# The Transliteration from Alphabet Queries to Japanese Product Names

**Rieko Tsuji[a], Yoshinori Nemoto[a], Wimvipa Luangpiensamut[a], Yuji Abe[a], Takeshi Kimura[a], Kanako Komiya[a], Koji Fujimoto[b], Yoshiyuki Kotani[a]**

[a]Department of Computer and Information Science, Tokyo University of Agriculture and Technology / 2-24-16 Nakamachi Koganei-shi Tokyo JAPAN

[b]Tensor Consulting/ 2-10-1 Koujimachi Chiyoda-ku Tokyo JAPAN

{Riekon.m, wimvipa, kittykimura}@gmail.com,

50012646127@st.tuat.ac.jp,wizdomowl@yahoo.co.jp,

koji.fujimoto@tensor.co.jp, {kkomiya, kotani}@cc.tuat.ac.jp

## Abstract

There are some cases where the non-Japanese buyers are unable to find products they want through the Japanese shopping Web sites because they require Japanese queries. We propose to transliterate the inputs of the non-Japanese user, i.e., search queries written in English alphabets, into Japanese Katakana to solve this problem. In this research, the pairs of the non-Japanese search query which failed to get the right match obtained from a Japanese shopping website and its transcribed word given by volunteers were used for the training data. Since this corpus includes some noise for transliteration such as the free translation, we used two different filters to filter out the query pairs that are not transliterlated in order to improve the quality of the training data. In addition, we compared three methods, BIGRAM, HMM, and CRF, using these data to investigate which is the best for the query transliteration. The experiment revealed that the HMM was the best.

## 1 Introduction

In recent years, e-commerce is widely used throughout the world and it enables people to purchase products from foreign countries. However, sometimes it is not easy for foreign buyers to find the products they want because of the language difference. In our case, the alphabetic queries that are input by non-Japanese buyers should be translated into Japanese to show product pages which they want to find.

There are many cases that non-Japanese people get no or wrong result from their research queries and they are classified into three cases. The first is the case where the non-Japanese people write Japanese product names in alphabets and we expected that this case would be solved by transliteration. The second is the case where non-Japanese people write English product names and this would be solved by translation. The final is the others, for example, the proper nouns such as the names of the animation characters etc., and the misspellings. Among them, we expected that the first case is the most frequent because 53.7% of them could be fully transliterated in the corpus. Hence, we propose the transliteration from the alphabetic queries to Japanese product names cf., from lunchbox to “ランチボックス (translation into English: lunchbox, pronunciation in Japanese: ranchibokkusu)”.

Also, many researches about transliteration have been accomplished for clean data, however, as far as we know, there have been no research about transliteration for noisy query data. Thus, we investigated which method is the best for query transliteration, using the parallel data of the alphabetic queries which did not provide any products when non-Japanese people searched (i.e.,





26 th Pacific Asia Conference on Language, Information and Computation pages 456-462

the Alphabet Queries) and the Japanese queries which are transcribed from them (i.e., the Correct Queries). We refer to this parallel data as the pair corpus and Table1 shows the examples of it. Here, the Alphabet Queries are the keywords which were actually used by non-Japanese user on a Japanese website and the Correct Queries were transcribed by volunteers. However, some pairs of them were not transliterated into Japanese phonogram, i.e., Katakana or Hiragana; they also have some free translations or Chinese characters. Instead of manually editing the raw data, we automatically filter out those word pairs using two filters: Chinese character filter (CF) and Chinese character and alphabet filter (CAF). The experiments revealed that the HMM worked the best which gave precision of 0.448 when the CF was used for the looser evaluation.

## 2 Related Works

Many works on transliteration have been accomplished so far including phonemic, orthographic, rule based approaches, and approaches which use machine learning. For example, Aramaki et al. (2009) presented the discriminative transliteration model using the CRF with the English-to-Japanese transliteration. In other language, Wang et al. (2011) worked on the English-Korean translation. They compared four methods: grapheme substring-based, phoneme substring-based, rule-based and mixture of them. Jing et al. (2011) developed the English-Chinese transliteration, which consists of many-to-many alignment and the CRF (conditional random fields) using accessor variety.

