

# Discovering Characteristic Expressions from Literary Works: a New Text Analysis Method beyond *N*-Gram Statistics and KWIC

Masayuki Takeda<sup>1</sup>, Tetsuya Matsumoto<sup>1</sup>, Tomoko Fukuda<sup>2</sup>, and Ichirō Nanri<sup>3</sup>

<sup>1</sup> Department of Informatics, Kyushu University 33, Fukuoka 812-8581, Japan

<sup>2</sup> Fukuoka Jo Gakuin College, Ogōri 838-0141, Japan

<sup>3</sup> Junshin Women's Junior College, Fukuoka 815-0036, Japan

{takeda, tetsuya}@i.kyushu-u.ac.jp

{tomoko-f@muc, nanri-i@msj}.biglobe.ne.jp

**Abstract.** We attempt to extract characteristic expressions from literary works. That is, our problem is, given literary works by a particular writer as *positive examples* and works by another writer as *negative examples*, to find expressions that appear frequently in the positive examples but do not so in the negative examples. It is considered as a special case of the *optimal pattern discovery* from textual data, in which only the *substring patterns* are considered. One reasonable approach is to create a list of substrings arranged in the descending order of their *goodness*, and to examine a first part of the list by a human expert. Since there is no word boundary in Japanese texts, a substring is often a fragment of a word or a phrase. How to assist the human expert is a key to success in discovery. In this paper, we propose (1) to restrict to the *prime* substrings in order to remove redundancy from the list, and (2) a way of browsing the neighbor of a focused string as well as its context. Using this method, we report successful results against two pairs of anthologies of classical Japanese poems. We expect that the extracted expressions will possibly lead to discovering overlooked aspects of individual poets.

## 1 Introduction

Analysis of expressions is one of the most fundamental methods in literary studies. In classical Japanese poetry Waka, there are strict rules in the choice and combination of poetic words. For instance, the word “Uguisu” (Japanese bush warbler) should be used linked with the word “Ume” (plum-blossom). It is significant in Waka studies, therefore, to consider how poets learned such rules and developed their own expressions. We would like to investigate how they learned certain expressions from their predecessors.

From this point of view, we have established a method that automatically extracts similar poems in expression from Waka database [12, 11]. Using this method, we discovered affinities of some unheeded poems with some earlier ones, and this raised an interesting issue for Waka studies and we could give a convincing conclusion to it.

- We have proved that the poem (As if it were in darkness, / My parental heart / Is blind and lost in / The ways of caring about my child) by Fujiwara-no-Kanesuke, one of the renowned thirty-six poets, was in fact based on a model poem found in *Kokinshū*. The same poem had been interpreted just to show “frank utterance of parents’ care for their child.” Our study revealed the poet’s techniques in composition half hidden by the heart-warming feature of the poem by extracting the same structure between the two poems.
- We have compared *Tametadashū*, the mysterious anthology unidentified in Japanese literary history, with a number of private anthologies edited after the middle of the Kamakura period (the thirteenth-century) using the same method, and found that there are about 10 pairs of similar poems between *Tametadashū* and *Sokonshū*, an anthology by Shōtetsu. The result suggests that the mysterious anthology was edited by a poet in the early Muromachi period (the fifteenth-century). There have been surmised dispute about the editing date since one scholar suggested the middle of Kamakura period as a probable one. We have had a strong evidence about this problem.

While continuing to find affinities among Waka poems, it is also necessary to give some additional conditions to the method when we compare poets in parent-child or teacher-student relationships. It is easily inferred that poet A (the master poet) greatly influences poet B (the disciple poet), and that the poems by poet B may have a lot of allusions to those by poet A. In fact, many scholars have already noticed such apparent literary relationships. In such cases, it is much more important to clarify the differences than to enumerate affinities. For example, when poet B hardly adopts the expressions frequently used by poet A, it will give us a chance to study their relationship in a different way. In this paper, we will compare two poets’ private anthologies and try to make clear the differences and features in the similar expressions on the basis of their frequencies. This is in fact derived from the same methodology we have been practicing so far.

Shimozono et al. [10] formulated the problem of finding *good* pattern that distinguishes two sets *Pos* and *Neg* of strings (called positive and negative examples, respectively) as an instance of *optimized pattern discovery*, originally proposed by Fukuda et al. [6]. The problem is:

**Given.** Two finite sets *Pos* and *Neg* of non-empty strings.

**Find.** A pattern  $\pi$  that minimizes a statistical measure function  $G(\pi; Pos, Neg)$ .

For instance, the *classification error*, the *information entropy*, the *Gini index*, and the  $\chi^2$  *index* are used as a statistical measure function  $G$ . Note that these measures are functions of 4-tuple  $(p_1, n_1, p_0, n_0)$ , where  $p_1$  and  $n_1$  are the numbers of strings in *Pos* and *Neg* that contain  $\pi$ , respectively, and  $p_0 = |Pos| - p_1$  and  $n_0 = |Neg| - n_1$ . Namely, the goodness of a pattern depends only on its frequencies in *Pos* and *Neg*. Shimozono et al. [10] showed an efficient algorithm for *proximity word-association patterns*. Although only the classification error measure is dealt with in [10], the algorithm works for many other measures [1].

Our problem is regarded as a special case of this problem. That is, we deal with only the patterns of the form  $*w*$ , where  $*$  is a wildcard that matches any string and  $w$  is a non-empty string. We call such patterns the *substring patterns*. We have a trivial  $O(m)$  time and space algorithm since there exist essentially  $O(m)$  candidates for the best substring  $w$ , where  $m$  is the total length of the strings in  $S = Pos \cup Neg$ , and therefore difficulty lies mainly on how to find an appropriate goodness measure.

