A Decision Tree-based Text Art Extraction Method without any Language-Dependent Text Attribute

Tetsuya SUZUKI
Department of Electronic Information Systems
College of Systems Engineering and Science
Shibaura Institute of Technology
Minuma, Saitama city
Saitama, 337-8570, Japan
tetsuya@sic.shibaura-it.ac.jp

ABSTRACT: *Text based pictures called text art or ASCII art are often used in Web pages, email text and so on. They enrich expression in text data, but they can be noise for text processing and display of text. For example, they can be obstacle for text-to-speech software and natural language processing, and some of them lose their shape in small display devices. With Text art extraction methods, which detects the area of text art in a given text data, we can ignore text arts in text data or replace them with other strings. Because a text data may include one or more natural languages, it is desirable that text art extraction methods are language-independent. In this paper, we propose a decision tree-based text art extraction method without any language-dependent text attribute. Our method uses attributes of a given text data which represent how the text data looks like text art while previously proposed methods use attributes of a given text data which represent how the text data looks like a specific language text. We tested 63 combinations of 7 text attributes including language-dependent attributes and language-independent attributes for text art recognition. The results shows that a combination of language-independent attributes is the best for text art recognition. The attributes are an attribute based on data compression ratio by Run Length Encoding and two text attributes based on text size. We also evaluated the performance of our text art extraction method with the language-independent attributes by an experiment.*

Keywords: Text processing, Automatic detection, Pattern recognition, Machine learning, Decision tree, C4.5

Received: 11 September 2009, Revised 9 October 2009, Accepted 13 October 2009

1. Introduction

Text based pictures called *text art* are often used in Web pages, email text and so on. They are also called *ASCII art* if the art consists of ASCII code only. For example, ':-)' is a very simple text art, which can be embedded in lines. Fig.1 shows a line-oriented text art, which is a cat-like character who stands around with smile. Like these examples, they enrich expression in text data.

Text art, however, causes problems for text processing and display of text. For example, they are noises for text-to-speech software and natural language processing. Because a text-to-speech software can not ignore text arts in a given text data and pronounces digits and some symbols scattering in the text arts, the speech confuses us. Another example is that some text arts lose their shape in small display devices and we can not recognize them as text arts.

Such problems can be solved by *text art extraction methods*, which detect the area of text art in a given text data. Text art extraction methods can be constructed by *text art recognition methods*, which tell if a given fragment of text data is a text art or not. With a text art extraction method, we can ignore text arts in text data or replace them with other strings. It is desirable that text art extraction methods are language-independent because a text data may include one or more natural languages.

The previous methods of text art recognition and text art extraction [2, 3, 5] are language-dependent. Each of them uses attributes of a given text data which represent how the text data looks like a specific language text. Such strategy will not work for other languages.

We propose a text art extraction method without any language-dependent text attribute [8, 9]. Our method uses attributes of a given text data which represent how the text data looks like text art. In that sense, our method is language-independent.

The rest of the paper consists as follows. In section 2, we explain related work. In section 3, we explain our text art extraction method. In section 4, we introduce text attributes used by our text art extraction method. In section 5 and section 6, we explain our experiments: a text art recognition test and a text art extraction test. In section 7, we discuss our method and our future work. We finally state our conclusion in section 8.

Figure 1. A line-oriented text art

2. Related Work

We introduce related work about text art recognition and text art extraction. The methods are language-dependent as follows.

2.1 AA scan

A software called "AA scan" [2] recognizes articles with text art in a Japanese BBS on the Web, and is freely distributed though the Web. Though the author does not disclose the detail of the recognition method, he describes that the recognition method is specific to Japanese language in its document. For example, it uses occurrence rates of characters in text data which include not only the English alphabet but also Japanese characters such as Hiragana, Katakana and Kanji.