However, as far as we know, the transliteration using noisy query data has not been accomplished so far. Hence, we propose to transliterate the Alphabet Queries into the Correct Queries using the pair corpus and compared three transliteration methods to investigate which is the best for query transliteration.

It is also possible to use the dictionary-based approaches, however, the pair corpus includes many new words like the title of the comics and the names of the animation characters that are not listed in the dictionaries. Therefore, the dictionary based approach is not so powerful for transliteration comparing with that for translation. Thus, we employed the phonemic approach and the probabilistic method or the machine learning was used for the transliteration from phonemes to Japanese product names (i.e., the Correct Queries).

## 3 Transformation from the Alphabet Query to Phoneme

We employed the phonemic approach; the Alphabet Queries were transformed into phonemes and then are transliterated. The transliteration was carried out as follows:

1. Transform the Alphabet Queries into phonemes using a English-Phoneme dictionary (Section 3.1)
2. Filter the Correct Queries to clean the noisy data (Section 3.2)
3. Calculate the translation probabilities from phonemes to Japanese characters (Section 3.3)
4. Align the phonemes and Japanese characters (Section 3.4)
5. Transliterate the phoneme queries into Japanese words using the probabilistic method or machine learning (Section 3.5)

The remainder of this section describes these five steps. The steps from one to four were the generation phase of the training data and the step five was the transliteration phase.

### 3.1 Transform the Alphabet Queries

CMU Pronunciation Dictionary[1] (CMUdict) was used for the transformation from the Alphabet Queries to phonemes. Thus, we targeted only the alphabetic queries which include at least one phoneme in it. We obtained 2833 Alphabet Queries after this process.

### 3.2 Filter

Since the pair corpus is noisy, the training data were narrowed down and were refined using the following two different filters:

[1]http://www. speech.cs.cmu.edu/cgi-bin/cmudict

| method | BASE | BIGRAM | HMM | CRF |
|---|---|---|---|---|
| system output | フャブーンク (fabuunku) | ファブリック (faburikku: the correct answer) | ファブリック (faburikku: the correct answer) | フブック (fubukku) |
| evaluation | 1 | 3 | 3 | 2 |

Table 2: The system output when the input was “fabric” (Alphabet Query) and evaluation

1. Chinese character filter (CF)
2. Chinese character and alphabet filter (CAF)

These two filters were compared to adjust the quality and the amount of the training data. CF filtered out the pair which has Chinese character Correct Queries and CAF filtered out the pair which has Chinese character Correct Queries and alphabetic Correct Queries. In other word, the pair filtered by CFA has only Katakana and Hiragana Correct Query

Table 1 lists the example of the pair corpus and the characteristics of the Alphabet and Correct Queries. Here, we focused on the character type of the Correct Queries because of the characteristics of the pair corpus.

As shown in the table, although we want to use only the transliteration pairs as the training data, it is not easy to distinguish them. (The pair corpus consists of only the Alphabet and Correct Queries.) The first problem was that some Correct Queries are written not only in Japanese phonogram, i.e., Katakana or Hiragana, but also in ideograms, i.e., Chinese characters that have many ways to pronounce (cf. Tokyo-東京 (Tokyo,toukyou)).

