR value for each q_1), we obtained a list of 47,139,976 pairs for the 14,929,497 distinct q_1, on average generating 3.2 candidates per q_1[3]. We took this set as comprising query-candidate pairs for our model, along with the set extracted by click-through data mining explained below.

Click-through data. This data extract is based on the idea that if two queries led to the same URLs being repeatedly clicked, we can reasonably infer that the two queries are semantically related. This is similar to computing the distributional similarity of terms given the context in which they appear, where context is most often defined as the words co-occurring with the terms. Here, the clicked URLs serve as their context.

One challenge in using the URLs as contextual information is that the contextual representation in this format is very sparse, as user clicks are rare events. To learn query similarities from incomplete click-through data, we used the random walk algorithm similar to the one described in Craswell and Szummer (2007). Figure 1 illustrates the basic idea: initially, document d_3 has a click-through link consisting of query q_2 only; the random walk algorithm adds the link from d_3 to q_1, which has a similar click pattern as q_2. Formally, we construct a click graph which is a bipartite-graph representation of click-through data. We use $\{q_i\}_{i=1}^{m}$ to represent a set of query nodes and $\{d_j\}_{j=1}^{n}$ a set of document nodes. We further define an $m \times n$ matrix W in which element W_{ij} represents the click count associated with (q_i, d_j). This matrix can be normalized to be a query-to-document transition matrix, denoted by A, where $A_{ij} = p^{(1)}(d_j|q_i)$ is the probability that q_i transits to d_j in one step. Similarly, we can normalize the transpose of W to be a document-to-query transition matrix, denoted by B, where $B_{j,i} = p^{(1)}(q_i|d_j)$. It is easy to see that using A and B we can compute the probability of transiting from any node to any other node in k steps. In this work, we use a simple measure which is the probability that one query transits to another in two steps, and the corresponding probability matrix is given by AB.

We used this probability and ranked all pairs of queries in the same raw query logs as in the query session data described above to generate additional candidates for term variation pairs. 20,308,693 pairs were extracted after applying the count cutoff of 5, generating on average 6.8 candidates for 2,973,036 unique queries.

It is interesting to note that these two data extracts are quite complementary: of all the data generated, only 4.2% of the pairs were found in both the session and click-through data. We believe that this diversity is attributable to the nature of the extracts: the session data tends to collect the term pairs that are issued by the same user as a result of conscious re-writing effort, such as typing error corrections and query specifications (Categories 1 and 11 in Table 1), while the click-though data collects the terms issued by different users, possibly with different intentions, and tends to include many spelling variants, synonyms and queries with different specificity (Categories 2, 8, 10 and 11).

5.2 Features

We used the same set of features for the maximum entropy and MART models, which are given in Table 2. They are divided into three main types: string similarity features (1-16), semantic similarity features (17, 18), and character type features (19-39). Among the string similarity features, half of them are based on Levenshtein distance applied to surface forms (1-8), while the other half is based on Levenshtein and string-to-string edit distance metrics computed over the Romanized form of the query, reflecting its pronunciation. The conversion into Roman characters was done deterministically for kana characters using a simple mapping table. For Romanizing kanji characters, we used the function available from Windows IFELanguage API (version

[3] We consider each query as an unbreakable term in this paper, so term variation is equivalent to query variation.