2.2 A Support Vector Machine-based Text Art Recognition Method

Tanioka, et al. proposed a support vector machine(SVM)-based text art recognition method [3]. Training data for SVM is a set of 262 dimension vectors. Each vector consists of two parts. The first 256 elements of the vector represent a *byte pattern*, whose i-th ($0 \leq i \leq 255$) element is the occurrence number of the byte data i in the byte stream of UTF-8 text data. The authors categorized Japanese parts of speech into 6 groups. The rest 6 elements of the vector represent the occurrence numbers of the groups in text data. Because it is specific to Japanese language, this method will not work well for other languages.

2.3 A Support Vector Machine-based Text Art Extraction Method

Nakazawa, et al. proposed a SVM-based text art extraction method [5]. The method detects parts of text art in text data by lines with byte patterns. The authors do not mention if the method is specific to a language or not in [5]. The method, however, would be language-dependent because byte patterns in training data for SVM depend on natural languages in the training data.

3. A Text Art Extraction Method

In this section, we explain our text art extraction method which deals with line-oriented text arts. It needs a text art recognition machine constructed by machine learning as its component. Whether the resulting text art extraction method is specific to a language or not depends on the component.

We consider a text data T with n lines as a sequence of lines $(l_0, l_1, ..., l_{n-1})$ where each l_i is a sequence of UTF-8 byte data. We represent the area of T from the i-th line to j-th line as $T[i, j]$ $(0 \leq i \leq j \leq n-1)$. In addition, we represent an empty string as $\in$.

In the following, we first explain three parts of our extraction method. The three parts are a procedure called scanning with window width k, a text art recognition machine, and a procedure called text area reduction. We then explain our text art extraction method.

3.1 Scanning with Window Width *k*

We define a procedure called *scanning with window width k*. Given a text data T , we watch successive k lines on T and move the area from the beginning to the end of T . We call the successive k lines as *a window*, and call the k as *the window width*.

The procedure of scanning with window width k is as follows. The input for the procedure are a text data T consisting of n lines $(l_0,l_1,...,l_{n-1})$ and a procedure P applied to windows. The output is a set of all the output of the procedure P .

1. $T' \leftarrow (\in_0,..., \in_{k-2},l_0,l_1,...,l_{n-1}, \in_0,..., \in_{k-2})$

2. $i \leftarrow 0$

3. We remove empty lines from the window $T'[i, i + k - 1]$, and apply the procedure P to the resulting window.

4. $i \leftarrow i + 1$

5. go to the step 3 if $i < n + k - 1$.

6. We output all the output of the procedure P .

3.2 A Text Art Recognition Machine

We use a text art recognition machine constructed by a machine learning algorithm. Its input is a set of text attributes and its output is whether true or false. The true value and the false value denote that the text is a text art or not respectively.

We represent a set of attributes extracted by scanning with window width k as A_k, and a recognition machine constructed by a machine learning algorithm with an attribute set A as M_A. For example, a recognition machine $M \cup_{i=1}^{w} A_i$ is a machine for text data with at most w lines.

3.3 Text Area Reduction

We define a procedure called *text area reduction* as follows. The input for the procedure are a text data T , its area $T[i, j]$ and a text art recognition machine M. The output is a part of the area $T[i, j]$.

1. We mark the following lines.
   - empty lines in $T[i, j]$
   - lines each of which M does not recognize as text art.
2. We remove the following lines from $T[i, j]$.
   - lines which are successively marked from the beginning of $T[i, j]$
   - lines which are successively marked from the end of $T[i, j]$
3. We output the resulting text area $T[i', j']$ as the result.

3.4 A Text Art Extraction Method

We define a text art extraction method with window width w. Given a text T and a text art recognition machine $M \cup_{i=1}^{w} A_i$, it works as follows.

1. We apply scanning with window width w to the text T where we recognize text art in the windows by $M \cup_{i=1}^{w} A_i$. It means that we use $M \cup_{i=1}^{w} A_i$ as the procedure P in the scanning.
2. For each chunk of successive windows in which text data has been recognized as text art, we record the text data in the chunk of the windows as a text art area candidate.
3. For each text art area candidate, we apply the text area reduction procedure with $M \cup_{i=1}^{w} A_i$.
4. We output the results of the text area reduction procedure as a set of text arts.

