Automatically Harvesting *Katakana*-English Term Pairs from Search Engine Query Logs

Eric Brill, Gary Kacmarcik, Chris Brockett

Microsoft Research
One Microsoft Way, Redmond, WA 98052, USA
{brill,garykac,chrisbkt}@microsoft.com

Abstract

This paper describes a method of extracting *katakana* words and phrases, along with their English counterparts from non-aligned monolingual web search engine query logs. The method employs a trainable edit distance function to find <*katakana*, English> pairs that have a high probability of being equivalent. These pairs can then be used to further bootstrap training of the edit distance function, resulting in improved back-transliteration from *katakana* to English. In addition, this is an effective method for mining large numbers of *katakana* strings to enhance a bilingual lexicon. The improved edit distance function and enhanced lexicon can be used for more accurate alignment of bitexts, and for application during runtime MT and multilingual IR.

1 Introduction

Katakana script is conventionally used to represent foreign loan words that have been imported into Japanese. Words written in this script – proper names in particular – constitute the single biggest source of Out-Of-Vocabulary (OOV) words in modern Japanese, and as such pose a notorious headache for human translators and a stumbling block for quality machine translation (MT) and multilingual information retrieval (IR). While unmodified transfer of a proper name is appropriate in MT among languages that use Latin script, monolingual users of a Japanese-to-English MT system are unlikely to find a *katakana* string acceptable in English output: a form such as ブリトニー・スピアーズ must be identifiably reconstructed as the name *Britney Spears*, a process that Knight and Graehl (1998) have called "back-transliteration." Likewise in multilingual IR: search engines must deal with the flood of OOV words generated by global popular culture. Since the data is far larger than can be effectively managed through human intervention, multilingual IR needs tools by which OOV cognates can be mined automatically, even in the absence of aligned bitexts.

The reader is referred to Knight & Graehl (1998) for detailed discussion of the issues posed by reverse transliteration from Japanese into English, and a partial solution based on phonological modeling of English words using the online CMU pronunciation dictionary. The problems of back-transliteration are not limited to Japanese, but are relevant to any language not written in Latin script. Stalls & Knight (1998) extend the phonological modeling solution to Arabic technical terms. An alternative tack is taken by Jeong, et al. (2000), who propose a minimal edit distance model to handle back-transliteration from Korean to English.

In this paper, we exploit a novel web-based resource to attack the back-transliteration problem. We present a method for automatically mining *katakana* strings along with their English counterparts. In addition to enhancing a translation dictionary, this data can be used to train a model to automatically back-transliterate *katakana* into English. For corpora, we utilize non-bitext data from search engine queries, to address the problem of back transliteration and the fact that *katakana* words are likely to be OOV in both Japanese and English.

Below we first describe the noisy channel error model used for data harvesting. Then we present the results of experiments using monolingual databases of web queries in English and Japanese. These experiments suggest that this data can be usefully mined, both to construct

bilingual lexicons for IR and MT and to improve the model in order to bootstrap further harvesting of transliterated data.

2 The Noisy Channel Error Model

When a Japanese-to-English MT system encounters a *katakana* phrase that appears in its translation dictionary, it can find the translation by table look-up, perhaps enhanced with contextual information in cases of ambiguity. However, since *katakana* use is highly productive, no matter how large our translation dictionary is, we will always encounter a great number of *katakana* not in the dictionary. To address this problem, we use a trainable back-transliteration model for *katakana* not appearing in our dictionary.

The back-transliteration model employed in this paper is a version of the noisy-channel error model explored by Brill & Moore (2000) in the context of English spelling correction. One could just as easily utilize any other model of back-transliteration, for example, that proposed by Knight & Graehl. The Brill & Moore model extends Damerau-Levenshtein edit distance (Damerau, 1964; Levenshtein, 1966) to allow all edit operations of the form **α → β,** where α and β are any (possibly null) strings and P(α → β) is the probability that the noisy channel outputs β where α was emitted by the source. For example, for English spelling correction, the misspelling of the word "enough" as the string "enuff" could be accounted for by a single error of the form ough→uff.

