Phrase Extraction for Japanese Predictive Input Method as Post-Processing

Yoh Okuno
Yahoo! JAPAN Corporation
yookuno@yahoo-corp.jp

Abstract

We propose a novel phrase extraction system to generate a phrase dictionary for predictive input methods from a large corpus. This system extracts phrases *after* counting n-grams so that it can be easily maintained, tuned, and re-executed independently. We developed a rule-based filter based on part-of-speech (POS) patterns to extract Japanese phrases. Our experiment shows usefulness of our system, which achieved a precision of 0.90 and a recall of 0.81, outperforming the N-gram baseline by a large margin.

1 Introduction

Predictive input methods for personal computers or mobile devices have been quite popular (MacKenzie and Tanaka-Ishii, 2007). They suggest options of entire words or phrases to select when a user inputs first few characters or words.

Recently, the growth of the Web has increased the availability of large corpora for natural language processing. Large corpora are effective in generating dictionaries, since they include frequently used words and phrases.

One of the possible simple ways to enlarge a dictionary is the n-gram approach. N-gram is a word sequence of length n. The N-gram approach consists of the following steps: count n-gram sequences in the corpus and show the most frequent n-grams for user input. This approach enables the dictionary to cover most of the useful options.

However, such a naive n-gram approach has three major problems:

Trade-offs between lengths and frequencies Longer n-grams always have lower frequencies than shorter n-grams. For predictive input methods, longer options are favorable because they reduce user keystrokes much more.

Halfway options N-gram contains partial portions of eligible phrases. For example, the trigram of "you very much" has high frequency, which may be a subsequence of "Thank you very much". These options distract users.

Enormous memory consumption N-grams are also too large for client-side input methods. Predictive dictionaries are preferable to fit into memory for rapid access. Since input methods always remain in memory, it should save memory for other applications.

To cope with these problems, phrase-based approaches are considered. These approaches use phrase extraction to reduce unnecessary n-grams. A *phrase* represents a semantic or syntactic unit of a word sequence in texts. For predictive input methods, phrases should be rather comprehensive; we want to extract various phrases which users possibly input, containing noun phrases, verbal phrases, proper noun, idioms, and so on.

There are two types of approaches to extract phrases from a large corpus: pre-processing and post-processing approaches.

In a pre-processing approach, phrase extraction is applied to a corpus before counting. This setting is similar to a chunking task (Sang and Buchholz, 2000), extracting non-overlapping chunks from a corpus. In this approach, each time we try a new algorithm, re-execution of counting is required to construct a phrase dictionary. This is too painful and expensive.

For these reasons, we adopted a post-processing approach. In a post-processing approach, phrases are picked out from n-grams after counting. In addition to counting, cutting off n-grams with low frequencies significantly reduce data size of n-grams. Therefore, we can develop and run the phrase extraction algorithm in a local machine, using commonly-available script languages. Additionally, once we count frequencies of n-grams,

Proceedings of the Workshop on Advances in Text Input Methods (WTIM 2011), pages 48–52,
Chiang Mai, Thailand, November 13, 2011.

we need no more counting frequencies again to generate a dictionary after changing of the algorithm.

In this paper, we focus on the Japanese language to utilize grammatical knowledge. Since the Japanese language has many characters than physical keys, Japanese people normally use input methods called Kana Kanji conversion, therefore predictive input methods are easily brought in.

The rest of this paper is organized as follows: section 2 introduces related work in similar tasks. Section 3 describes an algorithm that we developed. Section 4 explains experiments and evaluations of our algorithm. Section 5 summarizes the whole paper and future work.

2 Related Work

There are many researches about predictive input methods since early days (Masui, 1999; Ichimura et al., 2000). They used manually constructed dictionaries, which is expensive to maintain and cannot be extended to large scale sufficiently. Komatsu et al. (2005) as well as Unno and Tsuboi (2011) used small corpora to generate options of prediction, but their coverage is limited.

Okuno and Hagiwara (2009) used Google n-gram for prediction, but n-gram approaches have problems described in the previous section. Google Japanese IME (Kudo et al., 2011) adopt a pre-processing approach to extract phrases from a huge Web corpus. However, the pre-processing approaches need large computational resources and hard to tune iteratively.