Figure 2. An input text for our extraction method

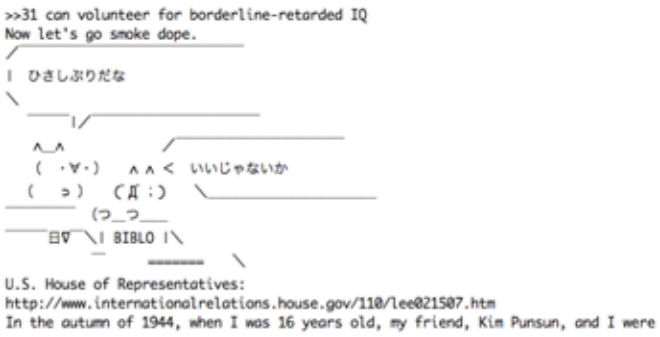

Figure 3. A text art area candidate

Fig.2 shows an example of input text data for our extraction method, where a text art is between English text. Fig.3 shows a text art area candidate obtained at the step 2 where there are redundant lines before and after the text art. Fig.4 shows a text art extracted at the step 3 where the redundant lines has been removed.

4. Text Attributes

In this section, we explain 7 text attributes for a text art recognition machine used in our text art extraction method. We categorize them into two groups: language-independent attributes and language-dependent attributes. We will evaluate combinations of them by a text art recognition test in the next section.

4.1 Language-Independent Attributes

We first explain text attributes which represent characteristics of text art: *H, R* and *G*. According to our observation of text art, same characters often occur successively and same sequences of characters occur two or more times in each text art. The three attributes reflect such characteristics of text art. We then explain text attributes which represent text size: *L* and *S*.

The attribute *H* is an attribute which represents the occurrence number of horizontally successive same two characters per a line. We show an example of the attribute *H*. Fig.5 shows a rectangle consisting of asterisks with 3 lines. In the figure, horizontally successive two characters "**" occurs 4 times. As a result, the attribute *H* of Fig.5 is $\frac{4}{3}$.

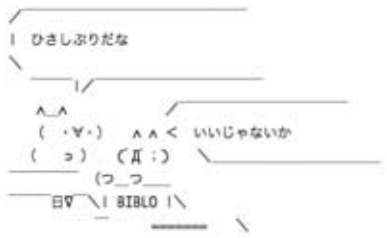

Figure 4. A result of our text art extraction

Figure 5. A rectangle consisting of asterisks

The next two attributes *R* and *G* are based on data compression ratios. To measure compression ratios, we use two data compression methods: Run Length Encoding (RLE) and LZ77. RLE is an encoding method which focuses on how many times same character occurs successively. For example, a string "AAABAAAA" is encoded as "3A1B4A" because there occur 3 of "A", 1 of "B" and 4 of "A" in this order. LZ77 [11] is a dictionary-based compression method. Scanning a given text data, LZ77 records strings in the given text to a dictionary and replaces a string in the given text with a pointer to the same string recorded in the dictionary.

The attribute *R* is an attribute based on data compression ratio by RLE. Given a text *T* consisting of n lines (l_0, l_1 ..., l_{n-1}), the attribute is defined as follow.

$$R \equiv \frac{\sum_{i=0}^{n-1} \left| RLE(l_i) \right|}{\left| T \right|} \tag{1}$$

where $|x|$ denotes the byte size of a string x and $RLE(x)$ denotes a string encoded from the string x by RLE. The attribute *R* is similar to the attribute *H*. *R*, however, can distinguish a string "AAAAAAB" from another string "AAABAAAA" while *H* can't. The attribute *G* is an attribute based on data compression ratio by LZ77. Given a text *T*, the attribute is defined as follow.

$$G \equiv \frac{\left| LZ77(T) \right|}{\left| T \right|} \tag{2}$$

where $|x|$ denotes the byte size of a string x and $LZ77(x)$ denotes a string encoded from the string x by LZ77. The attribute *G* can reflect both successive occurrences of same characters and occurrences of same strings in the entire text while the attributes *H* and *R* can reflect the former only.