However, it seems that no matter what measure we use, most of ‘good’ patterns are not so good in practice; they are obvious and worthless in many cases. For this reason, discovery requires an effort by domain experts to examine an upper part of list of patterns arranged in the decreasing order of the goodness. This corresponds to the step of *interpreting mined patterns*, a ‘postprocessing’ of data mining in knowledge discovery process [5]. We believe that how to support in this step the domain experts is a key to success. In this paper we tackle this problem with the weapon of *stringology*.

Let  $Sub(S)$  be the set of substrings of strings in  $S$ . All the strings in  $Sub(S)$  could be candidates for characteristic expressions. The candidate strings, however, are not independent each other in the sense that some strings subsume other ones. Moreover, we are frequently faced with the case that two strings in the superstring–substring relation have the same frequency, and therefore have the same value of goodness. (Recall that the goodness measures considered in this paper depend only on the frequencies in positive and negative examples.) For instance, in the first eight imperial anthologies (from Kokinshū to Shinkokinshū), every occurrence of the string “SHI-NO-NO-YA” (consisting of 4 syllables) is a substring of “YO-SHI-NO-NO-YA-MA” (6 syllables; Mt. Yoshino). In such case we want to remove shorter strings.

On the other hand, researchers are interested in not just frequency of a word but in its actual use. That is, they would access the context of each word appearance. In addition, a candidate string is often a fragment of a word or a phrase and seems to be meaningless. In order to find the word or the phrase that contains this fragment as a substring, the researchers need to check what string immediately precedes (follows) this fragment, for every occurrence of it. Thus, it is necessary for the researchers to see the possible superstrings of a focused candidate string.

In order to introduce a structure into  $Sub(S)$ , we use the equivalence relation, first defined by Blumer et al. [2], which has the following properties:

- Each equivalence class has a unique longest string which contains any other member as a substring, and we regard it as the representative of the class.
- All the strings in an equivalence class have the same frequency in  $S$  (and therefore have the same value of goodness.)
- The number of equivalence classes is linear with respect to the total length of the strings in  $S$ .

A string in  $Sub(S)$  is said to be *prime* if it is the representative of some equivalence class under this equivalence relation. The basic idea is to consider only the prime substrings as candidates for characteristic expressions. Any non-prime

substring with a high goodness value can be found as a part of the prime substring equivalent to it, even though we remove all the non-prime substrings from the list.

It should be stated that the data structure called the *suffix tree* [3] exploits a similar and more popular equivalence relation, which also satisfies the above three conditions. This equivalence relation is finer than the one by Blumer et al. [2], and therefore includes much more equivalence classes. From the viewpoint of computational complexity, this is not a significant difference because the two equivalence relations both satisfy the third condition. However, the difference is crucial for researchers who must check the candidate strings. In fact the number of equivalence classes was reduced to approximately 1/4 by using the one by Blumer et al. in our experiment using Waka poems, as will be shown in Section 5.

We would create a candidate list only of prime substrings. To support the domain experts who inspect the list, we develop a kind of browser to see:

- The non-prime substrings equivalent to each prime substring.
- The superstring–substring relationships between the prime strings.

The symmetric compact DAWG [2] for  $S$  is useful for this purpose. It is a directed acyclic graph such that the vertices are the prime substrings, and the labeled edges represent: *What happens when appending a possible letter to the left or right end of a prime substring?* For instance, in the first eight imperial anthologies, appending a single letter “YA” to the right end of the string “YO-SHI-NO-NO” yields “YO-SHI-NO-NO-YA-MA.” In the case of another letter “HA,” it yields the string “MI-YO-SHI-NO-NO-HA-NA.” (Notice that not only “HA-NA” is appended to the right, but also “MI” is appended to the left.) On the other hand, the result of appending “MI” to the left of the same string is simply “MI-YO-SHI-NO-NO.”

We would consider to draw interactively a subgraph of the symmetric compact DAWG whose nodes are limited to the ones reachable from/to a focused node (namely, they are substrings or superstrings of the focused string). This subgraph, however, is rather complicated for a reasonable size  $S$  in the real world. For this reason, we shall instead draw:

- The subgraph consisting of the prime substrings that are substrings of the focused one. (It is small enough because the focused string is rather short in practice.)
- The right and left context trees which represent the same information as the subgraph consisting of the prime substrings that are superstrings of the focused one.

The *right context tree* of a prime substring  $x$  is essentially the same as the subtree of the node  $x$  in the suffix tree [3] for  $S$ , but augmented by adding to every node  $xy$  a label of “ $\gamma[x]y$ ” such that  $\gamma xy$  is the prime substring that is equivalent to  $xy$ . The left context tree is defined in a similar way.

Most of the existing text analysis tools have the KWIC (Key Word In Context) display [8]. The left and the right context trees of a focused prime substring enable researchers to grasp the context of the string more quickly compared with

KWIC, especially when the number of appearances of the string being checked is relatively large.

Using the suggested method, we extracted the strings that differentiate two anthologies, scholarly scrutinized them, and succeeded in finding the characteristic expressions. We compared Sankashū by Saigyō with Shūgyokushū by Jien, and Shūigūsō by Fujiwara-no-Teika with Tameieshū by his son, Fujiwara-no-Tameie separately, and procured such expressions highlighting differences between each two anthologies.

Saigyō and Jien were both priests, but their lives were so contrastive in status and social circumstances. Being a son of the famous regent, Jien could not make a break with the government. However, it is well known that Jien sought much help from Saigyō in his last days not only about poetic composition but also about the way of life. For instance, he confessed his plan of becoming a hermit.

On the other hand, Tameie was rigorously trained by his own father, Teika, for he was in the direct descent of the Mikohidari dynasty famous for producing poets. The number of Teika's poems Tameie referred to is not small.

In each pair of anthologies, there necessarily exist similar poems. However, as we have demonstrated so far, we could collect certain differences in expressions. This, we expect, will possibly lead to discovering overlooked aspects of individual poets.

