Acquiring Vocabulary for Predictive Text Entry through Dynamic Reuse of a Small User Corpus

Kumiko TANAKA-Ishii[*] and **Daichi Hayakawa**[†] and **Masato TAKEICHI**[‡]
The University of Tokyo
7-3-1 Hongo, Bunkyoku, Tokyo, Japan.
kumiko,takeichi@mist.i.u-tokyo.ac.jp, daichi@oak.dti.ne.jp

Abstract

As mobile computing and communications have become popular, predictive text entry systems have become an increasingly important technology. Existing methods still need refinement, though, with respect to personalization, especially how to acquire vocabulary not pre-registered in the system dictionary. In this paper, we report on an automatic method that dynamically obtains a user specific vocabulary from the user's unanalyzed documents. When a user makes an entry, the system dynamically extracts the corresponding chunks from the user text and suggests them along with words suggested by the dictionary. With our method, texts in a particular style or concerning a specific domain can be entered using a predictive text entry system. We verified that a large amount of words not registered in the dictionary can be entered using our method.

1 Introduction

Recent advances in technologies now allow various means of information entry, such as character or speech recognition. Still, entry using a keyboard remains dominant because of its ease of implementation and its utility.

While various text entry methods can be used with a keyboard, our concern in this paper is *predictive text entry*. Any predictive method allows text to be entered continuously in three stages:

1. The user enters an *ambiguous* character sequence
2. The text entry system looks for corresponding candidates in a dictionary pre-attached to the software. It sorts the candidates with regard to the context and displays them to the user.
3. The user chooses his preferred word from among the candidates.

For example, Chinese pinyin-hanji conversion is a predictive text entry having an ambiguous sequence being pin yin and target words which are hanji words.

Historically, such predictive text entry systems have been popular only in East Asian countries whose languages use many characters. The problem with these languages was that the number of characters to be handled exceeds the number of keys on a keyboard. Therefore, predictive text entry was invented to enable projection of a wide range of characters using the limited number of keys on a keyboard.

This problem has become more international as smaller machines have been developed. Current devices can be as small as a wristwatch (IBM, 2001), so the number of characters in any language will exceed the number of buttons available for input. As a result, predictive text entry has been widely discussed in the academic and industrial domains. The T9 method of entering a digit sequence and predicting words is an example of one way to deal with this problem (Tegic, 2000). Research has even shown that an entry can be made with reasonable efficiency using only four buttons if a predictive text entry method is applied (Tanaka-Ishii et al., 2002).

The major drawback to this predictive text

[*] Language Informatics Laboratory, Information Technology Center

[†] Interfaculty Initiative in Information Studies

[‡] Graduate School of Information Science and Technology

Table 1: The Rate of Unknown Words

title (English)	R1	R2
Adventures Of Sherlock Holmes	2.54	7.48
Chat	14.78	15.11
The Merchant of Venice	7.34	13.00
Patent	2.94	7.74
RFC1459(Protocol Manual)	3.16	15.55
title (Japanese)	R1	R2
Patent	2.76	17.35
RFC1459J	16.36	38.13
Tales of Genji (8-9 th century)	13.03	2.91
I am a Cat (19 th century)	6.40	2.35
Chat	13.20	13.97

entry method is related to the dictionary use. The user cannot enter words *not registered* in the dictionary (what we refer to as **unregistered** words in the following). Thus, a conventional predictive entry system cannot easily handle text written using a special vocabulary or in an unusual style; for example, text written in a particular dialect, or using old words or words with specific technical meanings.

This problem is currently handled through the creation of a user dictionary. Users can enter unregistered words in some way (e.g., character by character) and register the words into the user dictionary. After that, the system ranks these words highly when the user enters the corresponding sequence. However, it is the user's responsibility to register the vocabulary into the user dictionary and this often becomes a cumbersome task.

To alleviate this problem, some companies offer dictionaries of vocabularies from specific domains; for example, the chat dictionary or a dictionary of dialect (JustSystems, 2002). However, the style that any one user enters into a computer is likely to be unique, or more precisely, *user specific*.

