Japanese Query Alteration Based on Semantic Similarity

Masato Hagiwara
Nagoya University
Furo-cho, Chikusa-ku
Nagoya 464-8603, Japan
hagiwara@kl.i.is.nagoya-u.ac.jp

Hisami Suzuki
Microsoft Research
One Microsoft Way
Redmond, WA 98052, USA
hisamis@microsoft.com

Abstract

We propose a unified approach to web search query alterations in Japanese that is not limited to particular character types or orthographic similarity between a query and its alteration candidate. Our model is based on previous work on English query correction, but makes some crucial improvements: (1) we augment the query-candidate list to include orthographically dissimilar but semantically similar pairs; and (2) we use kernel-based lexical semantic similarity to avoid the problem of data sparseness in computing query-candidate similarity. We also propose an efficient method for generating query-candidate pairs for model training and testing. We show that the proposed method achieves about 80% accuracy on the query alteration task, improving over previously proposed methods that use semantic similarity.

1 Introduction

Web search query correction is an important problem to solve for robust information retrieval given how pervasive errors are in search queries: it is said that more than 10% of web search queries contain errors (Cucerzan and Brill, 2004). English query correction has been an area of active research in recent years, building on previous work on general-purpose spelling correction. However, there has been little investigation of query correction in languages other than English.

In this paper, we address the issue of query correction, and more generally, query alteration in Japanese. Japanese poses particular challenges to the query correction task due to its complex writing system, summarized in Fig. 1[1]. There are four main character types, including two types of *kana* (phonetic alphabet - *hiragana* and *katakana*), *kanji* (ideographic - characters represent meaning) and Roman alphabet; a word can be legitimately spelled in multiple ways, combining any of these character sets. For example, the word for 'protein' can be spelled as たんぱくしつ (all in hiragana), タンパク質 (katakana+kanji), 蛋白質 (all in kanji) or たん白質 (hiragana+kanji), all pronounced in the same way (tanpakushitsu). Some examples of these spelling variants are shown in Fig. 1 with the prefix Sp: as is observed from the figure, spelling variation occurs within and across different character types. Resolving these variants will be essential not only for information retrieval but practically for all NLP tasks.

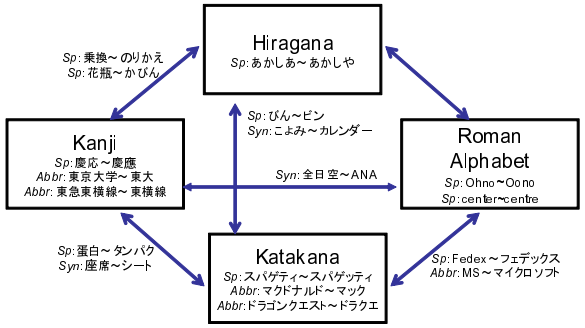

Figure 1: Japanese character types and spelling variants

A particularly prolific source of spelling variations in Japanese is katakana. Katakana characters are used to transliterate words from English and other foreign languages, and as such, the variations in the source language pronunciation as well as the ambiguity in sound adaptation are reflected in the katakana spelling. For example, Masuyama et al. (2004) report that at least six distinct transliterations of the word 'spaghetti' (スパゲッティ, スパゲティー, etc.) are attested in the newspaper corpus they studied. Normalizing katakana spelling variations has been the subject of research by itself (Aramaki et al., 2008; Masuyama et al., 2004). Similarly, English-to-katakana transliteration (e.g., 'fedex' as フェデックス fedekkusu in Fig. 1) and katakana-to-

[1]The figure is somewhat over-simplified as it does not include any word consisting of multiple character types. It also does not include examples of spelling mistakes and variants in word segmentation.

Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the ACL, pages 191–199,
Boulder, Colorado, June 2009.

English back-transliteration (e.g., フェデックス back into 'fedex') have also been studied extensively (Bilac and Tanaka, 2004; Brill et al., 2001; Knight and Graehl, 1998), as it is an essential component for machine translation. To our knowledge, however, there has been no work that addresses spelling variation in Japanese generally.

In this paper, we propose a general approach to query correction/alteration in Japanese. Our goal is to find precise re-write candidates for a query, be it a correction of a spelling error, normalization of a spelling variant, or finding a strict synonym including abbreviations (e.g., MS　マイクロソフト 'Microsoft', prefixed by Abbr in Fig. 1) and true synonyms (e.g., 座席 (translation of 'seat')　シート (transliteration of 'seat', indicated by *Syn* in Fig. 1)[2]. Our method is based on previous work on English query correction in that we use both spelling and semantic similarity between a query and its alteration candidate, but is more general in that we include alteration candidates that are not similar to the original query in spelling. In computing semantic similarity, we adopt a kernel-based method (Kandola et al., 2002), which improves the accuracy of the query alteration results over previously proposed methods. We also introduce a novel approach to creating a dataset of query and alteration candidate pairs efficiently and reliably from query session logs.