It may be relevant to mention that this work is a multidisciplinary study between the literature and the computer science. In fact, the second author from the last is a Waka researcher and the last author is a linguist in Japanese language.

## 2 Substring Statistics for Japanese Literary Studies

If we want to use a word statistics, we need to perform a task of word segmentation, because there is no word boundary in texts written in Japanese, like other non-Western languages. This is a difficult and time-consuming task. For example, it is reported in [9] that it took eight years to build a part-of-speech tagged corpus of “the Tale of Genji.” In the case of Waka poetry, building such corpora is more difficult because the frequently used technique, called “Kake-kotoba,”<sup>1</sup> which exploits the ambiguities of a word or part of word. We can say that a unique word segmentation is essentially impossible in such a situation.

Recently some researchers noticed that using substring statistics instead of word statistics is in fact useful in expression analysis, although a substring could be a fragment of a word or a phrase. For example, Kondo [7] proposed an expression analysis method based on  $n$ -gram statistics, and reported some differences between the expressions used by poets and poetess in Kokinshū, the most famous imperial anthology. The  $n$ -gram statistics is essentially the same as

---

<sup>1</sup> A sort of homonymic punning where the double meaning of a word or part of word is exploited. In English it is usually called *the pivot words* because it is used as a pivot between two series of sounds with overlapping syntactical and semantic patterns.

the substring statistics since  $n$  cannot be fixed. This work opened the door for the application of substring statistics to Japanese literary works.

However, Kondo restricted herself to the substrings which (1) are of length from 3 to 7, (2) occur more than once, and (3) are used only by male (not used by female), to ease the burden. In this paper, we will show that such a restriction can be removed by exploiting combinatorial properties on strings.

### 3 Prime Substrings

In this section, we give a formal definition of the prime substrings, and present some of their properties.

#### 3.1 Preliminary

Let  $\Sigma$  be a finite alphabet. An element of  $\Sigma^*$  is called a *string*. Strings  $x$ ,  $y$ , and  $z$  are said to be a *prefix*, *substring*, and *suffix* of the string  $u = xyz$ , respectively. A string  $u$  is said to be a *superstring* of a string  $y$  if  $y$  is a substring of  $u$ . The length of a string  $u$  is denoted by  $|u|$ . The empty string is denoted by  $\varepsilon$ , that is,  $|\varepsilon| = 0$ . Let  $\Sigma^+ = \Sigma^* - \{\varepsilon\}$ . The  $i$ th symbol of a string  $u$  is denoted by  $u[i]$  for  $1 \leq i \leq |u|$ , and the substring of a string  $u$  that begins at position  $i$  and ends at position  $j$  is denoted by  $u[i : j]$  for  $1 \leq i \leq j \leq |u|$ . For convenience, let  $u[i : j] = \varepsilon$  for  $j < i$ . Let  $Sub(w)$  denote the set of substrings of  $w$ . Let  $Sub(S) = \bigcup_{w \in S} Sub(w)$  for a set  $S$  of strings. For a set  $S$  of strings, denote by  $\|S\|$  the total length of the strings in  $S$ , and denote by  $|S|$  the cardinality of  $S$ .

#### 3.2 Definition of Prime Substrings

**Definition 1.** Let  $S$  be a non-empty finite subset of  $\Sigma^+$ . For any  $x$  in  $Sub(S)$ , let

$$\begin{aligned} \text{Beginpos}_S(x) &= \{\langle w, j \rangle \mid w \in S, 0 \leq j \leq |w|, x = w[j + 1 : j + |x|]\}, \\ \text{Endpos}_S(x) &= \{\langle w, j \rangle \mid w \in S, 0 \leq j \leq |w|, x = w[j - |x| + 1 : j]\}. \end{aligned}$$

For any  $x \notin Sub(S)$ , let  $\text{Beginpos}_S(x) = \text{Endpos}_S(x) = \emptyset$ .

For example, if  $S = \{babc, ababb\}$ , then  $\text{Beginpos}_S(a) = \text{Beginpos}_S(ab) = \{\langle babc, 1 \rangle, \langle ababb, 0 \rangle, \langle ababb, 2 \rangle\}$ ,  $\text{Beginpos}_S(c) = \{\langle babc, 4 \rangle\}$ , and  $\text{Endpos}_S(bb) = \text{Endpos}_S(abb) = \text{Endpos}_S(babb) = \{\langle babc, 4 \rangle, \langle ababb, 5 \rangle\}$ .

From here on, we omit the set  $S$ , and write simply as  $\text{Beginpos}$  and  $\text{Endpos}$ .

**Definition 2.** Let  $x$  and  $y$  be any strings in  $\Sigma^*$ . We write as  $x \equiv_L y$  if  $\text{Beginpos}(x) = \text{Beginpos}(y)$ , and write as  $x \equiv_R y$  if  $\text{Endpos}(x) = \text{Endpos}(y)$ . The equivalence class of a string  $x \in \Sigma^*$  with respect to  $\equiv_L$  (resp.  $\equiv_R$ ) is denoted by  $[x]_{\equiv_L}$  (resp.  $[x]_{\equiv_R}$ ).

If  $S = \{babbc, ababb\}$ , then  $[\varepsilon]_{\equiv_L} = [\varepsilon]_{\equiv_R} = \{\varepsilon\}$ ,  $[a]_{\equiv_L} = \{a, ab\}$ ,  $[bb]_{\equiv_R} = \{bb, abb, babb\}$ , and  $[c]_{\equiv_R} = \{c, bc, bbc, abbc, babbc\}$ .

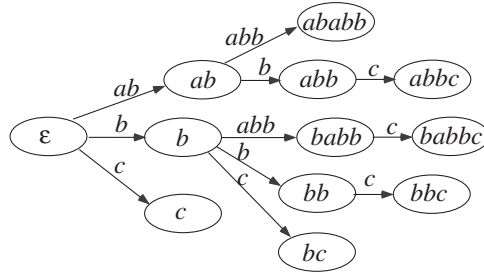
Note that all strings that are not in  $Sub(S)$  form one equivalence class under  $\equiv_L$  ( $\equiv_R$ ). This equivalence class called the *degenerate* class. All other classes are called *nondegenerate*.