We therefore propose a better method that works by dynamically processing a small user corpus. This is based on an interesting observation that a user typically *reuses* vocabulary at a 70% rate after entry of only a small amount of text. Based on this property, we have created a simple predictive text entry system that dynamically extracts unknown words from the user corpus. In this paper, we show how this system solves the problem of unregistered words. We start in the next section by explaining how we came to observe the property that the typical reuse rate is 70%.

2 The Property of Editorial Behavior

2.1 Data

In this paper, we discuss our experiments regarding English and Japanese texts. For our research, we used texts from diverse domains in these two languages, as shown in Table 1. Each text reflects individual's personal writing style.

We define an unregistered word as any vocabulary not appearing in the 30-Mbyte corpus of WSJ and Mainichi newspaper. Of course, the word dictionary used by the predictive text entry can be larger. However, this won't make any difference regarding our basic argument in this paper, since the problem of unregistered words always exists no matter what size of dictionary we use.

As our concern was the unregistered vocabulary, we first investigated the percentage of unregistered words. We measured two rates:

$$R1 = \frac{No.\ of\ unknown\ words}{No.\ of\ total\ words} \tag{1}$$

$$R2 = \frac{No.\ of\ different\ unregistered\ words}{No.\ of\ different\ words} \tag{2}$$

Japanese texts were all manually segmented (because they contained unregistered words), so statistics were calculated using the first 20 Kbytes of text. To be consistent, we used 20 Kbytes for English texts, too.

Table 1 shows $R1$ and $R2$ for our test texts, with results in the upper half for English texts and those in the lower half for Japanese. The rate of unknown words differed according to the type of text. It was especially high for the technical, colloquial, or old texts. Thus, both recent and old texts can contain a large number of unregistered words. As our *registered* words were extracted from newspapers, we would guess that newspapers usually use a standardized vocabulary based on words that have been used long

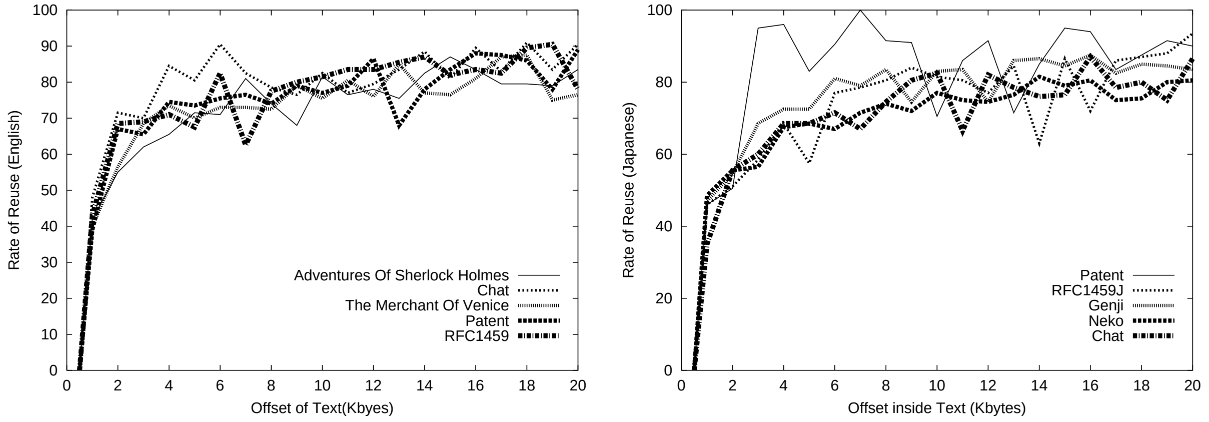

Figure 1: Vocabulary Reuse Rate and Offset (English and Japanese)

enough to be well known, but not yet forgotten. That unregistered words occur more in recent and colloquial texts shows how serious the problem of unregistered words can be.

2.2 Vocabulary Reuse

To improve predictive text entry, we need to provide an alternate source for finding missing vocabulary so that the system may suggest words from that source. Since the vocabulary depends on the user's context, it is natural to presume that the missing vocabulary should exist within the user's text. Therefore, we analyzed how vocabulary is reused while a user edits text.