2 Related Work

The key difference between traditional general-purpose spelling correction and search query correction lies in the fact that the latter cannot rely on a lexicon: web queries are replete with valid out-of-dictionary words which are not mis-spellings of in-vocabulary words. Cucerzan and Brill (2004) pioneered the research of query spelling correction, with an excellent description of how a traditional dictionary-based speller had to be adapted to solve the realistic query correction problem. The model they proposed is a source-channel model, where the source model is a word bigram model trained on query logs, and the channel model is based on a weighted Damerau-Levenshtein edit distance. Brill and Moore (2000) proposed a general, improved source model for general spelling correction, while Ahmad and Kondrak (2005) learned a spelling error model from search query logs using the Expectation Maximization algorithm, without relying on a training set of misspelled words and their corrections.

Extending the work of Cucerzan and Brill (2004), Li et al. (2006) proposed to include semantic similarity between the query and its correction candidate. They point out that *adventura* is a common misspelling of *aventura*, not *adventure*, and this cannot be captured by a simple string edit distance, but requires some knowledge of distributional similarity. Distributional similarity is measured by the similarity of the context shared by two terms, and has been successfully applied to many natural language processing tasks, including semantic knowledge acquisition (Lin, 1998).

Though the use of distributional similarity improved the query correction results in Li et al.'s work, one problem is that it is sparse and is not available for many rarer query strings. Chen et al. (2007) addressed this problem by using external information (i.e., web search results); we take a different approach to solve the sparseness problem, namely by using semantic kernels.

Jones et al. (2006a) generated Japanese query alteration pairs from by mining query logs and built a regression model which predicts the quality of query rewriting pairs. Their model includes a wide variety of orthographical features, but not semantic similarity features.

3 Query Alteration Model

3.1 Problem Formulation

We employ a formulation of query alteration model that is similar to conventional query correction models. Given a query string q as input, a query correction model finds a correct alteration c^* within the confusion set of q, so that it maximizes the posterior probability:

$$c^* = \arg \max_{c \in \mathrm{CF}(q) \subset C} P(c|q) \quad (1)$$

where C is the set of all white-space separated words and their bigrams in query logs in our case[3], and

[2] Our goal is to harvest alternation candidates; therefore, exactly how they are used in the search task (whether it is used to *substitute* the original query, to *expand* it, or simply to *suggest* an alternative) is not a concern to us here.

[3] In regular text, Japanese uses no white spaces to separate words; however, white spaces are often (but not consistently)

$\mathrm{CF}(q) \subset C$ is the confusion set of q, consisting of the candidates within a certain edit distance from q, i.e., $\mathrm{CF}(q) = \{c \in C | \mathrm{ED}(q, c) < \theta\}$. We set $\theta = 24$ using an unnormalized edit distance. The detail of the edit distance $\mathrm{ED}(q, c)$ is described in Section 3.2. The query string q itself is contained in $\mathrm{CF}(q)$, and if the model output is different from q, it means the model suggests a query alteration. Formulated in this way, both query error detection and alteration are performed in a unified way.

After computing the posterior probability of each candidate in $\mathrm{CF}(q)$ by the source channel model (Section 3.2), an N-best list is obtained as the initial candidate set C_0, which is then augmented by the bootstrapping method *Tchai* (Section 3.4) to create the final candidate list $C(q)$. The candidates in $C(q)$ are re-ranked by a maximum entropy model (Section 3.5) and the candidate with the highest posterior probability is selected as the output.

3.2 Source Channel Model

Source channel models are widely used for spelling and query correction (Brill and Moore, 2000; Cucerzan and Brill, 2004). Instead of directly computing Eq. (1), we can decompose the posterior probability using Bayes' rule as:

$$c^* = \arg \max_{c \in \mathrm{CF}(q) \subset C} P(c)P(q|c), \tag{2}$$

where the source model $P(c)$ measures how probable the candidate c is, while the error model $P(q|c)$ measures how similar q and c are.