It follows from the definition of  $\equiv_L$  that, if  $x$  and  $y$  are strings in the same nondegenerate class under  $\equiv_L$ , then either  $x$  is a suffix of  $y$ , or vice versa. Therefore, each nondegenerate equivalence class in  $\equiv_L$  has a unique longest member. Similar discussion holds for  $\equiv_R$ .

**Definition 3.** For any string  $x$  in  $Sub(S)$ , let  $\overleftarrow{x}$  and  $\overrightarrow{x}$  denote the unique longest members of  $[x]_{\equiv_L}$  and  $[x]_{\equiv_R}$ , respectively.

For any string  $x$  in  $Sub(S)$ , there uniquely exist strings  $\alpha$  and  $\beta$  such that  $\overleftarrow{x} = \alpha x$  and  $\overrightarrow{x} = x\beta$ . In the running example, we have  $\overleftarrow{\varepsilon} = \overrightarrow{\varepsilon} = \varepsilon$ ,  $\overleftarrow{a} = ab$ ,  $\overleftarrow{b} = b$ ,  $\overleftarrow{bb} = babb$ ,  $\overleftarrow{bab} = babbc$ , and  $\overleftarrow{c} = babbc$ .

Figure 1 shows the suffix tree [3] for  $S = \{babbc, ababb\}$ . Note that the nodes of the suffix tree are the strings with  $x = \overrightarrow{x}$ . On the other hand, Fig. 2 shows



**Fig. 1.** Suffix tree for  $S = \{babbc, ababb\}$ .

the directed acyclic word graph (DAWG for short) [3] for  $S$ . Note that the nodes are the nondegenerate equivalence classes in  $\equiv_R$ . The DAWG is the smallest automaton that recognizes the set of suffixes of the strings of  $S$  if we designate some nodes as *final states* appropriately.

**Definition 4.** For any string  $x$  in  $Sub(S)$ , let  $\overleftarrow{x}$  be the string  $\alpha x\beta$  such that  $\alpha$  and  $\beta$  are the strings satisfying  $\overleftarrow{x} = \alpha x$  and  $\overrightarrow{x} = x\beta$ .

In the running example,  $\overleftarrow{\varepsilon} = \varepsilon$ ,  $\overleftarrow{a} = ab$ ,  $\overleftarrow{b} = b$ ,  $\overleftarrow{ab} = ab$ ,  $\overleftarrow{abb} = babb$ ,  $\overleftarrow{bab} = babbc$ , and  $\overleftarrow{c} = babbc$ .

**Definition 5.** Strings  $x$  and  $y$  are said to be equivalent on  $S$  if and only if

1.  $x \notin Sub(S)$  and  $y \notin Sub(S)$ , or
2.  $x, y \in Sub(S)$  and  $\overleftarrow{x} = \overleftarrow{y}$ .

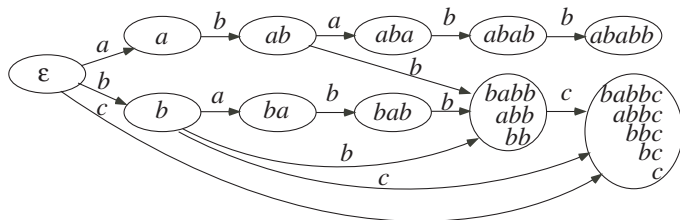


Fig. 2. DAWG for  $S = \{babbc, ababb\}$ .

This equivalence relation is denoted by  $x \equiv y$ . The equivalence class of  $x$  under  $\equiv$  is denoted by  $[x]_{\equiv}$ .

Notice that, for any string  $x$  in  $Sub(S)$ , the string  $\overleftarrow{x}$  is the longest member of  $[x]_{\equiv}$ . Intuitively,  $\overleftarrow{x} = \alpha x \beta$  means that:

- Every time  $x$  occurs in  $S$ , it is preceded by  $\alpha$  and followed by  $\beta$ .
- Strings  $\alpha$  and  $\beta$  are as long as possible.

Now, we are ready to define the prime substrings.

**Definition 6.** A string  $x$  in  $Sub(S)$  is said to be prime if  $\overleftarrow{x} = x$ .

**Lemma 1 (Blumer et al. (1987)).** The equivalence relation  $\equiv$  is the transitive closure of the relation  $\equiv_R \cup \equiv_L$ .

It follows from the above lemma that  $\overleftarrow{x} = \overleftarrow{\overleftarrow{x}} = \overleftarrow{\overleftarrow{\overleftarrow{x}}}$  for any string  $x$  in  $Sub(S)$ .

### 3.3 Properties of Prime Substrings

Recall that the number of all substrings of  $S$  is  $O(\|S\|^2)$ . This can be reduced to  $O(\|S\|)$  by considering the substrings  $x$  such that  $\overleftarrow{x} = x$ . In fact the suffix tree is a data structure that exploits this property. Similarly, the DAWG achieves its  $O(\|S\|)$  space complexity by identifying every substring  $x$  with the substring  $\overleftarrow{x}$ . Since the number of prime substrings of  $S$  is also  $O(\|S\|)$ , it seems that there is no advantage in considering only the prime substrings. In practical application, however, it has a big advantage because the users do not have to examine non-prime substrings. The next lemma gives more tight bounds.

**Lemma 2 (Blumer et al. (1987)).** Assume  $\|S\| > 1$ . The number of the nondegenerate equivalence classes in  $\equiv_L$  ( $\equiv_R$ ) is at most  $2\|S\| - 1$ . The number of the nondegenerate equivalence classes in  $\equiv$  is at most  $\|S\| + |S|$ .