String similarity features (16 real-valued features)
1. Lev distance on surface form
2. Lev distance on surface form normalized by *q1* length
3. Lev distance on surface form using character equivalence table
4. Lev distance on surface form normalized by *q1* length using character equivalence table
5. Lev distance on surface form w/o space
6. Lev distance on surface form normalized *q1* length w/o space
7. Lev distance on surface form using character equivalence table w/o space
8. Lev distance on surface form normalized by *q1* using character equivalence table w/o space
9. Lev distance on Roman
10. Lev distance on Roman normalized by *q1* length
11. Alpha-beta edit distance on Roman
12. Alpha-beta edit distance on Roman normalized by *q1* length
13. Lev distance on Roman w/o space
14. Lev distance on Roman normalized by *q1* length w/o space
15. Alpha-beta edit distance on Roman w/o space
16. Alpha-beta edit distance on Roman normalized by *q1* length w/o space
Features for semantic similarity (2 real-valued features)
17. LLR score
18. Click-though data probability
Character type features (21 binary features)
19. BothHira, 20. BothKata, 21. BothRoman, 22. BothKanji, 23. BothMixedNoKanji, 24. BothMixed, 25. HiraKata, 26. HiraKanji, 27. HiraRoman, 28. HiraMixedNoKanji, 29. HiraMixed, 30. KataKanji, 31.KataRoman, 32. KataMixedNoKanji, 33. KataMixed, 34. KanjiRoman, 35. KanjiMixedNoKanji, 36. KanjiMixed, 37. RomanMixedNoKanji, 38. RomanMixed, 39. MixedNoKanjiMixed

Table 2: Classifier Features

2).[4] The character equivalence table mentioned in the features 3,4,7,8 is a table of 643 pairs of characters that are known to be equivalent, including kanji allography (same kanji in different graphical styles). The alpha-beta edit distance (11, 12, 15, 16) is the string-to-string edit distance proposed in Brill and Moore (2001), which we trained over about 60K parallel English-to-katakana Wikipedia title pairs, specifically to capture the edit operations between English and katakana words, which are different from the edit operations between two Japanese words. Semantic similarity features (17, 18) use the LLR score from the session data, and the click-though pair probability described in the subsection above. Finally, features 19-39 capture the script types of the query-candidate pair. We first defined six basic character types for each query or candidate: Hira (hiragana only), Kata (katakana only), Kanji (kanji only), Roman (Roman alphabet only), MixedNoKanji (includes more than one character sets but not kanji) and Mixed (includes more than one character sets with kanji). We then derived 21 binary features by concatenating these basic character type features for the combination of query and candidate strings. For example, if both the query and candidate are in hiragana, BothHira will be on; if the query is Mixed and the candidate is Roman, then RomanMixed will be on. Punctuation characters and Arabic numerals were treated as being transparent to character type assignment. The addition of these features is motivated by the assumption that appropriate types of edit distance operations might depend on different character types for the query-candidate pair.

Since the dynamic ranges of different features can be drastically different, we normalized each feature dimension to a normal variable with zero-mean and unit-variance. We then used the same normalized features for both the maximum entropy and the MART classifiers.

5.3 Training and Evaluation Data

In order to generate the training data for the binary classification task, we randomly sampled the query session (5,712 samples) and click-through data (6,228 samples), and manually labeled each pair as positive or negative: the positive label was assigned when the term pair fell into Categories 1 through 5 in Table 1; otherwise it was assigned a negative label. Only 364 (6.4%) and 244 (3.9%) of the samples were positive examples for the query session and click-through data respectively, which makes the baseline per-

[4] http://msdn.microsoft.com/en-us/library/ms970129.aspx. We took the one-best conversion result from the API. The conversion accuracy on a randomly sampled 100 kanji queries was 89.6%.

formance of the classifier quite high (always predict the negative label – the accuracy will be 95%). Note, however, that these data sets include term variation candidates much more efficiently than a candidate set generated by the standard method that uses an edit distance function with a threshold. For example, there is a query-candidate pair *q*=家風情報 *kafuujouhou* 'house-style information' *c*= 花 粉 情 報 *kafunjouhou* 'pollen information') in the session data extract, the first one of which is likely to be a misspelling of the second.[5] If we try to find candidates for the query 家風情報 using an edit distance function naively with a threshold of 2 from the queries in the log, we end up collecting a large amount of completely irrelevant set of candidates such as 台風情報 *taifuujouhou* 'typhoon information', 株情報 *kabu jouhou* 'stock information', 降雨情報 *kouu jouhou* 'rainfall information' and so on – as many as 372 candidates were found in the top one million most frequent queries in the query log from the same period; for rarer queries these numbers will only be worse. Computing the edit distance based on the pronunciation will not help here: the examples above are within the edit distance of 2 even in terms of Romanized strings.