A text is formed from a word sequence. When a word is randomly picked from a sequence, we can group the word into two categories: *reused* or *unused*. Unused words appear for the first time in the sequence, and **reused** words have already appeared. Note that this differs from the notion of unregistered words: there are unregistered words that can be grouped as used or reused.

We investigated how the reused word rate changed according to the offset of a text. We marked the text at the offset of 500 bytes and counted the reuse word rate in the 1000-byte window. The results are shown in Figure 1, where the horizontal axis shows the offset from the head of the text (in Kbytes), and the vertical axis shows the vocabulary reuse rate. Results for the English texts are shown on the left, while those for the Japanese are shown on the right.

At the beginning of the 500-byte text, the reuse rate was 0% and then it increased as the text proceeded towards the end. This increase leveled off at around 70% to 80%, curiously, in both languages. Also, we were surprised to see how similar nine of the texts were to each other (the exception being the Japanese patent text). These results suggest that *context* is provided by 70% to 80% of the vocabulary and the story evolves through the rest.

From this observation, we would expect a typical user to reuse text equivalent to around 70% to 80% of the vocabulary only after an offset of several Kbytes. If so, unregistered words must also be reused. Therefore, we next studied the reuse rate for unregistered words. Here, we calculated the average reuse rate for unregistered words of all text types.

Figure 2 shows our results for English and Japanese. In both cases, the rate tended to rise and fluctuate between 20% to 80%. Note that this fluctuation was caused by the sparseness and also by the unregistered word rate differing according to the text. These results confirmed our expectation that unregistered words are also reused.

Our observations suggested that if unregistered words can be automatically extracted from a user corpus, they can be suggested to the user and this will partly solve the problem of unregistered words. From the next section on, we explain how we have built a system to realize this

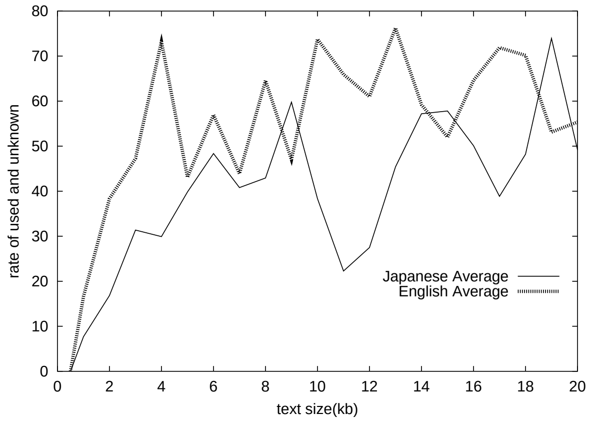

Figure 2: Reuse Rate of Unregistered Words and Offset (Average of English and Japanese)

idea based on such a reuse property.

3 Simple Evaluation System

Here, we explain the evaluation tool we built to verify the effectiveness of our idea. The system makes use of a small user corpus of about 20 Kbytes from which the unregistered words are extracted. The system is based on predictive text entry and allows text entry by repeating the following stages.

1. The user enters an ambiguous sequence.
2. The system looks for the corresponding parts in the user corpus and extracts chunks which might be the user's target.
3. The system sorts these chunks according to a particular evaluation function and shows them to the user.
4. At the same time, the ordinary candidates obtained from the system dictionary are also shown to the user.
5. The user chooses his target from among the two lists.

Note that stages 2 and 3 are added to the procedure we defined as predictive text entry in the §1. As for the ambiguous sequences, we chose to use the word-based prefix entry from among various predictive methods. This was because word-based entry is the simplest and most applicable worldwide, and because some papers have reported that entry by prefix increases entry efficiency (Tanaka-Ishii et al., 2000). Precisely, for English, we chose single-tap entry method for mobile phone with alphabet assignment of Figure 3 (similar to ZI's method (ZI-Corp., 2000)) on digits and for Japanese, we chose word-based kana-kanji conversion by prefix entry (similar to PO-Box method (Masui, 1999)). As an English example, '84' corresponds to *the*, *this*, *they*, *view*, *time* which are the candidates from the dictionary. With stages 2 and 3, '84' can acquire candidates such as 'thee' or 'thou' if the user corpus contains them.