The attributes *L* and *S* are the number of lines and the number of bytes of text data respectively.

4.2 Language-Dependent Attributes

The attribute *W* is an attribute which represent the occurrence number of natural language words per a line. Given a text *T* consisting of *n* lines, the attribute is defined as follow.

$$W \equiv \frac{Words(T)}{n} \tag{3}$$

where *Words(T)* denotes the occurrence number of natural language words in the string *T*. We need a dictionary to count words in text data.

The attribute B is a 256 dimension attribute which represent a byte pattern per a line. The byte pattern is a vector whose i-th ($0 \leq i \leq 255$) element is the occurrence number of the byte data i in the byte stream of UTF-8 text data. Given a text T consisting of n lines and its byte pattern (b_0, b_1, ..., b_{255}), the attribute is defined as follow.

$$B \equiv \frac{1}{n}\left(b_0, b_1, ..., b_{255}\right) \tag{4}$$

W and B are language-dependent attributes because W depends on words in the dictionary and B depends on characters used in text. The attributes used in the related work [2, 3, 5] are similar to these attributes as mentioned in section 2.

5. Recognition Test

To compare the performance of text art recognition machines with different combinations of the 7 text attributes in section 4, we had a recognition test as follows.

5.1 Text Data and Text Attributes

We used two sets of text data E and J for machine learning and recognition. The set of text data E consists of English text data with 289 text arts and 290 non-text arts, whose lines range from 1 to 118. The set of text data J consists of Japanese text data with 259 text arts and 299 non-text arts, whose lines range from 1 to 39.

We used 7 attributes of text data in section 4, which are *H, R, G, L, S, W* and *B*. To count the attribute *W*, we used a dictionary with 27,086 Japanese words and a dictionary with 70,221 English words. The former dictionary was generated from the morphological dictionary for Japanese [4]. The latter dictionary was generated from WordNet [7].

5.2 Training and Testing

We used the following 63 combinations of the 7 attributes.

- At least one of *H, R, G, L, S, W* and *B* is used.
- At most one of *H, R* and *G* is used.

For each combination of the 7 attributes, we had the following four cases.

- We used *E* as training data, *E* as test data and the English dictionary.
- We used *E* as training data, *J* as test data and the English dictionary.
- We used *J* as training data, *E* as test data and the Japanese dictionary.
- We used *J* as training data, *J* as test data and the Japanese dictionary.

For each of the four cases, we calculated the precision p, the recall r, the F-measure and the average of the F-measures. The F-measure is as follows.

$$F - measure \equiv \frac{2}{\frac{1}{p} + \frac{1}{r}} \tag{5}$$

We implemented a text attribute extraction program in Perl. To measure the attribute *W*, we constructed word recognition machines from the dictionaries. They are based on the Aho-Corasick algorithm [1]. We used decision trees as text art recognition machines. The decision trees were constructed by the C4.5 machine learning algorithm implemented in the data mining tool Weka [6, 10].

5.3 Results

Table 1, 2, 3 and 4 show the results of the recognition test. Each table shows combinations of attributes and averages of F-measures. The best cases without any of *H, R* and *G* are C4.5-9 and C4.5-10 where the average of the F-measure is 0.857 (Table 1). The best cases with *H* are C4.5-26 and C4.5-28 where the average of the F-measure is 0.901 (Table 2). The best case with *R* is C4.5-44 where the average of the F -measure is 0.967 (Table 3). The best cases with G are C4.5-49 and C4.5-57 where the average of the F-measure is 0.883 (Table 4). As a result, the average of F-measure in each of the best cases with any of *H, R* and *G* is higher than that in the best case without any of them.