For the source model, an n-gram based statistical language model is the standard in previous work (Ahmad and Kondrak, 2005; Li et al., 2006). Word n-gram models are simple to create for English, which is easy to tokenize and to obtain word-based statistics, but this is not the case with Japanese. Therefore, we simply considered the whole input string as a candidate to be altered, and used the relative frequency of candidates in the query logs to build the language model:

$$P(c) = \frac{\mathrm{Freq(c)}}{\sum_{c' \in C} \mathrm{Freq(c')}}. \tag{3}$$

For the error model, we used an improved channel model described in (Brill and Moore, 2000), which we call the *alpha-beta model* in this paper. The model is a weighted extension of the normal Damerau-Levenshtein edit distance which equally penalizes single character insertion, substitution, or deletion operations (Damerau, 1964; Levenshtein, 1966), and considers generic edit operations of the form $\alpha \rightarrow \beta$, where α and β are any (possibly null) strings. From misspelled/correct word pairs, *alpha-beta* trains the probability $P(\alpha \rightarrow \beta|\mathrm{PSN})$, conditioned by the position PSN of α in a word, where PSN $\in$ {start of word, middle of word, end of word}. Under this model, the probability of rewriting a string w to a string s is calculated as:

$$P_{\alpha\beta}(s|w) = \max_{R \in \mathrm{Part}(w), T \in \mathrm{Part}(s)} \prod_{i=1}^{|R|} P(R_i \rightarrow T_i|\mathrm{PSN}(R_i)),$$

which corresponds to finding best partitions R and T in all possible partitions $\mathrm{Part}(w)$ and $\mathrm{Part}(s)$. Brill and Moore (2000) reported that this model gave a significant improvement over conventional edit distance methods.

Brill et al. (2001) applied this model for extracting katakana-English transliteration pairs from query logs. They trained the edit distance between character chunks of katakana and Roman alphabets, after converting katakana strings to Roman script. We also trained this model using 59,754 katakana-English pairs extracted from aligned Japanese and English Wikipedia article titles. In this paper we allowed $|\alpha|, |\beta| \leq 3$. The resulting edit distance is obtained as the negative logarithm of the *alpha-beta* probability, i.e., $\mathrm{ED}_{\alpha\beta}(q|c) = -\log P_{\alpha\beta}(q|c)$.

Since the edit operations are directional and c and q can be any string consisting of katakana and English, distance in both directions were considered. We also included a modified edit distance $\mathrm{ED}_{\mathrm{hd}}$ for simple kana-kana variations after converting them into Roman script. The distance $\mathrm{ED}_{\mathrm{hd}}$ is essentially the same as the normal Damerau-Levenshtein edit distance, with the modification that it does not penalize character halving ($aa \rightarrow a$) and doubling ($a \rightarrow aa$), because a large part of katakana variants only differ in halving/doubling (e.g. スパゲティ (`supageti`) vs スパゲティー (`supagetii`)[4].

The final error probability is obtained from the minimum of these three distances:

used to separate words in Japanese search queries, due to their keyword-based nature.

[4]However, character length can be distinctive in katakana, as in ビル `biru` 'building' vs. ビール `biiru` 'beer'.

$$\mathrm{ED}(q,c) = \min[\mathrm{ED}_{\alpha\beta}(q|c), \mathrm{ED}_{\alpha\beta}(c|q), \mathrm{ED}_{\mathrm{hd}}(q,c)], \quad (4)$$

$$P(q|c) = \exp[-\mathrm{ED}(q,c)] \quad (5)$$

where every edit distance is normalized to [0, 1] by multiplying by a factor of $2/(|q||c|)$ so that it does not depend on the length of the input strings[5].

3.3 Kernel-based Lexical Semantic Similarity

3.3.1 Distributional Similarity

The source channel model described in Section 3.2 only considers language and error models and cannot capture semantic similarity between the query and its correction candidate. To address this issue, we use distributional similarity (Lin, 1998) estimated from query logs as additional evidence for query alteration, following Li et al. (2006).

For English, it is relatively easy to define the context of a word based on the bag-of-words model. As this is not expected to work on Japanese, we define context as everything but the query string in a query log, as Paşca et al. (2006) and Komachi and Suzuki (2008) did for their information extraction tasks. This formulation does not involve any segmentation or boundary detection, which makes this method fast and robust. On the other hand, this may cause additional sparseness in the vector representation; we address this issue in the next two sections.