Thus, we carried out the filtering by the character types to obtain as many transliteration pairs as possible. We expected that this process would improve the quality of the training data because in many cases, if the Correct Queries were in Katakana, they were transliterated. However, we have to keep in mind that the Correct Queries in Katakana could be free translation as shown in Table1 on the second line (cf. Miyazaki –ジブリ (translation into English: GHIBRI, pronunciation in Japanese: ziburi, meaning: a film studio name) .

| Alphabet Query (type of query) | Correct Query (translation into English, pronunciation in Japanese ) | transliteration (L) or translation(T) | Type of Characters of Correct Query |
|---|---|---|---|
| Doraemon (animation’s character name) | ドラえもん (Doraemon, doraemon) | L | Katakana, Hiragana |
| Miyazaki (person's name) | ジブリ (GHIBRI, ziburi) | T | Katakana |
| AKB48 poster (pop group’s name, poster) | AKB48 ポスター (AKB48 poster, eikeibii48 posutaa) | L | Katakana, Alphabet |
| Ufm rod (brand name, rod) | Ufm ロッド (Ufm rod, uefuemu roddo,) | L | Katakana, Alphabet |
| Tokyo adidas (place name, brand name) | 東京 adidas (Tokyo adidas, toukyou adidasu) | L | Chinese character, Alphabet |
| Dress Tokyo (general noun, place name) | 原宿　ドレス (Harajuku dress, Harajuku doresu) | L, T | Chinese character, Katakana |

Table 1: The example of the pair corpus and the characteristics of the Alphabet and Correct Queries

Here, we filtered out the pair which has alphabetic or Chinese character Correct Queries to refine the pair corpus more (CAF: The shaded data with light gray and the shaded data with gray were removed). However, if we filter out too many query pairs to improve the quality of the training data, we may not be able to obtain enough training data for the probabilistic methods or machine learning. Therefore, we filtered out the pair corpus which has Chinese character Correct Queries (CF: The shaded data with gray were removed). Namely, we used two kinds of filters to find out which of those is the best for query transliteration.

We could use 78.5% and 25.2% of the pair corpus to calculate the translation probabilities by using the CF and the CAF, respectively.

### 3.3 Calculation of Translation Probabilities

The transliteration probabilities, from the phonemes of the Alphabet Queries which were transformed in Section 3.1 to the Correct Queries which were filtered in Section3.2, were calculated using the filtered pair corpus. We used the GIZA++[2] toolkit (Och and Ney, 2003) to calculate them. Here, we set phonemes as the source language and Japanese character as the target language.

### 3.4 Alignment

The alignment of phonemes and Japanese characters which is necessary before the transliteration was carried out for each query pair. The Dijkstra algorithm was used to make alignments. Fig.1 shows the alignment of the phonemes of document and its transcribed word ドキュメント (document, dokyumento). In Fig 1, the horizontal axis represents the phonemes of the Alphabet Queries and the vertical axis represents the Correct Queries. We used the negative logarithm of the translation probabilities (which are calculated in Section3.3) as costs of the alignment. Also, we set logarithm of 10-20 as the cost when no translation probabilities were obtained. (cf., the horizontal direction and vertical direction in Fig 1 are the cases).

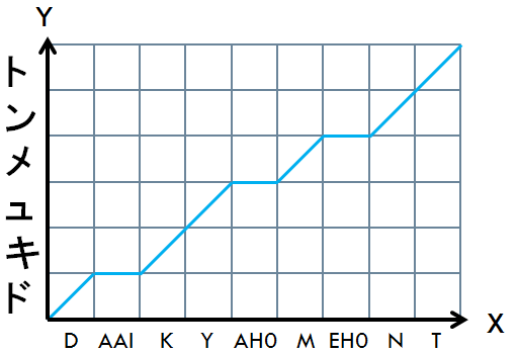


Figure 1: The alignment of the phonemes of *document* and its transcribed word ドキュメント (document, dokyumento)

Figure 2 shows the result of the alignment when the Alphabet Queries was *document* and the Correct Queries was ドキュメント (document, dokyumento). NULLJ and NULLP in Figure 2 represent the alignments in the horizontal and vertical directions respectively.

```
[D - ド(do)]
[AAI - NULLJ]
[K - キ(ki)]
[Y - ュ(yu)]
[AH0 - NULLJ]
[M - メ(me)]
[EH0 - NULLJ]
[N - ン(n)]
[T - ト(to)]
```

Figure 2: The result of the alignment of the phonemes of document and ドキュメント (document, dokyumento)

### 3.5 Transliteration

The transliteration was carried out using the probabilistic method or machine learning. We compared the following three different approaches were applied based on the alignments which were obtained in Section 3.4:

1. BIGRAM: The Bigram Model
2. HMM: The Hidden Marcov Model
3. CRF: The CRF model

We used NLTK[3] for BIGRAM and the HMM and adopted the CRF++[4] toolkit for the CRF. We trained the CRF models with the unigram, bigram, and trigram features. The features are shown in the following.