In the case of *katakana*-English back-transliteration, we adopt the fiction that the Japanese writer intended to type an English word, but that the word passed through a noisy channel and the original form of the word must now be recovered via the model. The model learns a set of string-to-string mapping parameters and probabilities. For any *katakana* word or phrase we wish to translate into English, then, the task is to find:

$$\mathbf{argmax}_{\text{English}}\ P(\text{English} \mid \textit{Katakana})$$

To train the α → β edit probabilities, we first align all of our <source, target> training samples using standard unweighted Levenshtein distance, augmented with doubling (a→aa) and halving (aa→a). For instance, given the training pair <コンテキスト, context>, this can be aligned, after initial conversion of the *katakana* form to Latin script[1], as:

```
k o n t e  k i s u t o
| | | | |  |     /
c o n t e  x    t
```

The alignment here consists of the atomic edit operations Substitute/**k**/**c** Match/**o**/**o** Match/**n**/**n** Match/**t**/**t** Match/**e**/**e** Substitute/**k**/**x** Insert/**i** Insert/**s** Insert/**u** Match/**t**/**t** Insert/**o**. Atomic edits are then conflated into larger edit strings, with each string being given a fractional count. If we allow |α|,|β| <= 4, the above alignment results in the following English → romanized-*katakana* non-match edits:

c → *k*
co → *ko*
con → *kon*
cont → *kont*
tex → *tek*
tex → *teki*
ex → *ek*
ex → *eki*
ex → *ekis*
x → *k*
x → *ki*
x → *kis*
x → *kisu*
NULL → *i*
NULL → *is*
NULL → *isu*
t → *isut*
t → *suto*
t → *uto*
t → *to*
NULL → *o*

The basic idea behind this training procedure is that real edits will accumulate partial counts over many training instances, whereas other edits will not. For English spelling correction, Brill and Moore (2000) showed that this model gives

[1] To obviate the need to modify code in existing tools, *katakana* words were transliterated into Latin script using a variant of the Hepburn romanization scheme that guaranteed that complex *katakana* sequences were uniquely mapped to Latin-script sequences for subsequent recovery; however, nothing in principle precludes training directly off *katakana* strings themselves.

significantly higher accuracy than standard weighted Damerau-Levenshtein distance.

Below we show some high probability edits learned to map English to romanized *katakana*:

mble -> *nburu* (ンブル)
foot -> *futto* (フット)
-up -> *appu* (アップ)
eezer -> *iizaa* ((-i)イザア)
phen -> *fen* (フェン)
eport -> *epoot* ((-e)ポー(t-))

Because of the romanization of the *katakana* string before processing, some of these mappings match partial *katakana* characters. For example, the final "t" in the eport -> epoot mapping does not correspond to a single *katakana* character but rather to any of タ(ta), ツ(tsu), テ(te) or ト(to). This allows this mapping to handle both "report" (レポート = repooto) and "reporting" (レポーティング = repooting).

Once the model is trained, we apply it taking a *katakana* string as input. The string is romanized and then we use the error model to find the English string most likely to have been the source of the *katakana* input. These experiments are described later in the paper.

3 Harvesting Training Data

Obtaining <*katakana*, English> string pairs is useful both to enhance a bilingual lexicon and as training data for the noisy channel error model described in the previous section. With the vast amount of on-line text currently available, we set out to determine whether we can utilize this data to extract such string pairs.

We first considered harvesting data from parallel on-line encyclopedias. This idea had a number of problems. First, the number of unique *katakana* strings in the encyclopedia is relatively small and is not growing over time. Second, encyclopedias are relatively static, and do not contain new names and phrases that are constantly being introduced into the population. Additionally, stylistic constraints are imposed in an encyclopedia, resulting in a set of *katakana* with much less variation than one encounters in colloquial writing. We are also faced with the problem of finding potential English candidates in the text. If we limit the possibilities to single words, this is not particularly difficult. However, if we allow proper names and phrases this becomes difficult.

We also considered using the web as a means for mining <*katakana*, English> pairs. Obviously, the web is vast and topical, therefore being great potential resource for finding data. If we wish to avoid the need for crawling the web to collect a large corpus, we could use a search engine as a key into the data. We explored different methods for doing this. While this does appear to hold promise as a means for mining these string pairs, we found a resource that proved much more effective.

3.1 Query Logs

Our next idea was to use search engine query logs for mining <*katakana*, English> pairs. We extracted a database of English and Japanese queries from the MSN Search query logs and discovered that the Japanese query logs have a very high concentration of *katakana*.

Whereas an electronic encyclopedia with 461,567 sentences yielded only 60,127 unique *katakana* strings, we were able to process nearly 100,000,000 queries to extract over 750,000 unique *katakana* strings from query logs over a 1 month period. Even at the end of this 1 month period we were still acquiring 10,000 new *katakana* strings each day. See the graph below in Figure 1[2].