Manning and Schütze (1999) and Wan Yin Li (2006) described POS patterns for phrase extraction, but they are limited to noun phrases for two or three words. Su et al. (1994) applied decision tree for compound extraction, but their supervised learning approach needs training datasets.

3 Phrase Extraction as Post-Processing

Figure 1 shows the data flow of our system and an example. In this section, we describe each component of the data flow. Note that the earlier processes are executed in distributed environment, i.e., MapReduce. The later processes are implemented as local scripts.

3.1 Morphological Analysis

We used internal morphological analyzer to split Japanese texts into words and add morphological

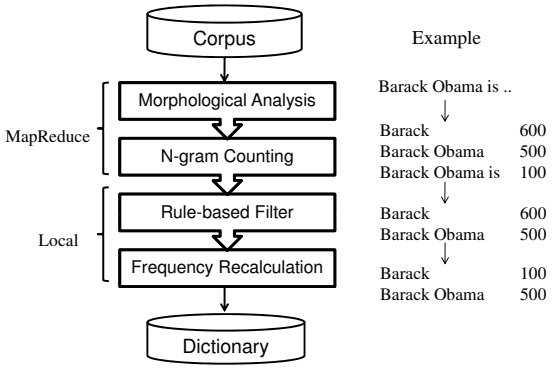

Figure 1: Data flow

Word	Read	POS	Form	Type
言う	いう	動詞	基本形	自立語
say	iu	verb	normal	content

Table 1: Morphological information

information shown in Table 1. POS and form (conjugation) tags are used for later rule-based filtering. We might use reading information in prediction time for Kana Kanji conversion.

3.2 N-gram Counting

We used MapReduce (Dean and Ghemawat, 2004) to count word n-grams. In the MapReduce framework, distributed file system stores a large corpus, and Mapper extracts n-grams in each machine. Then the system aggregates data into same n-gram groups, and Reducer calculates n-gram frequencies. Reducer also cuts off n-grams which have lower frequencies than a predefined threshold.

3.3 Rule-based Filtering

We developed a rule-based filtering based on POS patterns as regular expressions described in Table 2. The patterns are developed from preliminary investigation. There are two types of rules: valid and invalid rules. First, our filter leaves n-grams which match at least one valid rule and filters out others. Then it filters out n-grams which match at least one invalid rule and leaves others.

Table 3 shows the n-grams which match each rule. Valid rules are designed to leave n-grams which are interpreted as a Japanese segmentations or *bunsetsu*, consisting of at least one content word followed by function words. This is based on an assumption that users input units of segments one

Rule	Group	Validity	N value	Pattern	Description
1	Type	Valid	>2	^C+F+$	Constitute segment (文節)
2	Type	Valid	[1,3]	^C{1,3}$	Contain only contents (自立語)
3	POS	Invalid	All	^Suffix	Start with suffix (接尾辞)
4	POS	Invalid	All	Prefix$	End with prefix (接頭辞)
5	POS	Invalid	[2,3]	Adverb$	End with adverb (副詞)
6	Form	Invalid	>2	Continuative$	End with continuative form (連用形)
7	Form	Invalid	>2	Imperfect$	End with imperfect form (未然形)
8	Form	Invalid	[2,3]	Hypothetical$	End with hypothetical form (仮定形)

Table 2: POS rules. In rule 1 and 2, C means a content word and F means a function word.

Rule	Japanese	English translation
1(V)	食べ ま し た	I have eaten
2(V)	株式 会社	stock company
3(I)	的 な イメージ	image like
4(I)	明日 の プチ	tomorrow's petit
5(I)	この 話 は また	this talk is later
6(I)	行っ て き まし	have gone to
7(I)	わかり ませ	can't understand
8(I)	考え なけれ	have to think

Table 3: Examples (V:Valid, I:Invalid)

by one, and a prediction should display its options on boundaries of segments.

While valid rules roughly filter out n-grams which get across a border of segmentations, invalid rules filter out unnecessary n-grams in a rather fine-grained way. Invalid rules use POS tags to exclude n-grams whose leftmost word is a postposition particle, n-grams whose rightmost word is a prefix word, and so on. These rules are tuned for high precision, rather than high recall.