Figure 4 shows an example of our tool when "Romeo and Juliet" is adopted as the user corpus. The user is trying to enter Juliet's famous speech "O Romeo, Romeo! wherefore art thou Romeo?". Note that all of these English words are unregistered words. In this example, "O Romeo, Romeo!" was already entered (shown before the prompt '>'), and the user wanted to enter the rest. When the user enters the head part of the target text by digits (943733, which corresponds to 'wheref' of 'wherefore'), the system extracts possible chunks from the original Shakespearian text, and then suggests candidates to the user (only the candidates obtained from the user corpus are shown in Figure 4). The corresponding candidates in this example were *wherefore* (with a space afterwards), *wherefore*, and *wherefores*. The user would choose the first one, and thus the entry would continue. One might wonder as to who would want to enter Shakespear's text by using a mobile phone predictive text entry system. However, similar situations occur with technical or colloquial terms. In addition, such difficulty occurs usually in East Asian countries because their main text entry method is predictive even when using full-size computer keyboards.

Now, we have to explain two processes in detail: candidate extraction and ranking. We discuss these processes in the following two sections.

4 Candidate Handling

4.1 Extraction

Our problem was to obtain a particular chunk out of a text of about 20 Kbytes. One previous approach was to use term extraction methods as

0 symbols	1 symbols	2 ABC	3 DEF	4 GHI
5 JKL	6 MNO	7 PQRS	8 TUV	9 WXYZ

Figure 3: An Alphabet Assignment on Digits

```
.....
O Romeo, Romeo! > 943733
enter 'wheref'
(1.wherefore )
(2.wherefore)
(3.wherefores)
O Romeo, Romeo! > +1
choose No.1 candidate
O Romeo, Romeo! wherefore > 278
enter 'art'
(1.art thou happy.)
(2.art thou )
(3.art thou happy)
(4.aqua)
(5.art )
O Romeo, Romeo! wherefore > +2
choose No.2 candidate
O Romeo, Romeo! wherefore art thou > 76
enter 'Ro' of Romeo
(1.so)
(2.so )
(3.Romeo)
(4.sought )
(5.some )
O Romeo, Romeo! wherefore art thou > +3
choose No.3 candidate
O Romeo, Romeo! wherefore art thou Romeo> 0
enter the question mark (0 for symbols)
(1.?)
(2. )
O Romeo, Romeo! wherefore art thou Romeo> +1
choose No.1 candidate
O Romeo, Romeo! wherefore art thou Romeo?>
```

Figure 4: Entering 'Romeo and Juliet' with Our Simple Evaluation System (In a Manner of English Mobile Phone)

discussed in (Nakagawa and Mori, 2002). However, such methods are designed to obtain a limited number of strictly defined and specialized terms from a huge corpus assuming that taggers work properly. However, in our case, we should not assume that any language tools will work properly, because our target is the *unregistered words*. Therefore, since our target is to include non-segmented languages, our method should be string based.

Intuitively, what we want here is for the text chunk to be *used always together*. Under the constraint that the user corpus is small, what we may look at is the *repetition* of strings. However, if we extract all repeated strings, the number of candidates explodes and the user will be forced to look at inadequate chunks (because when "the" repeats, "th" or "t" are also repeated). Therefore, we decided to extract the *maximal repeated prefixes*. A maximal repeated prefix is a repeated string that does not occur as the prefix of another prefix string.

For example, if "abracadabra" is the user corpus, and the user entered "a", then

abra(2), abr(2), ab(2), a(5)

are the repeated strings that can be the candidates. Among these, "abr", "ab", and "a" (twice) occur as the prefix of two occurrences of "abra". Therefore, we eliminate these, which leaves:

abra(2), a(3) .

Another two "a" occurred as part of "abra", but it occurred as the *postfix* of "abra", so, it is shown among the candidates. As the maximal repeated prefixes can be quickly extracted using the suffix array (Manber and Myers, 1993), the system explained in the previous section transforms the user corpus into a suffix array when it is initiated.