Table 5 compares *F*-measures of C4.5-42 (*R, L* and *W*), C4.5-46 (*R, L, S* and *W*) and C4.5-44 (*R, L* and *S*) for each combination of the training data and the test data. C4.5-42 and C4.5-46 are the second best cases in the 63 cases and are the best cases in the cases with at least one language-dependent attributes. The average of the *F*-measure of them is 0.964. C4.5-44 is the best case in the 63 cases and it does not include any language-dependent attribute. Each *F*-measure of C4.5-44 is not less than the corresponding *F*-measures of C4.5-42 and C4.5-46. Especially when the training data is a set of English text data and the test data is a set of Japanese text data, the *F* measure of C4.5-44 is 1% higher relative to the corresponding *F* measures of C4.5-42 and C4.5-46.

Case.	H	R	G	L	S	W	B	Avg. of F
C4.5-1							√	0.834
C4.5-2						√		0.793
C4.5-3						√	√	0.829
C4.5-4					√			0.703
C4.5-5					√		√	0.854
C4.5-6					√	√		0.798
C4.5-7					√	√	√	0.829
C4.5-8				√				0.787
C4.5-9				√			√	0.857
C4.5-10				√		√		0.857
C4.5-11				√		√	√	0.829
C4.5-12				√	√			0.807
C4.5-13				√	√		√	0.854
C4.5-14				√	√	√		0.848
C4.5-15				√	√	√	√	0.829

Table 1. The results of the recognition test without any of the attributes H, R and G

Case.	H	R	G	L	S	W	B	Avg. of F
C4.5-16	√							0.850
C4.5-17	√						√	0.834
C4.5-18	√					√		0.872
C4.5-19	√					√	√	0.829
C4.5-20	√				√			0.847
C4.5-21	√				√		√	0.854
C4.5-22	√				√	√		0.846
C4.5-23	√				√	√	√	0.829
C4.5-24	√			√				0.876
C4.5-25	√			√			√	0.857
C4.5-26	√			√		√		0.901
C4.5-27	√			√		√	√	0.829
C4.5-28	√			√	√			0.901
C4.5-29	√			√	√		√	0.854
C4.5-30	√			√	√	√		0.883
C4.5-31	√			√	√	√	√	0.829

Table 2. The results of the recognition test with the attributes H

5.4 Evaluation

Though the text attributes *H, R* and *G* contribute to text art recognition, *G* does not contribute well as *H* and *R*. The reason would be that there are occurrences of same strings in both text art and natural language sentences. For example, a string like a horizontal line "---" may occur in different lines of a text art, and so-called stop words such as "the", "an", "is" and "are" often occur in English text.

The combination of the attributes *R, L* and *S* is the most language-independent in all the combinations as follows.

The combination of *R, L* and *S* recognizes text art as well as or better than the second best combinations including a language-dependent attribute when the languages of training data and test data are different.

Case.	H	R	G	L	S	W	B	Avg. of F
C4.5-32		√						0.956
C4.5-33		√					√	0.855
C4.5-34		√				√		0.959
C4.5-35		√				√	√	0.861
C4.5-36		√			√			0.965
C4.5-37		√			√		√	0.852
C4.5-38		√			√	√		0.963
C4.5-39		√			√	√	√	0.862
C4.5-40		√		√				0.966
C4.5-41		√		√			√	0.864
C4.5-42		√		√		√		0.964
C4.5-43		√		√		√	√	0.862
C4.5-44		√		√	√			0.967
C4.5-45		√		√	√		√	0.864
C4.5-46		√		√	√	√		0.964
C4.5-47		√		√	√	√	√	0.862

Table 3. The results of the recognition test with the attributes R

Case.	H	R	G	L	S	W	B	Avg. of F
C4.5-48			√					0.653
C4.5-49			√				√	0.883
C4.5-50			√			√		0.809
C4.5-51			√			√	√	0.829
C4.5-52			√		√			0.682
C4.5-53			√		√		√	0.841
C4.5-54			√		√	√		0.811
C4.5-55			√		√	√	√	0.829
C4.5-56			√	√				0.787
C4.5-57			√	√			√	0.883
C4.5-58			√	√		√		0.865
C4.5-59			√	√		√	√	0.829
C4.5-60			√	√	√			0.811
C4.5-61			√	√	√		√	0.841
C4.5-62			√	√	√	√		0.866
C4.5-63			√	√	√	√	√	0.829