3.4 Frequency Recalculation

Although most of unnecessary n-grams are filtered out by the rule-based filter, there are still some problems like halfway n-grams which have larger frequencies than longer eligible phrases. This problem is caused by duplicated count. For example, words in a 2-gram phrase may be double counted; words in a 3-gram phrase may be triple counted[1].

To handle this problem, we propose an algorithm to recalculate frequencies of n-grams. Figure 2 describes our recalculation algorithm. Our algorithm starts from the longest n-grams and processes shorter n-grams one by one. All subsequences of n-grams are extracted and their frequencies are reduced by the frequencies of the entire n-grams. Finally, we get phrases and their frequencies with almost no duplicated counting.

```
Recalculate(ngram, freq):
  for n = N_MAX to 1
    for each p in ngram[n]
      for each s in subsequence(p)
        if s is in ngram
          freq[s] -= freq[p]
  return freq
```

Figure 2: Frequency Recalculation Algorithm

For example, a 3-gram "機動 戦士 ガンダム" (MOBILE SUIT GUNDAM) has a lower frequency than "機動 戦士" (MOBILE SUIT), which is rarely used by itself. In this case, our frequency recalculation works well and the former frequency surpasses the later one.

4 Experiment

4.1 Methods and Metrics

In order to evaluate our system, we used human judgments for samples from n-gram as below:

1. Some samples are randomly extracted from original n-grams before filtering.
2. The samples are classified into necessary or not by human judgments.
3. Precision and recall are calculated by comparing manually annotated samples and extracted phrases.

Assuming n-gram contains all necessary phrases and approximating all n-grams by small samples, we obtain our metrics as below:

[1]Pre-processing approaches do not cause this problem.

$$\text{Precision} = \frac{\text{number of valid samples in dictionary}}{\text{number of samples in dictionary}}$$

$$\text{Recall} = \frac{\text{number of valid samples in dictionary}}{\text{number of valid samples}}$$

$$\text{F-measure} = \frac{2}{1/\text{Precision} + 1/\text{Recall}}$$

4.2 Dataset and Judgment

We used a Japanese blog corpus whose size is about 300GB, containing 70G words. Since blogs are written by ordinary people, we expect them to fit typical use cases.

We counted n-grams from the corpus with 20 machines of a Hadoop MapReduce cluster. The counting took 17 hours. We set n from 1 to 5 and cut-off threshold to 1,000. Resulted n-gram has about 6M unique n-grams and size of 700MB in plain text. Then we applied our rule-based filter extracting 1.2M different phrases and size of 100MB in plain text. The filtering took only 5 minutes in a local machine.

We conducted sampling from original n-grams in two ways: token-based and type-based. Token-based sampling means that samples are extracted from n-gram according to their probabilities or relative frequencies. Type-based sampling uniformly extracts entries from n-gram.

After sampling, 5 people judged the same 200 n-grams into necessary phrase or not by hand, for each token-based and type-based sampling. In addition to the definition of *phrase* described in section 1, we assumed typical Japanese blog writer as target user for clarification.

4.3 Result and Error Analysis

Table 4 shows our average evaluation results for both phrases extracted by our system and n-grams as baseline. N-gram as baseline has recall of 1.0 because of the assumption, but a low precision of 0.41 for the reasons described in section 1.

We found our rules achieve a high precision of 0.90 and a recall of 0.81 for token-based sampling, but a lower recall for type-based sampling. This is because tuning of our rules is based on mostly frequent n-grams.

Error analysis shows three types of errors:

Judgment inconsistency Human judgment disagrees in some ambiguous cases such as “この こと” (this thing). This is mainly caused by different rigor of annotators, namely, some annotator is too rigid and another is too loose. Average judgment disagreement rate between all pairs of annotators was 7.1% about token-based sampling.

Dictionary	Phrase		N-gram	
Sampling	Token	Type	Token	Type
Precision	0.90	0.85	0.41	0.37
Recall	0.81	0.51	1.00	1.00
F-measure	0.85	0.63	0.58	0.53

Table 4: Evaluation Result

Morphological analysis error Errors of word segmentation or POS tagging cause problems. For example, “ありがトン” (informal “thank you”) is split into “あり が トン” (ant is ton) and removed erroneously.