4.2 Ranking

Candidates are displayed in a certain order. This order is determined by an evaluation function, and we chose that it be done using the PPM (prediction by partial match) framework (Bell et al., 1990). We decided to use the PPM because the context provides the best information for selecting adequate candidates. PPM integrates such a concept by interpolating the statistics obtained from the user corpus and the initial language model obtained from a huge corpus. PPM language models have been used in much of the earlier works (Ward et al., 2000)(Tanaka-Ishii et al., 2002), and has been found to be superior to other language models such as fixed n-grams and co-occurrence-based methods (Maruyama et al., 2001).

PPM can be situated as variant n-gram language models. It interpolates the n-gram counts in the user corpus and the statistics in the base dictionary. The following formula is used to estimate a probability for the ith element w_i, $P(w_i)$:

$$P(w_i) = \sum_{k=-1}^{kmax} u_k P_k(w_i) \qquad (3)$$

Here, k, the order, indicates the number of elements before w_i that are used in the calculation of $P_k(w_i)$. For example, $P_2(w_i)$ is estimated on the basis of the occurrence of w_{i-1} and w_{i-2}. $kmax$ is the length of available context and is set at 4, in our study (thus maximally 5-grams are concerned). $P_k(w_i)$ is calculated as:

$$P_k(w_i) = \frac{c_k(w_i)}{C_k} \qquad (4)$$

where C_k is the frequency of the current k element context, and $c_k(w_i)$ is the frequency with which w_i occurs in that context. $P_{-1}(w_i)$ describes a base initial probability assuming no context. In our case, w_i is a registered word or an unregistered chunk. For an unregistered chunk, the initial probability cannot be obtained because the word is *unregistered*. Therefore, we set the initial probability at a constant value.

For other k, $P_k(w_i)$ is calculated from statistics obtained from the user corpus. Here, elements cannot correspond to words, because the user corpus is unanalyzed. Therefore, contextual elements are counted by characters, whereas the current element in question is the unregistered chunk or the registered dictionary word.

Finally, u_k is a weighting probability that is multiplied by $P_k(w_i)$ to obtain the final $P(w_i)$. There have been many studies of u_k (Teahan, 2000), and we have chosen to use PPM-A(Bell et al., 1990), the simplest form, because our preliminary experiments showed no significant difference in performance among the various methods we tried.

5 Evaluation

5.1 Settings

Using the tool described in §3, we examined to what extent the predictive text entry system was improved. The experiment was done by automatically entering test text into our system.

The test text was prepared according to the following stages.

1. All texts used in §2 were separated at the sentence level.
2. Sentences were sorted into a random order to measure the average capability of the method.
3. First 5000 bytes were taken as the test text. For Japanese, this test text was all manually analyzed and cut into words. The rest of the text was used as the learning corpus by differentiating the size of the corpus. The smaller learning corpus was included in the larger corpora.

Stages 2 and 3 were repeated five times so that five different sets of the test and the learning corpora were obtained. All of the graphs that follow show the average of the test results for these five different sets (five-times cross validation).

The automatic text entry proceeded as follows. First, an ambiguous sequence corresponding to the first word was entered. Two lists of candidates were then shown by the system, one list obtained from the word dictionary, and the other from the user corpus. The string that corresponded to the prefix of the target test text was then chosen (the correct candidate). When there were multiple correct candidates among the two lists, the highest ranked candidate was chosen. If two correct candidates were equally ranked on the two lists, the longer candidate was chosen.

5.2 Results

First, we consider the rate of unregistered words being successfully entered with our method. Figure 5 shows the results for English (left) and Japanese (right), where the horizontal axis shows the user corpus size and the vertical axis shows the rate of successfully entered unregis-