- Unigram: s-2, s-1, s0, s1, and s2
- Bigram: s-1s0 and s0s1
- Trigram: s−2s−1s0, s−1s0s1, and s0s1s2

We set parameters as f=50 and c=2. We set f=50 because the kinds of features were so variable.

[2] http://www.fjoch.com/GIZA++.html

[3] http://www.nltk.org/

[4] http://crfpp.googlecode.com/svn/trunk/doc/index.html

In addition, we used BASE method without machine leaning as baseline.

- BASE: The method where the most probable Japanese characters were selected for each phoneme from the translation probabilities.

## 4 Experiment and Evaluation

Five-fold cross validation was used in the experiments using the pair corpus. Note that we used 2833 Alphabet Queries which include at least one phoneme in the CMU dictionary. Here, only the training data were refined via two kinds of filter that are introduced in Section 3.1 because the system should not know the Correct Queries of the test data. Thus, the test data include some cases that cannot be transliterated, such as the case whose Correct Query is free translated from the Alphabet Query. One thousand five hundreds twenty one queries out of 2833 can be fully transliterated, which means a kind of upperbound of our system is 0. 537.

The system outputs were evaluated by twenty native Japanese speakers. We used human raters rather than the automatic evaluation such as the automatic method which uses the edit distance to evaluate this system because the Correct Queries is noisy and not always transliterated. The evaluations were graded on three scales (three is the highest and one is the lowest). Table2 presents the system outputs and evaluations when the input is “fabric”. In this table, the evaluation score is three when we got the ideal output, i.e., “ファブリック” (fabric, faburikku). We defined “precision 3” and “precision 3 or 2” as follows:

$$\text{precision } 3 = \frac{\left(\begin{array}{c}\textit{The total number of system outputs}\\ \textit{which are evaluated as } 3\end{array}\right)}{\textit{The total number of Query pairs}}$$

$$\text{precision } 3 \text{ or } 2 = \frac{\left(\begin{array}{c}\textit{The total number of system outputs}\\ \textit{which are evaluated as } 3 \textit{ or } 2\end{array}\right)}{\textit{The total number of Query pairs}}$$

Tables 3 and 4 summarize the precision of strict and looser evaluation respectively (i.e., the precision 3 and the precision 3 or 2). We also evaluated the system of BIGRAM and HMM without those filters and Table 5 show their precisions.

| | CF | CAF |
|---|---|---|
| BASE | 0.036 | 0.044 |
| BIGRAM | 0.029 | 0.071 |
| HMM | 0.062 | **0.121** |
| CRF | 0.064 | 0.046 |

Table 3: “The precision 3” of strict evaluation

| | CF | CAF |
|---|---|---|
| BASE | 0.323 | 0.209 |
| BIGRAM | 0.190 | 0.270 |
| HMM | **0.448** | 0.373 |
| CRF | 0.316 | 0.199 |

Table 4: “The precision 3 or 2” of looser evaluation.

| | The precision 3 | The precision 3 or 2 |
|---|---|---|
| BIGRAM | 0.032 | 0.151 |
| HMM | 0.043 | 0.273 |

Table 5: The precisions of BIGRAM and HMM without the filters.