Lack of features for additional rules There are no features such as POS tags which we can use for additional rules. For example, a necessary phrase “マリナーズ の イチロー” (Ichiro in Mariners) has the same POS tags as an unnecessary phrase “衣装 の サンタ” (Santa Claus in costume).

The effect of frequency recalculation was unclear for the small samples. However, we investigated 2-gram pattern of family and first name and discovered that about 50% of top 100 frequent personal names are predicted correctly, defeating 1-gram candidates in terms of frequency.

A simulation shows that our system enables users to save 24% of keystrokes in terms of kana input. We assumed that the system offers 10 most frequent words when users input their first 3 characters for sampled 100 words in the dictionary.

5 Conclusion

We proposed a phrase extraction system for predictive input methods, extracting necessary phrases from a large corpus. Our system adopts a post-processing approach, which enables us to easily customize our rules and filters.

Our future work is to incorporate statistical metrics such as pointwise mutual information in a n-gram and entropy of adjacent words.

Acknowledgments

Mamoru Komachi, Manabu Sassano and other colleagues improved English greatly. Noriko Hiramura provided linguistical view.

References

Jeffrey Dean and Sanjay Ghemawat. 2004. MapReduce: Simplified Data Processing on Large Clusters. *Usenix SDI*.

Yumi Ichimura, Yoshimi Saito, Kazuhiro Kimura, and Hideki Hirakawa. 2000. Kana-Kanji Conversion System with Input Support Based on Prediction. In *Proceedings of the 18th Conference on Computational Linguistics*, volume 1, pages 341–347. Association for Computational Linguistics.

Hiroyuki Komatsu, Satoru Takabayashi, and Toshiyuki Masui. 2005. Corpus-based Predictive Text Input. In *Active Media Technology, 2005.(AMT 2005). Proceedings of the 2005 International Conference on*, pages 75–80. IEEE.

Taku Kudo, Hiroyuki Komatsu, Toshiyuki Hanaoka, Jun Mukai, and Yusuke Tabata. 2011. Mozc: Statistical Kana to Kanji Conversion System (統計的かな漢字変換システム Mozc in Japanese). In *Proceedings of the Seventeenth Annual Meeting of the Association for Natural Language Processing*, pages 948–951. The Association for Natural Language Processing.

I. Scott MacKenzie and Kumiko Tanaka-Ishii. 2007. *Text Entry Systems: Mobility, Accessibility, Universality*. Morgan Kaufmann.

Christopher D. Manning and Hinrich Schütze, 1999. *Foundations of Statistical Natural Language Processing*, chapter 5. MIT Press.

Toshiyuki Masui. 1999. POBox: An Efficient Text Input Method for Handheld and Ubiquitous Computers. In *Handheld and Ubiquitous Computing*, pages 289–300. Springer.

Yoh Okuno and Masafumi Hagiwara. 2009. Japanese Input Method based on the Internet. In *Information Processing Society of Japan and Special Interest Group on Natural Language (IPSJ-SIGNL)*, volume 2009-NL-190, pages 1–6.

Erik F. Tjong Kim Sang and Sabine Buchholz. 2000. Introduction to the CoNLL-2000 Shared Task: Chunking. In *Proceedings of the 2nd Workshop on Learning Language in Logic and the 4th Conference on Computational Natural Language Learning*, pages 13–14.

Keh-Yih Su, Ming-Wen Wu, and Jing-Shin Chang. 1994. A Corpus-based Approach to Automatic Compound Extraction. In *Proceedings of the 32nd Annual Meeting on Association for Computational Linguistics*, pages 242–247. Association for Computational Linguistics.

Yuya Unno and Yuta Tsuboi. 2011. Domain-Dependent Input Support based on Frequent Context (頻出文脈に基づく分野依存入力支援 in Japanese). In *Proceedings of the Seventeenth Annual Meeting of the Association for Natural Language Processing*, pages 1107–1110. The Association for Natural Language Processing.

James Liu Wan Yin Li, Qin Lu. 2006. TCtract-A Collocation Extraction Approach for Noun Phrases Using Shallow Parsing Rules and Statistic Models. In *Proceedings of the 20th Pacific Asia Conference on Language, Information and Computation*.