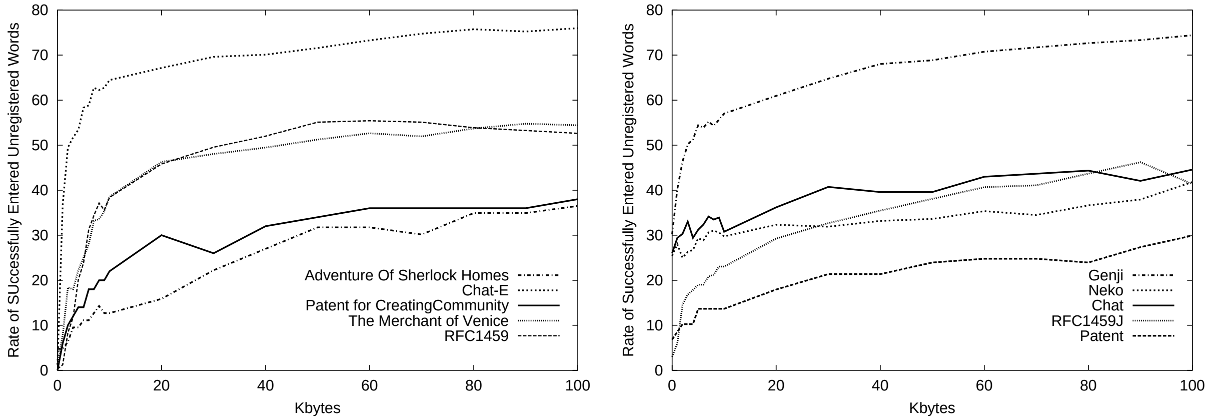

Figure 5: Successfully Entered Unregistered Words and Corpus Size

tered words[1]. Above 20 Kbytes, 20 to 75 % of the unregistered words were successfully entered. Depending on the amount of the unregistered text, the rate was as high as around 75%.

All unsuccessful cases were caused by low occurrence of unregistered words in the learned user corpus. More precisely,

- User corpus size is insufficient. The text with low unregistered word rate tends to have this problem (for example, Holmes or Patent in English).
- Unregistered words occurred *only once* in the user corpus. As our candidate extraction method is based on repeated strings, candidates of one occurrence cannot be extracted.

Additionally, when no target unregistered words occurred in the user corpus, nothing can be done with our method. However, if it appears once, we might be able to obtain it comparing the string with the dictionary although there still lies the problem of how to evaluate their importance. This point is one of our future work.

As this result was achieved by choosing candidates obtained dynamically from the user corpus lists, we plotted the choice rate from the user corpus list (Figure 6) among two lists (user corpus, and the dictionary). The horizontal axis again shows the user corpus size and the vertical axis shows the rate of candidates chosen from the user corpus (including the entry of registered words). Since we found little difference between the Japanese and English results, we show only the English results here. Interestingly, this rate was around 70% to 90% after using a 20-Kbyte corpus. Clearly this experiment reflects our observation in §2 that more than 70% of the vocabulary is reused from the user corpus. Thus, the problem of unregistered words can be partly solved by our system under the condition that the user can prepare a small text that has the same context as the current one.

One current drawback of our idea is that the number of candidates suggested to the user tends to grow as the corpus size increases. Figure 7 shows the average ranking change of the chosen words according to the user corpus size. It fluctuates until 20Kbytes, and from 20 Kbytes on, for some text, the ranking descends while the user text size increases. Although the property described in §2 suggests that the user corpus size is sufficient at about 10 to 20 Kbytes, the mechanism to rank candidates need to be refined. This will be the most important part of our future work.

6 Conclusion

Predictive text entry is now an important technology because of the scaling down of device size. Since the predictive entry methods are based on prediction using a word dictionary, a problem arises as to how to enter unregis-

[1] As for the Japanese case, some amount of unregistered words are already entered even without any user corpus. This is caused by 1. homonyms, or 2. differently segmented registered words forming the target.