## 5 Discussion

Although there were some reports that say the CRF model achieved high accuracy for transliteration when English to non-Japanese language was carried out (Shishtla et al 2009), the HMM was the best in this research according to Tables 3 and 4. We think this is because that we used trigram features for the CRF in this experiment. When the Alphabet Query is a compound word which contains two or more words, we could not find that those words are separated and they are treated as one word. For example, suppose that the Alphabet Query was "super mario", and their phonemes were” S UW1 P ER0 M AA1 R IY0 OW0”. When the system considered the transliteration of M, it used the P in “S UW1 P ER0", which is two phonemes before M, as a feature. However, this "P" is unrelated with “M AA1 R IY0 OW0". These features sometimes caused some errors for the CRF in this manner.

In addition, according to these tables, the HMM and the CRF were always superior to BASE but BIGRAM was not the case. This indicates that BIGRAM should not be used for the query transliteration.

Next, according to Tables 3, 4, and 5, the precisions without the filters were completely lower than those with the CF and CAF. It indicates that the filters were useful for transliteration of the noisy data.

In addition, as mentioned in Section 3.2, the amount of training data after the CAF was used (714 records) is much less than those after CA was used (2223 records). Nevertheless, as shown in Table 3, the CAF had the better result for the strict evaluation. These results revealed that it is better to use the CAF if we could obtain much more data.

Moreover, according to Table 3, the precisions when the CAF was used are higher than when the CF was used except the case when the CRF was used. In contrast, the CAF filter outperformed the CAF filter except the case when BIGRAM was used for machine learning in Table 4. In other words, the CAF is superior to the CF in Table 3, i.e., the precision of the strict evaluation, but the CF was superior to the CAF in Table 4, i.e., the precision of the looser evaluation. We think that these results indicate that the CAF should be used to obtain transliteration whose quality is high and the CF should be used if we want loose but many transliterations. These results indicate that the filters should be selected depending on the amount of the training data and the purpose of the application.

Then, we counted frequencies of the Alphabet Queries whose score is three and found that many of them frequently occurred. For example, the word *figure* appeared 102 times in the Alphabet Queries. Here, Table 6 lists the number of the Alphabet Queries and their averaged scores according to their frequencies when the HMM and the CAF were used. For example, the Alphabet Queries which occurred once were 417 and their averaged score was 1.77. Figure 3 shows the relation between the frequencies of the Alphabet Queries in the training data and their averaged score when the HMM and the CAF were used.

These table and figure show that the Alphabet Queries which occur many times tend to be high quality. We think this indicates that the precision of the transliteration may improve if we can have more data.

Furthermore, the number of Japanese characters tended to be smaller than that of the phonemes of the Alphabet Queries. We think that this is because the tag NULLJ frequently occurred in the alignment step and the precision may improve if the cost of NULLJ was selected more carefully.

Finally, we think we can use the translation system using the other methods such as the dictionary-based approach in conjunction with our transliteration system to get the right match for many queries. We think we can also try the orthographic approach in the future.

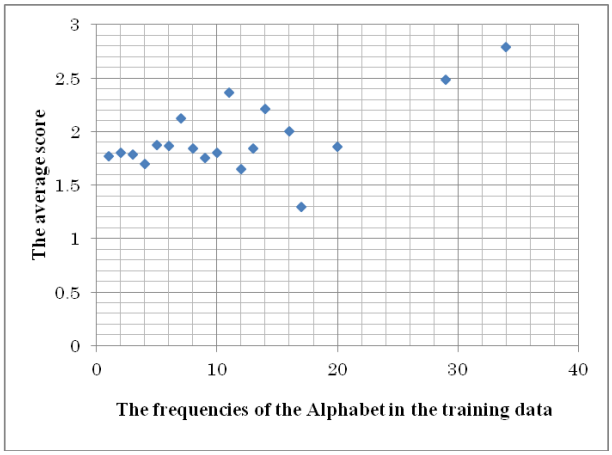


Figure 3: The relation between the frequencies of the Alphabet Queries in the training data and their averaged score when the HMM and the CAF were used.