Once the context of a candidate c_i is defined as the patterns that the candidate co-occurs with, it can be represented as a vector $\mathbf{c}_i = [\mathrm{pmi}(c_i, p_1), \ldots, \mathrm{pmi}(c_i, p_M)]'$, where M denotes the number of context patterns and x' is the transposition of a vector (or possibly a matrix) x. The elements of the vector are given by pointwise mutual information between the candidate c_i and the pattern p_j, computed as:

$$\mathrm{pmi}(c_i, p_j) = \log \frac{|c_i, p_j|}{|c_i, *||*, p_j|}, \quad (6)$$

where $|c_i, p_j|$ is the frequency of the pattern p_j instantiated with the candidate c_i, and '*' denotes a wildcard, i.e., $|c_i, *| = \sum_p |c_i, p|$ and $|*, p_j| = \sum_c |c, p_j|$. With these defined, the distributional similarity can be calculated as cosine similarity. Let $\hat{\mathbf{c}}_i$ be the L2-normalized pattern vector of the candidate c_i, and $X = \{\hat{\mathbf{c}}_i\}$ be the candidate-pattern co-occurrence matrix. The candidate similarity matrix K can then be obtained as $K = X'X$. In the following, the (i, j)-element of the matrix K is denoted as K_{ij}, which corresponds to the cosine similarity between candidates c_i and c_j.

3.3.2 Semantic Kernels

Although distributional similarity serves as strong evidence for semantically relevant candidates, directly applying the technique to query logs faces the sparseness problem. Because context patterns are drawn from query logs and can also contain spelling errors, alterations, and word permutations as much as queries do, context differs so greatly in representations that even related candidates might not have sufficient contextual overlap between them. For example, a candidate "YouTube" matched against the patterns "YouTube+movie", "movie+YouTube" and "You-Tube+movii" (with a minor spelling error) will yield three distinct patterns "#+movie", "movie+#" and "#+movii"[6], which will be treated as three separate dimensions in the vector space model.

This sparseness problem can be partially addressed by considering the correlation between patterns. Kandola et al. (2002) proposed new kernel-based similarity methods which incorporate indirect similarity between terms for a text retrieval task. Although their kernels are built on a document-term co-occurrence model, they can also be applied to our candidate-pattern co-occurrence model. The proposed kernel is recursively defined as:

$$\hat{K} = \beta X'\hat{G}X + K, \quad \hat{G} = \beta X\hat{K}X' + G, \quad (7)$$

where $G = XX'$ is the correlation matrix between patterns and β is the factor to ensure that longer range effects decay exponentially. This can be interpreted as augmenting the similarity matrix K through indirect similarities of patterns $\hat{G}$ and vice versa. Semantically related pairs of patterns are expected to be given high correlation in the matrix $\hat{G}$ and this will alleviate the sparseness problem. By

[5] We did not include kanji variants here, because disambiguating kanji readings is a very difficult task, and the majority of the variations in queries are in katakana and Roman alphabet. The framework proposed in this paper, however, can incorporate kanji variants straightforwardly into $\mathrm{ED}(q, c)$ once we have reasonable edit distance functions for kanji variations.

[6] '+' denotes a white space, and '#' indicates where the word of interest is found in a context pattern.

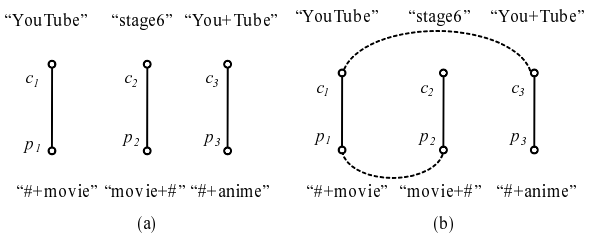

Figure 2: Orthographically Augmented Graph

solving the above recursive definition, one obtains the *von Neumann kernel*:

$$\hat{K}(\beta) = K(I - \beta K)^{-1} = \sum_{t=1}^{\infty} \beta^{t-1} K^t. \quad (8)$$

This can also be interpreted in terms of a random walk in a graph where the nodes correspond to all the candidates and the weight of an edge (i, j) is given by K_{ij}. A simple calculation shows that K_{ij} equals the sum of the products of the edge weights over all possible paths between the nodes corresponding c_i and c_j in the graph. Also, K^t_{ij} corresponds to the probability that a random walk beginning at node c_i ends up at node c_j after t steps, assuming that the entries are all positive and the sum of the connections is 1 at each node. Following this notion, Kandola et al. (2002) proposed another kernel called *exponential kernel*, with alternative faster decay factors:

$$\tilde{K}(\beta) = K \sum_{t=1}^{\infty} \frac{\beta^t K^t}{t!} = K \exp(\beta K). \quad (9)$$

They showed that this alternative kernel achieved a better performance for their text retrieval task. We employed these two kernels to compute distributional similarity for our query correction task.