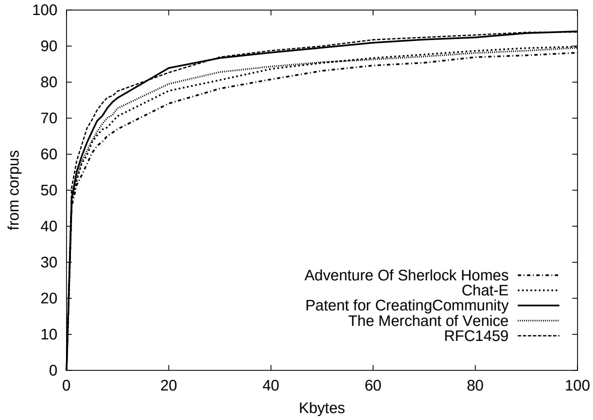

Figure 6: Choice Rate of Candidates from the User Corpus (English)

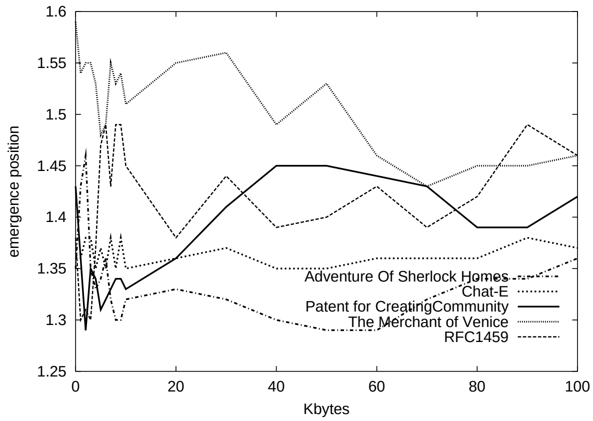

Figure 7: Average Ranking of Words (English)

tered words. In this paper, we have described a method to use a small amount of user corpus and dynamically extract chunks as candidates to complement the missing vocabulary.

Our approach is based on an observation regarding the editorial behavior of writers: 70% of the vocabulary is reused after only 10 to 20 Kbytes of text. Similarly, the unregistered words in the dictionary are also reused at a certain rate.

We have built a simple word-based predictive text entry system, and attached it to a tool that dynamically extracts unregistered words and suggests these to the user. We found that a large amount of unregistered words can be automatically acquired and entered. This was achieved by choosing more than 70% of the candidates obtained from user corpus of 20 Kbytes; this rate corresponded to our initial observation regarding unregistered words.

We are now building a front-end tool that can be attached to the predictive text entry system. This tool suggests candidates that are dynamically obtained from the corpus, merged with candidates from the system dictionary.

References

T.C. Bell, J.G. Cleary, and I. H. Witten. 1990. *Text Compression*. Prentice Hall.

IBM. 2001. Watchpad 1.5. Available from `http://www.trl.ibm.com/projects/ngm/`.

JustSystems. 2002. Atok 16 home page. `www.justsystem.co.jp/atok/atok16w/`.

U. Manber and G. Myers. 1993. Suffix arrays: a new method for on-line string searches. *SIAM Journal of Computing*.

T. Maruyama, K. Tanaka-Ishii, and M. Takeichi. 2001. Learning model for kana-kanji conversion system using PPM method. In *IPSJ NL-Seminar Note in (Japanese)*, volume 146.

T. Masui. 1999. Po_box an efficient text input method for handheld and ubiquitous computers. In *the ACM Symposium on User Interfface Software and Technology*.

H. Nakagawa and T. Mori. 2002. A simple but powerful automatic termextraction method. In *Computerm2: 2nd International Workshop on Computational Terminology (COLING-Workshop)*, pages 29–35.

K. Tanaka-Ishii, Y. Inutsuka, and M. Takeichi. 2000. Japanese input system with digits–Can Japanese be input only with consonants?—. In *Human Language Technology Conference 2001*.

K. Tanaka-Ishii, Y. Inutsuka, and M. Takeichi. 2002. Entering text using a four button device. In *the 19th International Conference on Computational Linguistics*, pages 988–994.

W.J. et al. Teahan. 2000. Probability estimation for PPM. In *NZCSRSC'95*. `http://www.cs.waikato.ac.nz/ wjt/papers/NZCSRSC.ps.gz`.

Tegic. 2000. Tegic home page. `http://www.t9.com`.

D. Ward, A.F. Blackwell, and D.J.C. MacKay. 2000. Dasher:A data entry interface using continuous gestures and language models. In *the ACM Symposium on User Interfface Software and Technology*.

ZI-Corp. 2000. Zi home page. Available from `http://207.229.18.241/`.