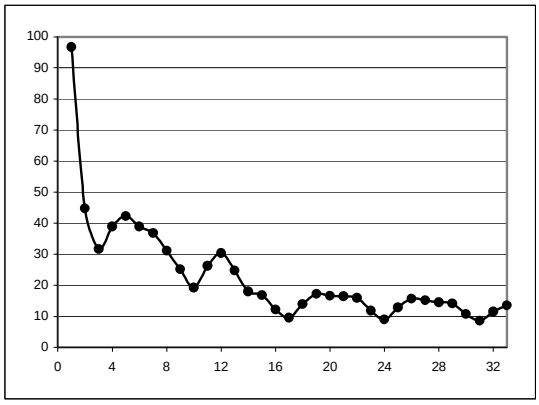

Figure 1:
# of New *Katakana* Strings (in 1000s) vs. # of Days Spent Collecting

[2] The minimum that occurs every 7 days corresponds to Saturday.

Given the nature of on-line search, query logs contain topical terms, and in particular just those terms that a static translation lexicon is unlikely to cover. In addition, when people type queries they typically type noun phrases. As such, we do not have to worry about any linguistic processing of the string. We can treat every query as an atomic string and learn mappings between *katakana* queries found in the Japanese query logs with English queries found in the English query logs.

Query logs seem to be a particularly good source for <*katakana*, English> pairs given:

(1) the high concentration of *katakana* in Japanese query logs (~60% of the queries contain *katakana* strings),

(2) the fact that many of the *katakana* terms are likely to be topical words or phrases that will also occur in English query logs,

(3) query logs are a vast, always growing and highly topical resource.

Many of these *katakana* strings harvested will represent typos or other user errors, so we cleaned up the data by filtering out those entries that occurred less than a given threshold number of times. Using a threshold of 50, we still end up with over 60,000 unique *katakana* strings during the one-month harvesting period.

Below we present experiments we ran to harvest <*katakana*, English> string pairs from query logs and to use these pairs to improve automatic back-transliteration.

3.2 Harvesting from Non-Aligned Query Databases

Given a *katakana* string from the Japanese query logs, we attempt to find an English string in the English query logs that is likely to be an equivalent string. This is done as follows. We begin with a seed training set of 1000 <*katakana*, English> pairs that had been hand-checked for accuracy, and a dictionary consisting of queries found in the English query logs, in addition to 112,000 English words[3]. We trained our noisy channel error model using these pairs. We then iteratively did the following:

EXTRACTION ALGORITHM

(1) For a set of N *katakana* strings

  (a) For each k ε N

    (i) For each string d in the English Dictionary compute Distance(k,d)

    (ii) If we find an English string d' that is sufficiently close to k, then output <k, d'> as a matching pair.

(2) Place all matching pairs into an auxiliary training set file.

(3) Retrain noisy channel error model using the original base pairs combined with the newly created training set file.

4 Growing a Bilingual Lexicon

We first tested how well the algorithm presented in the last section performed at mining novel <*katakana*, English> string pairs for a bilingual lexicon. A bilingual lexicon created in this fashion promises to be very dynamic and topical resource, with a large number of new *katakana* strings appearing every day.

We set the various thresholds of the model to give us a 10% yield (in other words, we reject 90% of the *katakana* strings encountered). We then hand checked a randomly chosen sample of 1500 unique <*katakana*, English> pairs that we automatically extracted from the query logs. Of these, 97.5% were correct matches.

Of the errors in the mined pairs, many of them were reasonable suggestions: "shingle" (vs. the correct "single") for シングル (= *shinguru*), "fiver" (vs. "fiber") for ファイバー (= *faibaa*), and "packman" (vs. "pac-man") for パックマン (= *pakkuman*). Some other errors exposed *faux-amis* between English and Japanese, like: マイム (= *maimu*) returning "maim" instead of "mime", ライニング (= *rainingu*) returning

[3] In practice, the English lexicon would also expand each day as new query data is added. In our experiments, we held the English lexicon constant so that we could more reasonably compare the model from one iteration to the next.