Table 4. The results of the recognition test with the attributes G

Training data	Test data	C4.5-42	C4.5-46	C4.5-44
E E J J	E J E J	1.000 0.899 1.000 0.956	1.000 0.899 1.000 0.958	1.000 0.908 1.000 0.958

Table 5. F -measures of C4.5-42, C4.5-46 and C4.5-44

The combination of *R, L* and *S* does not require any dictionary while one of the second best combination includes the language-dependent attribute *W* needs a dictionary to count words in text data.

6. Extraction Test

By extraction test, we measured the performance of our text art extraction methods with the combination of three attributes *R, L* and *S*. The combination is the best case in the recognition test and does not include any language-dependent attribute. We also measured the effect of text area reduction in our extraction method.

6.1 Text Data and Text Attributes

We constructed training data and test data as follows. We divided the set of text data *E* and *J* used in the recognition test into two groups *A* and *B*. Each of *A* and *B* consists of English text and Japanese text. We then made 800 text data from *A* as test data. Each of the 800 text data consists of three parts *X*, *Y* and *Z* where *X* and *Z* are randomly selected non-text art data from *B* and *Y* is randomly selected text art data from *B*. Each of *X, Y* and *Z* is English or Japanese text data. Fig.6 shows an example of test data where *X* and *Z* are English text and *Y* is a Japanese text art. We also made 800 text data from *B* similarl.

Figure 6. An example of test data for the extraction test

Window width	Avg. of Precision	Avg. of Recall	Avg. of F -measure
1	0.939	0.879	0.908
2	0.925	0.917	0.921
3	0.918	0.926	0.922
4	0.916	0.932	0.924
5	0.906	0.933	0.919

Table 6. The results of the extraction test with the text area reduction procedure

Window width	Avg. of Precision	Avg. of Recall	Avg. of F -measure
1	0.939	0.879	0.908
2	0.774	0.980	0.865
3	0.776	0.954	0.856
4	0.732	0.960	0.831
5	0.690	0.961	0.803

Table 7. The results of the extraction test without the text area reduction procedure

6.2 Training and Testing

1. We constructed a recognition machine by the C4.5 machine learning algorithm with the attributes *R, L* and *S* extracted from *A*. We also constructed a recognition machine by C4.5 with the same attributes extracted from *B*.
2. For each of the cases with and without the text area reduction procedure, we had the following.
   (a) Changing window width from 1 to 5, we extracted text arts. We used the 800 text data constructed from *B* as test data and the recognition machine constructed from *A*. We calculated the average of the precision, that of the recall and that of the *F* -measure by each window width.

(b) We swapped A and B, and extracted text arts similarly.

(c) For each window width, we calculated the average of the precision, that of the recalls and that of the F -measure.

We used Perl and Weka in this experiment as we used them in the recognition test.

6.3 Results

The table 6 shows the results of the extraction test with the text area reduction procedure. While window width changes from 1 to 5, the average of the precision decreases from 0.939 to 0.906 and the average of the recall increases from 0.879 to 0.933. The average of the F -measure takes the highest value 0.924 at window width 4, and takes the lowest value 0.908 at window width 1.

The table 7 shows the results of the extraction test without the text area reduction procedure. While window width changes from 1 to 5, the average of the precision decreases from 0.939 to 0.690 and the average of the recall increases from 0.879 to 0.961. The average of the F -measure takes the highest value 0.908 at window width 1, and takes the lowest value 0.803 at window width 5.