3.3.3 Orthographically Augmented Kernels

Although semantic relatedness can be partially captured by the semantic kernels introduced in the previous section, they may still have difficulties computing correlations between candidates and patterns especially for only sparsely connected graphs. Take the graph (a) in Fig. 2 for example, which is a simplified yet representative graph topology for candidate-pattern co-occurrence we often encounter. In this case $K = X'X$ equals I, meaning that the connections between candidates and patterns are too sparse to obtain sufficient correlation even when semantic kernels are used.

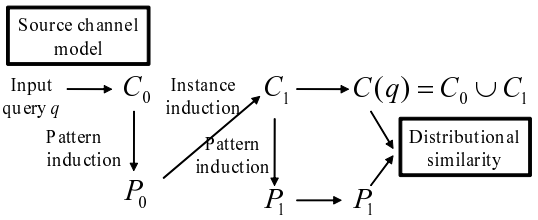

Figure 3: Bootstrapping Additional Candidates

In order to address this issue, we propose to augment the graph by weakly connecting the candidate and pattern nodes as shown in the graph (b) of Fig. 2 based on prior knowledge of orthographic similarity about candidates and patterns. This can be achieved using the following candidate similarity matrix K^+ instead of K:

$$K^+ = \gamma S_C + (1 - \gamma) X' \left[\delta S_P + (1 - \delta) I\right] X \quad (10)$$

where $S_C = \{s_c(i, j)\}$ is the orthographical similarity matrix of candidates in which the (i, j)-element is given by the edit distance based similarity, i.e., $s_c(i, j) = \exp\left[-\mathrm{ED}(c_i, c_j)\right]$. The orthographical similarity matrix of patterns $S_P = \{s_P(i, j)\}$ is defined similarly, i.e., $s_P(i, j) = \exp[-\mathrm{ED}(p_i, p_j)]$. Note that using this similarity matrix K^+ can be interpreted as a random walk process on a bipartite graph as follows. Let C and P as the sets of candidates and patterns. K^+ corresponds to a single walking step from C to C, by either remaining within C with a probability of γ or moving to "the other side" P of the graph with a probability of $1-\gamma$. When the walking remains in C, it is allowed to move to another candidate node following the candidate orthographic similarity S_C. Otherwise it moves to P by the matrix X, chooses either to move within P with a probability γS_P or to stay with a probability $1 - \gamma$, and finally comes back to C by the matrix X'. Multiplication $(K^+)^t$ corresponds to repeating this process t times. Using this similarity, we can define two orthographically augmented semantic kernels which differ in the decaying factors, augmented von Neumann kernel and exponential kernel:

$$\hat{K}^+(\beta) = K^+(I - \beta K^+)^{-1} \quad (11)$$

$$\tilde{K}^+(\beta) = K^+ \exp(\beta K^+). \quad (12)$$

3.4 Bootstrapping Additional Candidates

Now that we have a semantic model, our query correction model can cover query-candidate pairs

which are only semantically related. However, previous work on query correction all used a string distance function and a threshold to restrict the space of potential candidates, allowing only the orthographically similar candidates.

To collect additional candidates, the use of context-based semantic extraction methods would be effective because semantically related candidates are likely to share context with the initial query q, or at least with the initial candidate set C_0. Here we used the *Tchai* algorithm (Komachi and Suzuki, 2008), a modified version of *Espresso* (Pantel and Pennacchiotti, 2006) to collect such candidates. This algorithm starts with initial seed instances, then induces reliable context patterns cooccurring with the seeds, induces instances from the patterns, and iterates this process to obtain categories of semantically related words. Using the candidates in C_0 as the seed instances, one bootstrapping iteration of the *Tchai* algorithm is executed to obtain the semantically related set of instances C_1. The seed instance reliabilities are given by the source channel probabilities $P(c)P(q|c)$. Finally we take the union $C_0 \cup C_1$ to obtain the candidate set $C(q)$. This process is outlined in Fig. 3.

3.5 Maximum Entropy Model

In order to build a unified probabilistic query alteration model, we used the maximum entropy approach of (Beger et al., 1996), which Li et al. (2006) also employed for their query correction task and showed its effectiveness. It defines a conditional probabilistic distribution $P(c|q)$ based on a set of feature functions $f_1, \ldots, f_K$:

$$P(c|q) = \frac{\exp \sum_{i=1}^{K} \lambda_i f_i(c, q)}{\sum_c \exp \sum_{i=1}^{K} \lambda_i f_i(c, q)}, \qquad (13)$$

where $\lambda_1, \ldots, \lambda_K$ are the feature weights. The optimal set of feature weights λ^* can be computed by maximizing the log-likelihood of the training set.