Thus, the number of prime substrings of  $S$  is at most  $\|S\| + |S|$ . In practice, the number of prime substrings is usually smaller than both the number of substrings  $x$  with  $\overleftarrow{x} = x$  and the number of substrings  $x$  with  $\overleftarrow{\overleftarrow{x}} = x$ , which are upper-bounded by  $2\|S\| - 1$ .

Let  $Prime(S)$  be the set of prime substrings of  $S$ , i.e.  $Prime(S) = \{\overleftarrow{x} \mid x \in Sub(S)\}$ .

**Definition 7.** The symmetric compact DAWG for  $S$  is the triple  $(V, E_L, E_R)$  where  $V = \text{Prime}(S)$  is the set of vertices, and  $E_L, E_R \subseteq V \times V \times \Sigma^+$  are two kinds of labeled edges defined by:

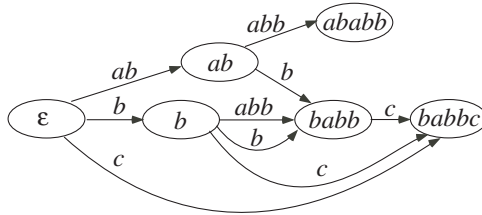
$$E_L = \{(x, \gamma\sigma x\delta, \gamma\sigma) \mid x \in V, \sigma \in \Sigma, \gamma, \delta \in \Sigma^*, \gamma\sigma x\delta = \overleftarrow{\sigma x}\}$$

$$E_R = \{(x, \delta x\sigma\gamma, \sigma\gamma) \mid x \in V, \sigma \in \Sigma, \delta, \gamma \in \Sigma^*, \delta x\sigma\gamma = \overrightarrow{x\sigma}\}.$$

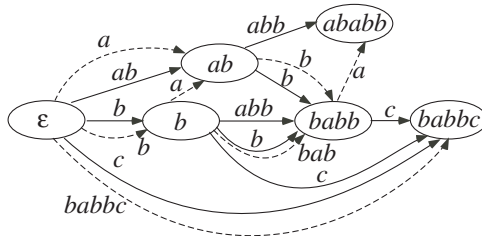
The compact DAWG for  $S$  is the graph  $(V, E_R)$ .

Figure 3 show the compact DAWG for the running example. Compared with the suffix tree of Fig. 1 and with the DAWG of Fig. 2, the nodes represent only the prime substrings, and therefore it is expected that the number of nodes are smaller than those of the suffix tree and the DAWG.

Figure 4 shows the symmetric compact DAWG for the running example.



**Fig. 3.** Compact DAWG for  $S = \{babbc, ababb\}$ . The edge labeled by  $\sigma\gamma$  from the node  $x$  to the node  $y$  corresponds to the fact that  $x\sigma$  is equivalent to  $y$  under  $\equiv$ , where  $x, y \in \text{Prime}(S)$ ,  $\sigma \in \Sigma$ ,  $\gamma \in \Sigma^*$ . For instance, the arrow labeled “ $abb$ ” from the node “ $b$ ” to the node “ $babb$ ” means that  $ba$  is equivalent to  $babb$  under  $\equiv$ .



**Fig. 4.** Symmetric compact DAWG for  $S = \{babbc, ababb\}$ . The solid and the broken arrows represent the edges in  $E_R$  and  $E_L$ , respectively.

**Lemma 3 (Blumer et al. (1987)).** Both the compact DAWG and the symmetric compact DAWG for  $S$  can be constructed in  $O(\|S\|)$  time and space.

## 4 Browser for Finding Characteristic Substrings

We would find ‘good’ patterns from text strings in the following manner.

1. Choose an appropriate measure of the *goodness* of the patterns which depends only on their frequencies.
2. Compute the goodness of all possible patterns, and create a list of patterns arranged in the decreasing order.
3. Evaluate an upper part of the list by a domain expert.

However, most of patterns in the upper part of the list are not so good in practice. They are obvious and/or worthless in many cases. For this reason, it is most important to develop an effective way of supporting the domain expert in the third step.

The patterns we are dealing with are restricted to the *substring patterns*, which are of the form  $*w*$ , where  $*$  is the wildcard that matches any string in  $\Sigma^*$  and  $w$  is a non-empty string in  $\Sigma^+$ . Since we assume that the goodness of a substring pattern depends only on its frequencies, we can restrict  $w$  to a prime substring. Thus, we exclude the non-prime substrings from the list of candidates for characteristic expressions. There is no risk of overlooking any good non-prime substring  $x$  because it must appear as a substring in the prime substring  $\overleftarrow{x}$  with the same value of goodness.

To support the expert we develop a browser to see the following information.

- Among the substrings of the focused string, which are equivalent to it?
- Among the prime substrings, which are superstrings (substrings) of the focused string?

The symmetric compact DAWG is useful for this purpose. That is, the strings equivalent to a focused prime substring  $x$  can be obtained from the incoming edges of the node  $x$ . If  $(x', x, \sigma\gamma) \in E_R$ , then any string  $z$  such that  $x'\sigma \succeq z \succeq x$  is equivalent to  $x$ , where  $u \succeq v$  means that  $u$  is a substring of  $v$ . Similarly, if  $(x', x, \gamma\sigma) \in E_L$ , any string  $z$  such that  $\sigma x' \succeq z \succeq x$  are equivalent to  $x$ . The string  $x'\sigma$  ( $\sigma x'$ ) appears exactly once within the string  $x$ , and so it is easy to grasp the strings  $z$  satisfying the inequality.