Another advantage of generating the annotated data using the result of query log data mining is that the annotation process is less prone to subjectivity than creating the annotation from scratch. As Cucerzan and Brill (2004) point out, the process of manually creating a spelling correction candidate is seriously flawed as the intention of the original query is completely lost: for the query *gogle*, it is not clear out of context if the user meant *goggle*, *google*, or *gogle*. Using data mined from query logs solves this problem: an annotator can safely assume that if *gogle-goggle* appears in the candidate set, it is very likely to be a valid term variation intended by the user. This makes the annotation more robust and efficient: the inter-annotator agreement rate for 2,000 query pairs by two annotators was 95.7% on our data set, each annotator spending only about two hours to annotate 2,000 pairs.

5.4 Results and Discussion

In order to compare the performance of two classifiers, we first built maximum entropy and MART classifiers as described in Section 4 using all the features in Section 5.2. We run five experiments using different random split of training and test data: in each run, we used 10,000 samples for training and the remaining 1,940 samples for testing, and measured the performance of the two classifiers on the task of term variation identification in terms of the error rate i.e., 1–accuracy. The results, average over five runs, were 4.18 for the maximum entropy model, and 3.07 for the MART model. In all five runs, the MART model outperformed the maximum entropy classifier. This is not surprising given the superior performance of tree-boosting algorithms previously reported on similar classification tasks (e.g., Hastie et al., 2001). In our task where different types of features are likely to perform better when they are combined (such as semantic features and character types features), MART would be a better fit than linear classifiers because the decision trees generated by MART optimally combines features in the local sense. In what follows, we only discuss the results produced by MART for further experiments. Note that the baseline classifier, which always predicts the label to be negative, achieves 95.04% in accuracy (or 4.96% error rate), which sounds extremely high, but in fact this baseline classifier is useless for the purpose of collecting term variations, as it learns none of them by classifying all samples as negative.

Features	Error rate (%)
A. All features (1-39 in Table 2)	3.07
B. String features only (1-16)	3.49
C. Surface string features only (1-8)	4.9
D. No semantic feats (1-16,19-39)	3.28
E. No character type feats (1-18)	3.5

Table 3: Results of Features Ablation Experiments Using MART Model

For evaluating the contribution of different types of features in Section 5.2, we performed feature ablation experiments using MART. Table 3 shows the results in error rate by various MART classifiers using different combination of features. The results in this table are also averaged over five run with random training/test data split. From Table 3, we can see that the best performance was achieved by the model using all features (line A of the table), which reduces the baseline error rate (4.96%) by 38%. The improvement is statistically significant according to the McNemar test ($P < 0.05$). Models that use string edit distance features only (lines B and C) did not perform well: in particular, the model that uses surface edit distance features only

[5] 家風情報 does not make any sense in Japanese; on the other hand, information about cedar pollen is commonly sought after in spring due to widespread pollen allergy.

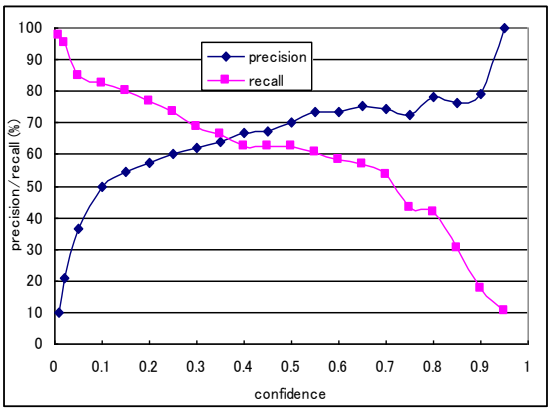

Figure 2: Precision/Recall Curve of MART

without considering the term pronunciation performed horribly (line C), which confirms the results reported by Jones et al. (2006b). However, unlike Jones et al. (2006b), we see a positive contribution of semantic features: the use of semantic features reduced the error rate from 3.28 (line D) to 3.07 (line A), which is statistically significant. This may be attributable to the nature of semantic information used in our experiments: we used the user session and click-though data to extract semantic knowledge, which may be semantically more specific than the probability of word substitution in a query collection as a whole, which is used by Jones et al. (2006b). Finally, the character type features also contributed to reducing the error rate (lines A and E). In particular, the observation that the addition of semantic features without the character type features (comparing lines B and E) did not improve the error rate indicates that the character type features are also important in bringing about the contribution of semantic features.