6.4 Evaluation

The text area reduction procedure contributes to keeping the F -measure high. Why the procedure increased the maximum of the average of the precision is that the procedure removed non-text art lines around text art in window. Why the procedure decreased the maximum of the average of the recall is that the procedure also removed the edges of text art sometimes. As a result, the average of the F -measure is at least 0.908 in the case with the procedure while it is at most 0.908 in the case without the procedure.

7. Discussion

The combination of text attributes *R, L* and *S*, which took the highest *F*-measure in our text art recognition test, does not need any dictionary. It is important not only for language independency but also for consumed memory size. One application area of our text art extraction method would be handheld computers with small displays such as cellular phones. They have the small amount of memory relative to desktop PCs and server computers.

If it is enough to use a dictionary consisting of stop words only though the two dictionaries used in the text art recognition test contain 27,086 Japanese words and 70,221 English words respectively, we can reduce the memory sizes consumed by text art recognition machines with the attribute W .

A future work is to improve the performance of our text art extraction method with new text attributes. The attribute *R* reflects horizontally successive occurrences of same characters. If we introduce an attribute which reflects vertically successive occurrences of same characters, the performances of text art extraction would be improved. we, however, have to take into account font widths of each characters in text data to measure such attribute correctly. It is also a future work that we use other machine learning methods such as SVM and ensemble learning.

8. Conclusion

We proposed a text art extraction method based on a text art recognition machine constructed by a machine learning algorithm. We tested the recognition machines with 63 combinations of 7 text attributes including both language-independent attributes and language-dependent attributes. According to the results of the recognition test, the best combination consists of language-independent attributes, which are an attribute based on data compression ratio by Run Length Encoding (RLE) and two attributes based on text size. The attribute based on RLE captures the characteristics of text art such that same characters occur successively. We also tested our text art extraction method with the recognition machine. The highest average of *F*-measure of the precision and the recall is 0.924 in the extraction test.

References

[1] Aho, A. V., Corasick, M. J. (1975). Efficient string matching: an aid to bibliographic search. *Communications of the ACM*, 18 (6) 333–340.

[2] EGG. AAscan(in japanese). http://www11.plala.or.jp/egoo/download/download index.html (Retrieved on Dec. 14, 2008).

[3] Hiroki, T., Minoru, M (2005). Ascii Art Pattern Recognition using SVM based on Morphological Analysis. *Technical report of IEICE. PRMU*, 104 (670) 25–30. 0218.
[4] ICOT. Morphological Dictionary for Japanese. http://www.icot.or.jp/ARCHIVE/Museum/IFS/abst/033-J.html , Accessed on Dec. 14, 2008.
[5] Nakazawa, M., Matsumoto, K., Yanagihara, T., Ikeda, K., Takishima, Y., Hoashi, K. (2010). Proposal and its Evaluation of ASCII-Art Extraction, *In*: *Proceedings of the 2nd Forum on Data Engineering and Information Management* (DEIM2010), p. C9–4.
[6] T. U. of Waikato. Weka 3 -Data Mining with Open Source Machine Learning Software in Java. http://www.cs.waikato.ac.nz/ml/weka/, Accessed on Dec. 14, 2008.
[7] Princeton University Cognitive Science Laboratory. WordNet. http://wordnet.princeton.edu/ Accessed on Dec. 14, 2008.
[8] Suzuki, T., Hayashi, K. (2009). A Language-Independent Text Art Extraction Method. *In*: Proceedings of the 2nd International Conference on the Applications of Digital Information and Web Technologies, p. 462 – 467. IEEE Computer Society.
[9] Suzuki, T., Hayashi, K. (2010). Text Data Compression Ratio as a Text Attribute for a Language-Independent Text Art Extraction Method. *In:* Proceedings of the 3rd International Conference on the Applications of Digital Information and Web Technologies.
[10] Witten, I. H., Frank, E (2005). Data Mining: Practical Machine Learning Tools and Techniques (Second Edition). Morgan Kaufmann.
[11] Ziv, J., Lempel, A (1977). A Universal Algorithm for Sequential Data Compression. *IEEE Transactions of Information Theory*, 23 (3) 337– 343.