To give the domain expert an illustration of the superstring–substring relation on the prime substrings, we would consider to draw interactively the subgraph of the symmetric compact DAWG in which the nodes are restricted to the ones reachable from/to a focused node. However, the subgraph is still large and complicated for a reasonable size set  $S$  of strings in the real world. Instead, we shall draw

- The subgraph consisting of the prime substrings that are substrings of the focused one. (It is small enough because the focused string is rather short in practice.)
- The right and left context trees which represent the same information as the subgraph consisting of the prime substrings that are superstrings of the focused one. The former represents the information in  $E_R$ , and the latter in  $E_L$ .

The *right context tree* of a prime substring  $x$  is essentially the same as the subtree of the node  $x$  in the suffix tree [3] for  $S$ , but augmented by adding to every node  $xy$ ,  $xy = \overrightarrow{xy}$  ( $y \in \Sigma^*$ ), a label “ $\gamma[x]y$ ” such that  $\gamma xy = \overrightarrow{xy}$ . (The left context tree is defined in a similar way.) There may be two nodes  $xy_1$  and  $xy_2$  such that  $y_1 \neq y_2$  but  $\overrightarrow{xy_1} = \overrightarrow{xy_2}$ . Therefore more than one node may have the same label if ignoring the square brackets ( $[, ]$ ). Figure 5 shows the left and the right context trees of  $x = b$  for  $S = \{babbc, ababb\}$ .

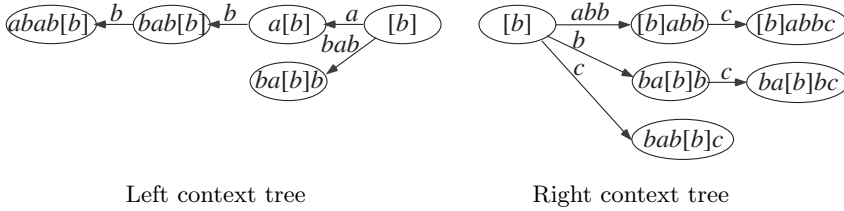


Fig. 5. Left and right context trees of  $x = b$  for  $S = \{babbc, ababb\}$ .

## 5 Experimental Results

We carried out experiments for Waka poems and prose texts.

### 5.1 Goodness Measures

The statistical measures like the *classification error*, the *information entropy*, and the *Gini index* are abstracted as follows [4]. Let  $p_1$  and  $n_1$  denote the numbers of strings in *Pos* and *Neg* that contain  $\pi$ , respectively, and let  $p_0 = |\text{Pos}| - p_1$  and  $n_0 = |\text{Neg}| - n_1$ . Let

$$f(p_1, n_1, p_0, n_0) = \frac{p_1 + n_1}{N} \cdot \psi\left(\frac{p_1}{p_1 + n_1}\right) + \frac{p_0 + n_0}{N} \cdot \psi\left(\frac{p_0}{p_0 + n_0}\right),$$

where  $N = p_1 + n_1 + p_0 + n_0$  and  $\psi$  is a non-negative function with the following properties:

- $\psi(1/2) \geq \psi(r)$  for any  $r \in [0, 1]$ .
- $\psi(0) = \psi(1) = 0$ .
- $\psi(r)$  increases in  $r$  on  $[0, 1/2]$  and decreases in  $r$  on  $[1/2, 1]$ .

The information entropy measure is the function  $f$  such that

$$\psi(r) = -r \log r - (1 - r) \log(1 - r).$$

The classification error and the Gini index measures are also obtained by letting  $\psi(r) = \min(r, 1 - r)$  and  $\psi(r) = 2r(1 - r)$ , respectively.

In our experiment, we used the information entropy measure. We also tested the classification error and the Gini index, but no substantial differences were observed.

## 5.2 Text Strings We Used

Our experiments were performed against the following classical Japanese literary works.

- (A) Two private anthologies Sankashū by the priest Saigyō (1118–1190), and Shūgyokushū by the priest Jien (1155–1225). It is well-known that Saigyō was a great influence on Jien. In fact, Jien composed many poems using similar expressions preferred by Saigyō.
- (B) Two private anthologies Shūigusō by Fujiwara no Teika (1162–1241), and Tameieshū by Fujiwara no Tameie (1198–1275). Teika is a poet and a literary theorist, who ranks among the greatest of Waka poets. The poems of his son Tameie were influenced by the poems and the theory of Teika.
- (C) The Tale of Genji, written by Murasaki Shikibu (Lady Murasaki), which is considered the first novel ever written. It consists of 54 chapters. Some scholars have convinced that the Tale of Genji is not all by the same writer. Especially, it is often claimed that the author of the main chapters and the author(s) of the Uji chapters (the last 10 chapters) are not the same person. The aim is to compare the last 10 chapters with the other 44 chapters.

Table 1 shows the numbers of nondegenerate equivalence classes for the text strings of (A), (B), and (C), under the three equivalence relations  $\equiv_L$ ,  $\equiv_R$ , and  $\equiv$ . The result imply that the limitation to the prime substrings drastically reduce the number of the substrings to be examined by human experts. Thus, the notion of the prime substrings makes the substring statistics based text analysis be realistic.

**Table 1.** Comparison of the numbers of nondegenerate equivalence classes against the three equivalence relations,  $\equiv_L$ ,  $\equiv_R$ , and  $\equiv$ .

Anthologies		S	S	Sub(S)	# nondegen. equiv. classes		
Pos	Neg				$\equiv_L$	$\equiv_R$	$\equiv$
Sankashū (1,552 poems)	Shūgyokushū (5,803 poems)	7,355	229,728	2,817,436	259,576	265,238	65,149
Shūigusō (2,985 poems)	Tameieshū (2,101 poems)	5,086	158,290	1,989,446	183,358	185,987	46,288
Tale of Genji (Chap. 1–44)   (Chap. 45–54)		54	859,796	1,493,709,707	1,182,601	1,181,439	251,343

## 5.3 Characteristic Expressions Extracted

It should be noted that a Waka poem consists of five lines, and we can restrict to the substrings of lines of Waka poems. In order to exclude the substrings which stretch over two or more lines, we let  $S$  be the set of lines of Waka poems in the anthologies. Then, we have 50,345 and 37,477 prime substrings to be examined

for (A) Sankashū and Shūgyokushū, and for (B) Shūigushō and Tameieshū, respectively.