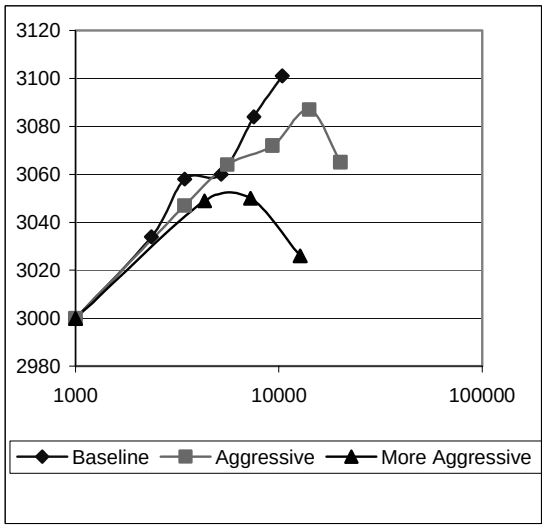

Figure 2:
 # Correct vs. # of Training Pairs
for different levels of *katakana* pair harvesting

"raining" instead of "lining", and プリズム (= *purizumu*) matching "purism" instead of the correct "prism". The final category of errors encountered were the basic modeling errors that passed through our filter: "past" (vs. "post") for ポスト (= *posuto*), "retrace" (vs. "lettuce") for レタス (= *retasu*), and "kangaroo" (vs. "bungalow") for バンガロー (= *bangaroo*).

5 Improving the Noisy Channel Error Model

We next explored whether training on these mined pairs improved the accuracy of our noisy channel error model for back-transliteration of *katakana* to English, compared to our baseline system trained on the initial clean 1000 <*katakana*, English> pairs.

For our test data set, we extracted *Katakana*-English word pairs culled from several sources: the *Kenkyusha New College Japanese-English Dictionary*, terms extracted from in house localization databases, and word pairs extracted from etymological information contained in the *Iwanami Kokugojiten* pocket dictionary. These data consist of a rather general collection of technical and non-technical terms and proper names, mostly geographical names. The lists were hand-vetted by an independent contractor to ensure that the Japanese forms constituted valid transliterations (not translations) of the English, that there was no morphological variation that might be a source of noise (e.g., "Icelandic," as opposed to "Iceland"). English forms were normalized to lower case, and multiple word tokens in both languages had white spaces replaced with underscores. Our test set consists of 4500 word pairs.

For testing, we used the English lexicon described earlier. To ensure that the "truth" was always available in the dictionary, any English words in the test set that were not the reference lexicon were added to the lexicon.

In Figure 2, we show the accuracy of our error model on the test set as a function of the number of automatically mined <*katakana*, English> training pairs. In this graph, three plots are given: the baseline plot uses a fairly conservative filter to determine which pairs are used for the next training iteration whereas the other 2 use progressively more aggressive filters (which allow more "noisy" data through).

While the model initially seems to be able to handle the noisy data, at some point the feedback from the noise overwhelms the model and results in a significant decrease in model accuracy.

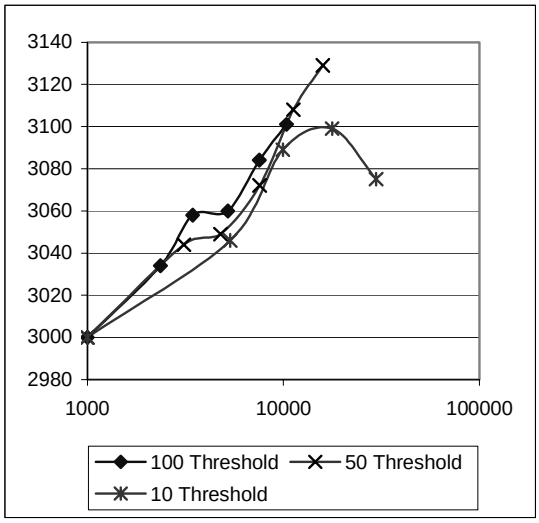

Figure 3:
 # Correct vs. # of Training Pairs
for different *katakana* lexicon thresholds

We found similar results when we modified the threshold frequency for inclusion in our

katakana lexicon. Our baseline model required that the *katakana* string occur at least 100 times in the query data. When we lowered this threshold to 50 we got better results from the additional data, but lowering it further to 10 allowed too much noise into the system, which eventually caused the model performance to degrade.

6 Residual Issues

The treatment of *katakana* in this paper has been deliberately naïve. Not all *katakana* terms are in fact transliterations of English words: examination of a random sample of 2500 *katakana* queries suggests that as many as 13.1% of the terms may not have transliteration equivalents. Native names of animals and plants are commonly written in *katakana* especially in scientific contexts (ネコ *neko* "cat"); these accounted for approximately 9.8% of terms without transliteration equivalents this sample. Likewise, *katakana* may be used for emphasis (ダメ *dame* "no good"), or to indicate special or technical uses of native or Sino-Japanese words (e.g., イジメ *ijime* "bullying" in educational contexts). Approximately 7.7% of the terms from the group of untransliteratable items were identified as falling into this latter class.