We used the Generalized Iterative Scaling (GIS) algorithm (Darroch and Ratcliff, 1972) to optimize the feature weights. GIS trains conditional probability in Eq. (13), which requires the normalization over all possible candidates. However, the number of all possible candidates C obtained from a query log can be very large, so we only calculated the sum over the candidates in $C(q)$. This is the same approach that Och and Ney (2002) took for statistical machine translation, and Li et al. (2006) for query spelling correction.

We used the following four categories of functions as the features:

1. *Language model feature*, given by the logarithm of the source model probability: $\log P(c)$.
2. *Error model features*, which are composed of three edit distance functions: $-\mathrm{ED}_{\alpha\beta}(q|c)$, $-\mathrm{ED}_{\alpha\beta}(c|q)$, and $-\mathrm{ED}_{\mathrm{hd}}(q, c)$.
3. *Similarity based feature*, computed as the logarithm of distributional similarity between q and c: $\log \mathrm{sim}(q, c)$, which is calcualted using one of the following kernels (Section 3.3): $K, \hat{K}, \tilde{K}, \hat{K}^+$, and $\tilde{K}^+$. The similarity values were normalized to [0, 1] after adding a small discounting factor $\varepsilon = 1.0 \times 10^{-5}$.
4. *Similarity based correction candidate features*, which are binary features with a value of 1 if and only if the frequency of c is higher than that of q, and distributional similarity between them is higher than a certain threshold. Li et al. (2006) used this set of features, and suggested that these features give the evidence that q may be a common misspelling of c. The thresholds on the normalized distributional similarity are enumerated from 0.5 to 0.9 with the interval 0.1.

4 Experiment

4.1 Dataset Creation

For all the experiments conducted in this paper, we used a subset of the Japanese search query logs submitted to Live Search (www.live.com) in November and December of 2007. Queries submitted less than eight times were deleted. The query log we used contained 83,080,257 tokens and 1,038,499 unique queries.

Models of query correction in previous work were trained and evaluated using manually created query-candidate pairs. That is, human annotators were given a set of queries and were asked to provide a correction for each query when it needed to be rewritten. As Cucerzan and Brill (2004) point out, however, this method is seriously flawed in that the intention of the original query is completely lost to the annotator, without which the correction is often impossible: it is not clear if *gogle* should be corrected to *google* or *goggle*, or neither — *gogle* may be a brand new product name. Cucerzan and Brill

therefore performed a second evaluation, where the test data was drawn by sampling the query logs for successive queries (q_1, q_2) by the same user where the edit distance between q_1 and q_2 are within a certain threshold, which are then submitted to annotators for generating the correction. While this method makes the annotation more reliable by relying on user (rather than annotator) reformulation, the task is still overly difficult: going back to the example in Section 1, it is unclear which spelling of 'protein' produces the best search results — it can only be empirically determined. Their method also eliminates all pairs of candidates that are not orthographically similar. We have therefore improved their method in the following manner, making the process more automated and thus more reliable.

We first collected a subset of the query log that contains only those pairs (q_1, q_2) that are issued successively by the same user, q_2 is issued within 3 minutes of q_1, and q_2 resulted in a click of the resulting page while q_1 did not. The last condition adds the evidence that q_2 was a better formulation than q_1. We then ranked the collected query pairs using log-likelihood ratio (LLR) (Dunning, 1993), which measures the dependence between q_1 and q_2 within the context of web queries (Jones et al., 2006b). We randomly sampled 10,000 query pairs with LLR ≥ 200, and submitted them to annotators, who only confirm or reject a query pair as being synonymous. For example, $q_1 =$ *nikon* and $q_2 =$ *canon* are related but not synonymous, while we are reasonably sure $q_1 =$ *ipot* and $q_2 =$ *ipod* are synonymous, given that this pair has a high LLR value. This verification process is extremely fast and consistent across annotators: it takes less than 1 hour to go through 1,000 query pairs, and the inter-annotator agreement rate of two annotators on 2,000 query pairs was 95.7%. We annotated 10,000 query pairs consisting of alpha-numerical and kana characters in this manner. After rejecting non-synonymous pairs and those which do not co-occur with any context patterns, 6,489 pairs remained, and we used 1,243 pairs for testing, 628 as a development set, and 4,618 for training the maximum entropy model.

4.2 Experimental Settings

The performance of query alteration was evaluated based on the following measures (Li et al., 2006). The input queries, correct suggestions, and outputs were matched in a case-insensitive manner.

- *Accuracy*: The number of correct outputs generated by the system divided by the total number of queries in the test set;
- *Recall*: The number of correct suggestions for altered queries divided by the total number of altered queries in the test set;
- *Precision*: The number of correct suggestions for altered queries divided by the total number of alterations made by the system.