On the other hand, there are no punctuations in the texts of “the Tale of Genji” we used. We have 54 text strings each of which corresponds to a chapter of the book. For this reason, we define  $p_1$  ( $n_1$ ) to be the number of occurrences of pattern in  $Pos$  ( $Neg$ ), not to be the number of strings that contain it.

We created lists of prime substrings arranged using the information entropy measure. Table 2 shows the best 40 prime substrings for (A) Sankashū and Shūgyokushū, and (B) Shūigushō and Tameieshū. (We omitted the list for “the Tale of Genji.”) From the upper part of the list, we can notice the following, by using a prototype of our browser.

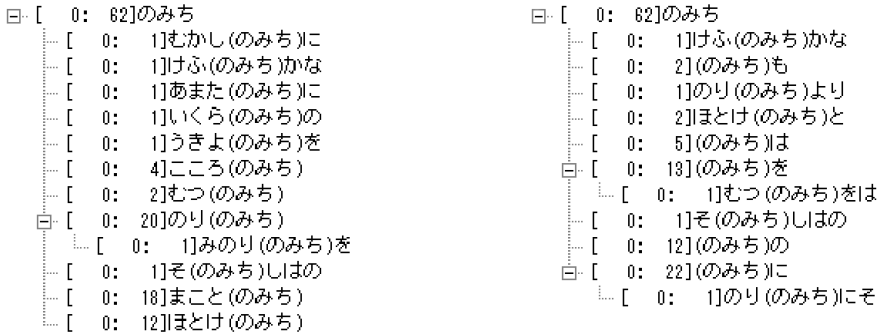
- While Shūgyokushū has Buddhist terms like “Nori-no-michi” (the road of dharma), “Nori-no-hana” (the flower of dharma), and “Makoto-no-michi” (the road of truth) in 20, 16, and 18 poems, respectively, there is no such expression in Sankashū. It should be mentioned that the first two terms were obtained by browsing the right context tree of the string “NO-RI-NO,” which ranks the 15th, and the last one was obtained from the left context tree of the string “NO-MI-CHI,” which ranks the 24th. See Fig. 6.
- There are 21 examples of “...wo Ikanisemu” (most of them are “Mi wo Ikanisemu”) in Shūgyokushū and there is none in Sankashū. This is interesting and seems to suggest their differences in their beliefs in Buddhism and in their ways of lives. The expression “wo Ikanisemu” was obtained from the left context tree of the string “I-KA-NI-SE,” which ranks the 31st.
- In Tameieshū, there are many expressions using “Oi-no-nezame” (wakeful night for the aged) and “Oi-no-Namida” (tears of the aged), but there is not such in Shūigushō at all. Though Tameie was conscious of old age in his poems, he did not live longer than his father had done. (Teika died at 80 and Tameie at 78.) Surveying Shinpen-Kokkatakai, a collection of 1,162 anthologies of Waka poems (about 450,000 poems in total), we find that the expressions “Oi-no-mezame” and “Oi-no-Namida” most frequently appear in Tameieshū. These expressions, therefore, definitely characterize Tameie’s poetry. In his last days, he was involved in the family feud about his successor. It was resulted in dividing the dynasty into three as Nijō, Kyōgoku, and Reizei in the next generation. This is quite a contrast to the case of Teika, who could decide Tameie as his only one successor.
- In the narrative or the characters’ speech in the Uji chapters of the Tale of Genji, we observed some characteristic expressions like “Ikanimo-ikanimo,” which ranks the 90th, and “Shikiwazakana,” which ranks the 108th. But most of the strings collected from the book are proper nouns like characters’ names, titles and place names, and they largely depend on the story settings and the characters. So we have to say that this is far from discovery. We could have removed such strings under conditions either of  $p_1 = 0$  or of  $n_1 = 0$ , but there was risk of excluding some other important strings too. We need to consider how to adapt a filtering method to prose works.

**Table 2.** Best 40 prime substrings from two pairs of private anthologies. The hyphens ‘-’ are inserted between syllables, each of which was written as one Kana character although romanized here.