Additionally, many *katakana* words, although originally loan words, are not transliterations from English (ペンチ *penchi* "pliers"; イギリス *igirisu* "England"). Further classes of problematic word types are presented by abbreviations (パソコン *pasokon* "personal computer") and truncations (コンビニ *konbini* "convenience store"). Noise may be further introduced into the model by *katakana* words that may incorrectly match with valid English forms, sometimes with inappropriate results: エッチ *etchi* "libidinous", for example, may falsely match with "etch", in contrast with エッチング *etchingu* which correctly matches with "etching". Instances such as these generally need to be identified and lexicalized or normalized to other scripts in order to reduce the potential for noise. However, inasmuch as the presence of false positive matches in the harvested training data does not impede performance of the model, we believe that degradation is likely to be minimal.

7 Conclusions

First, given an appropriate learning model, large non-aligned data sets such as monolingual queries can be a viable source for learning bilingual OOV terms in cases where borrowing involves multiple scripts. We have seen that use of queries has the potential of permitting the harvesting of as many as 10,000 new terms a day, even after a month of data collection, with little human intervention required. Over time, it is expected that filtering for typographical errors and other noise reduction techniques might permit the extraction of additional data at lower thresholds.

Second, the noisy-channel error model that provides the alignments of non-aligned query data appears to be of robust utility in acquiring <*katakana*, English> pairs, thereby permitting rapid development of large-scale bilingual cognate word lexicons for IR and MT. Since the generic error model employed in our experiments does not rely on any intrinsic aspects of Japanese orthography, it is portable to other languages that use non-Latin scripts. The model, moreover, is robust against (although not completely immune to) noise introduced by sporadic mining of incorrect matches.

8 Future Work

Reviewing cases where the model failed to come up with a reasonable suggestion revealed that additional support for compounds could have a significant impact. For example, アスペンホテル (= *asupenhoteru* "Aspen Hotel") is not found because the English form is not in our lexicon. In this case, we recognize both "aspen" and "hotel" individually, so compounding support would improve the performance of the system.

To address the situation where a single *katakana* string can reasonably match more than one English phrase, we also plan to investigate applying contextual information to help disambiguate similar sounding alternatives. We plan to investigate unigram probabilities derived from query logs and n-gram language models to assist in this disambiguation process.

Brill and Moore (2000) also mention experiments they ran in using their error model

for English spelling correction where they iterate the training, in a method akin to the E-M algorithm. They state that they did not see any accuracy improvements in doing so. Given that the alignment between romanized *katakana* and English is much noisier than between misspelled and correctly spelled English, we believe that for our problem accuracy would benefit from retraining. The noisy alignments will lead to many more spurious edit types, a problem that could be cured by iterative training. We plan to investigate this to see how it affects our model, in addition to exploring the efficacy of other models.

Acknowledgements

We are grateful to a number of people who assisted us in this project. We would like to thank Bill Bliss, Raman Chandrasekar and the entire MSN Search team for assembling the query data and making it available to us, Robert Rounthwaite for his help in providing an English lexicon, and Hisami Suzuki for her comments on drafts of this paper. Yumi Takeuchi, Kazuko Robertshaw, Monica Corston-Oliver, and Jeff Stevenson helped with checking the data used in seed and test data sets. We remain solely responsible for the content of this paper.

References

Brill, Eric, and Robert. C. Moore. 2000. *An improved error model for noisy channel spelling correction*. ACL2000, 286-293.

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

Fujii, Atsushi, and Tetsuya Ishikawa. 1999. Cross-Language Information Retrieval for Technical Documents. *Proceedings of the Joint ACL SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*. 29-37.

Jeong, Kil Soon, Sung Hyun Myaeng, Jae Sung Lee and Key-Sun Choi. 1999. Automatic Identification and Back-Transliteration of Foreign Words for Information Retrieval. *Information Processing and Management*. **35**(4) 523-540.

Kang, Byung-Ju and Key-Sun Chol. 2000. Automatic Transliteration and Back-Transliteration by Decision-Tree Learning. *Proceedings of the 2nd International Conference on Language Resources and Evaluation (LREC 2000)*. 1135-1141.

Knight, Kevin, and Jonathan Graehl. 1998. Machine Transliteration. *Computational Linguistic*. **24**(4) 599-612.

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

Stalls, Bonnie Glover, and Kevin Knight. 1998. Translating Names and Technical Terms in Arabic Text. *Proceedings of the COLING/ACL Workshop on Computational Approaches to Semitic Languages.*