Figure 2 displays the test data precision/recall curve of one of the runs of MART that uses all features. The x-axis of the graph is the confidence score of classification $P(c|q)$, which was set to 0.5 for the results in Table 3. At this confidence, the model achieves 70% precision with the recall slightly higher than 60%. In the graph, we observe a familiar trade-off between precision and recall, which is useful for practical applications that may favor one over the other.

In order to find out where the weaknesses of our classifiers lie, we performed a manual error analysis on the same MART run whose results are shown in Figure 2. Most of the classification errors are false negatives, i.e., the model failed to predict a case of term variation as such. The most conspicuous error is the failure to capture abbreviations, such as failing to capture the alteration between 十条中学校 *juujoochuugakkou* 'Juujoo middle school' and 十条中 *juujoochuu*, which our edit distance-based features fail as the length difference between a term and its abbreviation is significant. Addition of more targeted features for this subclass of term variation (e.g., Okazaki et al., 2008a) is called for, and will be considered in future work. Mistakes in the Romanization of kanji characters were not always punished as the query and the candidate string may contain the same mistake, but when they occurred in either in the query or the candidate string (but not in both), the result was destructive: for example, we assigned a wrong Romanization on 水銀燈 as *suiginnakari* 'mercury lamp', as opposed to the correct *suiginntou*, which causes the failure to capture the alteration with 水銀痘 *suiginntou*, (a misspelling of 水銀燈). Using N-best (N>1) candidate pronunciations for kanji terms or using all possible pronunciations for kanji characters might reduce this type of error. Finally, the features of our models are the edit distance functions themselves, rather than the individual edit rules or operations. Using these individual operations as features in the classification task directly has been shown to perform well on spelling correction and normalization tasks (e.g., Brill and Moore, 2000; Okazaki et al., 2008b). Okazaki et al.'s (2008b) method of generating edit operations may not be viable for our purposes, as they assume that the original and candidate strings are very similar in their surface representation – they target only spelling variants and inflection in English. One interesting future avenue to consider is to use the edit distance functions in our current model to select a subset of query-candidate pairs that are similar in terms of these functions, separately for the surface and Romanized forms, and use this subset to align the character strings in these query-candidate pairs as described in Brill and Moore (2000), and add the edit operations derived in this manner to the term variation identification classifier as features.

6 Conclusion

In this paper we have addressed the problem of acquiring term variations in Japanese query logs for the purpose of query expansion. We generate term variation candidates efficiently by mining query log data, and our best classifier, based on the MART algorithm, can make use of both edit-distance-based and semantic features, and can identify term variation with the precision of 70% at the recall slightly higher than 60%. Our next

goal is to use and evaluate the term variation collected by the proposed method in an actual search scenario, as well as improving the performance of our classifier by using individual, character-dependent edit operations as features in classification.

References

Ahmad, Farooq, and Grzegorz Kondrak. 2005. Learning a spelling error model from search query logs. In *Proceedings of EMNLP*, pp.955-962.

Aitchison, J. and A. Gilchrist. 1987. *Thesaurus Construction: A Practical Manual*. Second edition. ASLIB, London.

Aramaki, Eiji, Takeshi Imai, Kengo Miyo, and Kazuhiko Ohe. 2008. Orthographic disambiguation incorporating transliterated probability. In *Proceedings of IJCNLP*, pp.48-55.

Baeza-Yates, Ricardo, and Berthier Ribeiro-Neto. 1999. *Modern Information Retrieval*. Addison Wesley.

Berger, A.L., S. A. D. Pietra, and V. J. D. Pietra. 1996. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1): 39-72.