The parameters for the kernels, namely, β, γ, and δ, are tuned using the development set. The finally employed values are: $\beta = 0.3$ for $\hat{K}$, $\tilde{K}$, and $\hat{K}^+$, $\beta = 0.2$ for $\tilde{K}^+$, $\gamma = 0.2$ and $\delta = 0.4$ for $\hat{K}^+$, and $\gamma = 0.35$ and $\delta = 0.7$ for $\tilde{K}^+$. In the source channel model, we manually scaled the language probability by a factor of 0.1 to alleviate the bias toward highly frequent candidates.

As the initial candidate set C_0, top-50 instances were selected by the source channel model, and 100 patterns were extracted as P_0 by the *Tchai* iteration after removing generic patterns, which we detected simply by rejecting those which induced more than 200 unique instances. Finally top-30 instances were induced using P_0 to create C_1. Generic instances were not removed in this process because they may still be alterations of input query q. The maximum size of P_1 was set to 2,000, after removing unreliable patterns with reliability smaller than 0.0001.

Table 1: Performance results (%)

Model	Accuracy	Recall	Precision
SC	71.12	39.29	45.09
ME-NoSim	74.58	44.58	52.52
ME-Cos	74.18	**45.84**	50.70
ME-vN	74.34	45.59	52.16
ME-Exp	73.61	44.84	50.57
ME-vN+	75.06	44.33	53.01
ME-Exp+	**75.14**	44.08	**53.52**

4.3 Results

Table 1 shows the evaluation results. SC is the source channel model, while the others are maximum entropy (ME) models with different features. ME-NoSim uses the same features as SC, but considerably outperforms SC in all three measures, confirming the superiority of the ME approach. Decomposing the three edit distance functions into three

separate features in the ME model may also explain the better result. All the ME approaches outperformed SC in accuracy with a statistically significant difference ($p < 0.0001$ on McNemar's test).

The model with the cosine similarity (ME-Cos) in addition to the basic set of features yielded higher recall compared to ME-NoSim, but decreased accuracy and precision, which are more important than recall for our purposes because a false alteration does more damage than no alteration. This is also the case when the kernel-based methods, ME-vN (the von Neumann kernel) and ME-Exp (the exponential kernel), are used in place of the cosine similarity. This shows that using semantic similarity does not always help, which we believe is due to the sparseness of the contextual information used in computing semantic similarity.

On the other hand, ME-vN+ (with augmented von Neumann kernel) and ME-Exp+ (with augmented exponential kernel) increased both accuracy and precision with a slight decrease of recall, compared to the distributional similarity baseline and the non-augmented kernel-based methods. ME-Exp+ was significantly better than ME-Exp ($p < 0.01$).

Note that the accuracy values appear lower than some of the previous results on English (e.g., more than 80% in (Li et al., 2006)), but this is because the dataset creation method we employed tends to over-represent the pairs that lead to alteration: the simplest baseline (= always propose no alteration) performs 67.3% accuracy on our data, in contrast to 83.4% on the data used in (Li et al., 2006).

Manually examining the suggestions made by the system also confirms the effectiveness of our model. For example, the similarity-based models altered the query *ipot* to *ipod*, while the simple ME-NoSim model failed, because it depends too much on the edit distance-based features. We also observed that many of the suggestions made by the system were actually reasonable, even though they were different from the annotated gold standard. For example, ME-vN+ suggests a re-write of the query *2tyann* as 2 ちゃんねる ('2-channel'), while the gold standard was an abbreviated form 2 ちゃん ('2-chan').

To incorporate such possibly correct candidates into account, we conducted a follow-up experiment where we considered multiple reference alterations, created automatically from our data set in the following manner. Suppose that a query q_1 is corrected as q_2, and q_2 is corrected as q_3 in our annotated data. If this is the case, we considered $q_1 \rightarrow q_3$ as a valid alteration as well. By applying this chaining operation up to 5 degrees of separation, we re-created a set of valid alterations for each input query. Note that directionality is important — in the above example, $q_1 \rightarrow q_3$ is valid, while $q_3 \rightarrow q_1$ is not. Table 2 shows the results of evaluation with multiple references. The numbers substantially improved over the single reference cases, as expected, but did not affect the relative performance of each model. Again, the differences in accuracy between the SC and ME models, and ME-Exp and ME-Exp+ were statistically significant ($p < 0.01$).