(A) Sankashū vs. Shūgyokushū				(B) Shūigusō vs. Tameieshū					
	$G$	$P_1$	$N_1$	Substring		$G$	$P_1$	$N_1$	Substring
1	0.5115	33	397	RU-NO	1	0.6668	14	103	O-I
2	0.5115	26	351	HA-RU-NO	2	0.6685	2	63	O-I-NO
3	0.5118	16	271	MI-YO	3	0.6717	6	54	WO-KU-RA
4	0.5120	919	2822	TE	4	0.6723	11	60	WO-KU
5	0.5122	29	344	NO-SO	5	0.6731	246	74	SO-RA
6	0.5124	41	28	KO-KO-CHI	6	0.6735	4	38	NI-KE-RU
7	0.5124	44	33	KO-CHI	7	0.6736	109	168	KO-SO
8	0.5126	21	273	NO-SO-RA	8	0.6738	3	34	KU-RA-YA-MA
9	0.5127	60	495	SO-RA	9	0.6739	54	106	I-NO
10	0.5129	7	163	MI-YO-SHI	10	0.6739	3	33	WO-KU-RA-YA-MA
11	0.5131	3	120	SU-MI-YO	11	0.6740	0	23	O-I-NO-NE
12	0.5131	99	167	MA-SHI	12	0.6741	69	7	NA-KA-ME
13	0.5132	7	150	MI-YO-SHI-NO	13	0.6746	6	35	KU-RA-YA
14	0.5132	3	114	SU-MI-YO-SHI	14	0.6747	0	19	O-I-NO-NE-SA-ME
15	0.5133	7	147	NO-RI-NO	15	0.6747	88	16	SO-TE-NO
16	0.5134	487	2281	YO	16	0.6748	292	115	I-RO
17	0.5134	53	418	YU-FU	17	0.6748	57	99	NI-KE
18	0.5134	8	150	TSU-KA-SE	18	0.6752	53	6	KI-E
19	0.5135	0	67	NO-KO-RU	19	0.6752	9	36	RA-YA-MA
20	0.5135	117	722	HA-RU	20	0.6753	2	22	RA-NO-YA-MA
21	0.5136	1354	5327	NO	21	0.6753	77	114	NA-MI-TA
22	0.5136	3	101	SU-MI-YO-SHI-NO	22	0.6754	53	7	KU-YO
23	0.5136	2	90	YO-HA-NO	23	0.6754	2	21	WO-KU-RA-NO
24	0.5137	0	62	NO-MI-CHI	24	0.6754	141	173	KA-MI
25	0.5137	14	3	KA-NA-SHI-KA	25	0.6755	234	92	SO-TE
26	0.5138	1	75	KO-RU	26	0.6755	39	3	I-KU-YO
27	0.5138	9	0	TSU-TSU-MA	27	0.6755	134	166	NA-RI
28	0.5138	55	79	NI-TE	28	0.6756	42	74	KE-RE
29	0.5138	6	119	MA-TSU-KA-SE	29	0.6756	54	8	HO-HI
30	0.5138	1	73	SU-MI-NO	30	0.6756	2	20	NO-NE-SA-ME
31	0.5138	4	102	I-KA-NI-SE	31	0.6757	91	123	RI-KE
32	0.5139	37	306	YA-MA-NO	32	0.6757	47	6	NI-HO-HI
33	0.5139	152	848	KA-SE	33	0.6757	0	13	WO-KU-RA-NO-YA-MA-NO
34	0.5139	2	81	HO-TO-KE	34	0.6758	4	23	RA-NO-YA
35	0.5139	129	744	A-KI	35	0.6758	208	226	NA-SHI
36	0.5139	330	912	TSU-KI	36	0.6758	7	28	NE-SA-ME
37	0.5140	23	223	NA-HO	37	0.6758	1	16	NA-MI-TA-NA
38	0.5140	21	211	NO-RI	38	0.6758	1	16	WO-KU-RA-NO-YA-MA
39	0.5140	19	11	MI-KE	39	0.6758	133	44	TE-NO
40	0.5140	74	131	TE-NI	40	0.6759	1350	1089	SA

## 6 Concluding Remarks

We have reported successful results for Waka poems, but not for prose texts. We considered all prime substrings as candidates for characteristic expressions. However it seems that some filtering process is needed for prose texts. In [13], we successfully discovered from Waka poems characteristic patterns, called Fushi, which are regular patterns whose constant parts are restricted to sequences of auxiliary verbs and postpositional particles. To find a good filtering for prose texts will be our future work.



**Fig. 6.** Left and right context trees of “NO-MI-CHI.” In both trees the top string is “NO-MI-CHI.” In the left context tree (the left one), the 9th and 12th strings from the top are “NO-RI-NO-MI-CHI” and “MA-KO-TO-NO-MI-CHI,” respectively.

## References

1. H. Arimura. Text data mining with optimized pattern discovery. In *Proc. 17th Workshop on Machine Intelligence*, Cambridge, July 2000.
2. A. Blumer, J. Blumer, D. Haussler, R. Mcconnell, and A. Ehrenfeucht. Complete inverted files for efficient text retrieval and analysis. *J. ACM*, 34(3):578–595, 1987. Previous version in: STOC’84.
3. M. Crochemore and W. Rytter. *Text Algorithms*. Oxford University Press, 1994.
4. L. Devroye, L. Gy orfi, and G. Lugosi. *A Probablistic Theory of Pattern Recognition*. Springer, 1997.
5. U. M. Fayyad, G. P.-Shapiro, and P. Smyth. From data mining to knowledge discovery: an overview. In *Advances in Knowledge Discovery and Data Mining*, pages 1–34. The AAAI Press, 1996.
6. T. Fukuda, Y. Morimoto, S. Morishita, and T. Tokuyama. Data mining using two-dimensional optimized association rules. In *Proc. 1996 SIGMOD*, pages 13–23, 1996.
7. M. Kondo. Studies on classical Japanese literature based on string analysis using  $n$ -gram statistics. Technical report, Chiba University, March 2000. (in Japanese).
8. H. Luhn. Keyword-in-context index for technical literature (KWIC index). *American Documentation*, 11:288–295, 1960.
9. M. Murakami and Y. Imanishi. On a quantitative analysis of auxiliary verbs used in Genji Monogatari. *Transactions of Information Processing Society of Japan*, 40(3):774–782, 1999. (in Japanese).
10. S. Shimozono, H. Arimura, and S. Arikawa. Efficient discovery of optimal word-association patterns in large databases. *New Gener. Comput.*, 18(1):49–60, 2000.
11. M. Takeda, T. Fukuda, I. Nanri, M. Yamasaki, and K. Tamari. Discovering similar poems from anthologies of classical Japanese poems. *Proceedings of the Institute of Statistical Mathematics*, 48(2), 2000. to appear (in Japanese).
12. K. Tamari, M. Yamasaki, T. Kida, M. Takeda, T. Fukuda, and I. Nanri. Discovering poetic allusion in anthologies of classical Japanese poems. In *Proc. 2nd Int. Conf. Discovery Science*, LNAI 1721, pages 128–138. Springer-Verlag, 1999.
13. M. Yamasaki, M. Takeda, T. Fukuda, and I. Nanri. Discovering characteristic patterns from collections of classical Japanese poems. *New Gener. Comput.*, 18(1):61–73, 2000. Previous version in: DS’98, LNAI 1532.