| Frequencies | The number of the Alphabet Queries | Averaged scores |
|---|---|---|
| 1 | 417 | 1.77 |
| 2 | 124 | 1.78 |
| 3 | 57 | 1.79 |
| 4 | 21 | 1.67 |
| 5 | 14 | 1.87 |
| 6 | 13 | 1.87 |
| 7 | 4 | 2.13 |
| 8 | 4 | 1.84 |
| 9 | 3 | 1.75 |
| 10 | 5 | 1.81 |
| 11 | 2 | 2.36 |
| 12 | 3 | 1.65 |
| 13 | 1 | 1.85 |
| 14 | 1 | 2.21 |
| 16 | 1 | 2.00 |
| 17 | 1 | 1.29 |
| 20 | 2 | 1.86 |
| 29 | 1 | 2.48 |
| 34 | 1 | 2.79 |

Table 6: The number of the Alphabet Queries and their averaged scores according to their frequencies when the HMM and the CAF were used.

## 6 Conclusion

In this paper, we proposed to transliterate the inputs of the non-Japanese user i.e., search queries written in English alphabets, into Japanese Katakana using the pair corpus. Since this corpus includes some noise for transliteration such as the free translation, we carried out the filtering using the character types. Two kinds of filters, i.e., the CF and the CAF, were compared to adjust the quality and amount of the train data. The experiments revealed that the filters should be selected depending on the amount of the training data and the purpose of the application.

In addition, we compared three probabilistic or machine leaning methods, i.e., BIGRAM, the HMM, and the CRF using the pair corpus to investigate which is the best for query transliteration. The experiments show that the HMM methods worked the best. We think the HMM outperformed the CRF because we used trigram features for the CRF. Since the Correct Queries include many compound words, they caused some errors.

Finally, the experiments also indicate that the precision of the transliteration may improve if we can have more data or if the cost of NULLJ was selected more carefully in the alignment step.

## Acknowledgement

We would like to thank jGrab (http://www.j-grab.com/) which provide us the parallel data of alphabet queries and Japanese product names.

## References

Eiji ARAMAKI and Takeshi ABEKAWA. 2009. Fast decoding and Easy Implementation:Transliteration as Sequential Labeling, Proceedings of the 2009 Named Entities Workshop , ACL-IJNLP 2009, pages 65-68.

Steven Bird, Ewan Klein, and Edward Loper. 2009. Natural Language Processing with Python: Analyzing Text with The Natural Language Toolkit, O'Reilly.

Mike Tia-Jian Jiang, Chan-Hung Kuo, Wen-Lian Hsu. 2011. English-Chinese Machine Translitaration using accessor Variety Features of Source Graphemes. Proceedings of the 2011 Named Entities Workshop, IJCNLP 2011, pages 86-90.

Canasai Kruengkrai, Thatsanee Charoenporn, Virach Sornelertlamvanich 2011. Simple Discriminative Training for Machine Transliteration. Proceedings of the 2011 Named Entities Workshop, IJCNLP 2011, pages 28-31.

Franz Joseph Och, Hermann Ney 2003. A sys-tenmatic comparison of various statistical alighnment models. Association for Computational Linguistics, ACL 2003, 29(4):417-449.

Praneeth Shishtla, Surya Ganesh V, Sethuramalingam Subramaniam, and Vasudeva Varma. 2009. A Language-Independent transliteration Schema-Using Character Aligned Model At NEWS 2009, Proceedings of the 2009 Named Entities Workshop , IJNLP 2009, pages 40-43.

Yu-Chun Wang, Richard Tzong-Han Tsai. 2011. English-Korean Named Entity Transliteration Using Statistical Substring-based and Rule-based Approaches. Proceedings of the 2011 Named Entities Workshop, IJCNLP 2011, pages .32-35.

Min Zhang, Haizhou L, A Kumaran and Haizhou Li. 2011. Report of NEWS 2011 Machine Transliteration Shared Task. Proceedings of the 2011 Named Entities Workshop, IJCNLP 2011, pages 1–13.