Table 2: Performance with the multiple reference model

Model	Accuracy	Recall	Precision
SC	75.30	48.61	55.78
ME-NoSim	79.49	56.17	66.17
ME-Cos	79.32	**58.19**	64.35
ME-vN	79.24	57.18	65.42
ME-Exp	78.52	56.42	63.64
ME-vN+	**79.89**	55.67	66.57
ME-Exp+	79.81	54.91	**66.67**

5 Conclusion and future work

In this paper we have presented a unified approach to Japanese query alteration. Our approach draws on previous research in English spelling and query correction, Japanese katakana variation and transliteration, and semantic similarity, and builds a model that makes improvements over previously proposed query correction methods. In particular, the use of orthographically augmented semantic kernels proposed in this paper is general and applicable to other languages, including English, for query alteration, especially when the data sparseness is an issue. In the future, we also plan to investigate other methods, such as PLSI (Hofmann, 1999), to deal with data sparseness in computing semantic similarity.

Acknowledgments

This research was conducted during the first author's internship at Micorosoft Research. We thank the colleagues, especially Dmitriy Belenko, Chris Brockett, Jianfeng Gao, Christian König, and Chris Quirk for their help in conducting this research.

References

Farooq Ahmad and Grzegorz Kondrak. 2005. Learning a spelling error model from search query logs. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP-2005)*, pages 955–962.

Eiji Aramaki, Takeshi Imai, Kengo Miyo, and Kazuhiko Ohe. 2008. Orthographic disambiguation incorporating transliterated probability. In *Proceedings in the third International Joint Conference on Natural Language Processing (IJCNLP-2008)*, pages 48–55.

Adam L. Beger, Stephen A. Della Pietra, and Vincent J. Della Pietra. 1996. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–72.

Slaven Bilac and Hozumi Tanaka. 2004. A hybrid back-transliteration system for japanese. In *Proceedings of the 20th international conference on Computational Linguistics (COLING-2004)*, pages 597–603.

Eric Brill and Robert C. Moore. 2000. An improved error model for noisy channel spelling. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics (ACL-2000)*, pages 286–293.

Eric Brill, 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)*, pages 393–399.

Qing Chen, Mu Li, , and Ming Zhou. 2007. Improving query spelling correction using web search results. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 181–189.

Silviu Cucerzan and Eric Brill. 2004. Spelling correction as an iterative process that exploits the collective knowledge of web users. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP-2004)*, pages 293–300.

Fred Damerau. 1964. A technique for computer detection and correction of spelling errors. *Communication of the ACM*, 7(3):659–664.

J.N. Darroch and D. Ratcliff. 1972. Generalized iterative scaling for log-linear models. *Annuals of Mathematical Statistics*, 43:1470–1480.

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

Thomas Hofmann. 1999. Probabilistic latent semantic indexing. In *Research and Development in Information Retrieval*, pages 50–57.

Rosie Jones, Kevin Bartz, Pero Subasic, and Benjamin Rey. 2006a. Automatically generating related queries in japanese. *Language Resources and Evaluation (LRE)*, 40(3-4):219–232.

Rosie Jones, Benjamin Rey, Omid Madani, and Wiley Greiner. 2006b. Generating query substitutions. In *Proceedings of the 15th international World Wide Web conference (WWW-06)*, pages 387–396.

Jaz Kandola, John Shawe-Taylor, and Nello Cristianini. 2002. Learning semantic similarity. In *Neural Information Processing Systems (NIPS 15)*, pages 657–664.

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

Mamoru Komachi and Hisami Suzuki. 2008. Minimally supervised learning of semantic knowledge from query logs. In *Proceedings of the 3rd International Joint Conference on Natural Language Processing (IJCNLP-2008)*, pages 358–365.

Vladimir I. Levenshtein. 1966. Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physice - Doklady*, 10:707–710.

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

Dekang Lin. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of COLING/ACL-1998*, pages 786–774.

Takeshi Masuyama, Satoshi Sekine, and Hiroshi Nakagawa. 2004. Automatic construction of japanese katakana variant list from large corpus. In *Proceedings of Proceedings of the 20th international conference on Computational Linguistics (COLING-2004)*, pages 1214–1219.

Franz Och and Hermann Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *Proceedings of the 40th annual meeting of ACL*, pages 295–302.

Marius Paşca, Dekang Lin, Jeffrey Bigham, Andrei Lifchits, and Alpa Jain. 2006. Organizing and searching the world wide web of facts - step one: the one-million fact extraction challenge. In *Proceedings of the 21st National Conference on Artificial Intelligence (AAAI-06)*, pages 1400–1405.

Patrick Pantel and Marco Pennacchiotti. 2006. Espresso: Leveraging generic patterns for automatically harvesting semantic relations. In *Proceedings of COLING/ACL-2006*, pages 113–120.