Bilac, Slaven, and Hozumi Tanaka. 2004. A hybrid back-transliteration system for Japanese. In *Proceedings of COLING*, pp.597-603.

Brill, Eric, Gary Kacmarcik and Chris Brockett. 2001. Automatically harvesting katakana-English term pairs from search engine query logs. In *Proceedings of the Sixth Natural Language Processing Pacific Rim Symposium* (NLPRS-2001), pp.393-399.

Brill, Eric, and Robert C. Moore. 2000. An improved error model for noisy channel spelling. In *Proceedings of ACL*, pp.286-293.

Chen, Qing, Mu Li and Ming Zhou. 2007. Improving query spelling correction using web search results. In *Proceedings of EMNLP-CoNLL*, pp.181-189.

Craswell, Nick, and Martin Szummer. 2007. Random walk on the click graph. In *Proceedings of SIGIR*.

Cucerzan, Silviu, and Eric Brill. 2004. Spelling correction as an iterative process that exploits the collective knowledge of web users. In *Proceedings of EMNLP*, pp.293-300.

Deerwester, S., S.T. Dumais, T. Landauer and Harshman. 1990. Indexing by latent semantic analysis. In *Journal of the American Society for Information Science*, 41(6): 391-407.

Dunning, Ted. 1993. Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics*, 19(1): 61-74.

Frakes, W.B. 1992. Stemming algorithm. In W.B.Frakes and R.Baeza-Yates (eds.), *Information Retrieval: Data Structure and Algorithms*, Chapter 8. Prentice Hall.

Friedman, J. 2001. Greedy function approximation: a gradient boosting machine. *Annals of Statistics*, 29(5).

Jones, Rosie, Benjamin Rey, Omid Madani and Wiley Greiner. 2006a. Generating query substitutions. In *Proceedings of WWW*, pp.387–396.

Jones, Rosie, Kevin Bartz, Pero Subasic and Benjamin Rey. 2006b. Automatically generating related aueries in Japanese. *Language Resources and Evaluation* 40: 219-232.

Hastie, Trevor, Robert Tibshirani and Jerome Friedman. 2001. *The Elements of Statistical Learning.* Springer.

Knight, Kevin, and Jonathan Graehl. 1998. Machine transliteration. *Computational Linguistics*, 24(4): 599-612.

Komachi, Mamoru and Hisami Suzuki. 2008. Minimally supervised learning of semantic knowledge from query logs. In *Proceedings of IJCNLP*, pp.358–365.

Li, Mu, Muhua Zhu, Yang Zhang and Ming Zhou. 2006. Exploring distributional similarity based models for query spelling correction. In *Proceedings of COLING/ACL*, pp.1025-1032.

Li, Xiao, Ye-Yi Wang and Alex Acero. 2008. Learning query intent from regularized click graphs. In *Proceedings of SIGIR*.

Masuyama, Takeshi, Satoshi Sekine, and Hiroshi Nakagawa. 2004. Automatic construction of Japanese katakana variant list from large corpus. In *Proceedings COLING*, pp.1214-1219.

Okazaki, Naoaki, Mitsuru Ishizuka and Jun'ichi Tsujii. 2008a. A discriminative approach to Japanese abbreviation extraction. In *Proceedings of IJCNLP*.

Okazaki, Naoaki, Yoshimasa Tsuruoka, Sophia Ananiadou and Jun'ichi Tsujii. 2008b. A discriminative candidate generator for string transformations. In *Proceedings of EMNLP*.

Rasmussen, E. 1992. Clustering algorithm. In W.B.Frakes and R.Baeza-Yates (eds.), *Information Retrieval: Data Structure and Algorithms*, Chapter 16. Prentice Hall.

Salton, G., and C. Buckley. 1990. Improving retrieval performance by relevance feedback. *Journal of the American Society for Information Science*, 41(4): 288-297.

Sekine, Satoshi, and Hisami Suzuki. 2007. Acquiring ontological knowledge from query logs. In *Proceedings of WWW*, pp.1223-1224