RESEARCH REPORT ISIS-RR-94-3E

Text Classification and Automatic Extraction of Keywords by Learning Decision Trees *

Yasubumi Sakakibara Kazuo Misue
Takeshi Koshiba

March, 1994

Institute for Social Information Science (*ISIS*),
FUJITSU LABORATORIES LTD.

Numazu office
140 Miyamoto, Numazu-shi, Shizuoka 410-03, Japan
Telephone: (Numazu) 0559-23-2222 Telex: 3922508J Fax: 0559-24-6180

Tokyo office
1-9-3, Nakase, Mihama-ku, Chiba-shi, Chiba 261, Japan
Telephone: (Chiba) 043-299-3211 Fax: 043-299-3075

Text Classification and Automatic Extraction of Keywords by Learning Decision Trees *

Yasubumi Sakakibara Kazuo Misue Takeshi Koshiba

Institute for Social Information Science (*ISIS*),

FUJITSU LABORATORIES LTD.

140 Miyamoto, Numazu-shi, Shizuoka 410-03, Japan

Email: {yasu,misue,koshiba}@iias.flab.fujitsu.co.jp

Abstract

In this paper, we show an interesting and useful application of machine learning to information retrieval. We propose a completely new approach to the problem of text classification and extracting keywords by using machine learning techniques. We introduce a class of representations for classifying text data based on decision trees, that is, decision trees over attributes on strings, and present an algorithm for learning it inductively. Our algorithm has the following features: it does not need any natural language processing technique, and it is robust for noisy data. We show that our learning algorithm can be used for automatic extraction of keywords for text retrieval and automatic text categorization. We also demonstrate some experimental results using our algorithm on classifying text data and extracting keywords in order to show that our approach is a most successful and promising way to apply machine learning techniques.

*An extended abstract of the paper was presented at 9th Conference on Artificial Intelligence for Applications (CAIA'93), Florida, USA.

1 Introduction

Recently text categorizations and keyword extractions for those text categorization tasks have been central to the domain of information retrieval. Especially techniques for processing those tasks by computer are expected to be developed, because several electric storage systems like CD-ROM are now available to store a large amount of texts (e.g., full text database) easily and there is a great demand for techniques to retrieve necessary information efficiently from those large storages. Several works on this problem have been studied by using natural language processing techniques (e.g., [RL92]), data compression techniques, and so on. However those proposed methods are too heavy and complicated for the ordinal users to use easily.

In this paper, we propose a completely new approach to this problem by using machine learning techniques. One important and hard problem on information retrieval is how to extract adequate keywords for retrieving necessary information. This is a professional work and very hard to be solved by computer automatically. We apply machine learning techniques to solving this problem. We introduce a new class of representations to represent rules for classifying texts, and propose an approach of producing such a rule and simultaneously extracting keywords by learning inductively from already classified texts. This approach can be viewed as an inductive learning from the training sample. Hence we can employ many machine learning techniques for it.

In this paper, we first introduce a new class of representations for classifying text data based on decision trees, called *text-classification trees*. Decision trees are used for classification tasks when concepts are defined in terms of a set of attribute-value pairs. A classification task is to assign an element of the domain to one of a specified number of disjoint classes. A text-classification tree classifies an input text (sentence, string) into one category according to several tests (attributes) whether the input text has some specific features, e.g., the text contains some important keywords. Next we present a learning algorithm to construct such a text-classification tree from already classified texts. Our algorithm has the following features: it does not need any natural language processing technique, and it is robust for noisy data. Then adequate and relevant keywords (features) are now extractable from the text-classification tree output by the learning algorithm. We also demonstrate some experimental results using our algorithm on the problem of classifying books according to their titles from a given training sample, and show that our approach is a most successful and promising way to apply machine learning techniques.

2 Decision trees over attributes on strings for classifying text data

We introduce a class of representations for classifying text data, that is, *decision trees over attributes on strings*.

Decision trees are used for classification tasks when concepts are defined in terms of a set of attribute-value pairs. A classification task is to assign an element of the domain to one of a specified number of disjoint classes. For example, the diagnosis of a medical condition from symptoms is a classification task, in which the classes could be either the various disease states or the possible therapies.

Here we define such decision trees over attributes on strings. Attributes on strings can be defined as functions from Σ^* to $\{0,1\}$, where Σ is a finite alphabet and Σ^* denote the set of all finite strings over Σ. In this paper, we use simple attributes on strings, called *keyword attributes* and denoted $key(w)$, defined as follows: For a string w in Σ^*,

$$key(w)(x) = \begin{cases} 1 & \text{if a string } x \text{ contains } w \text{ as a substring,} \\ 0 & \text{otherwise.} \end{cases}$$

Let CT be the set of all keyword attributes on Σ^*, that is, the set $\{key(w) \mid w \in \Sigma^*\}$. We call decision trees defined over CT on the domain Σ^* *text-classification trees*. Thus a text-classification tree is a binary tree where each internal node is labeled with a string and each leaf is labeled with a class name. Figure 1 shows an example of a text-classification tree that classifies books according to their titles.

Each text-classification tree classifies an input string as follows. An input string determines a unique path from the root to a leaf: at each internal node the right (respectively left) edge to a child is taken if the input string contains the string labeled at that internal node as a substring (respectively does not contain the labeled string). The class that the input string is classified into is the class at the leaf reached. For example, the book entitled "Proceedings of IEEE symposium on circuits and systems" is classified into the class "3" by the tree shown in Figure 1.

3 Noise-tolerant algorithm for learning text-classification trees

In this section, we present an algorithm for learning text-classification trees, and in the next section, we show some experimental results of our learning algorithm.

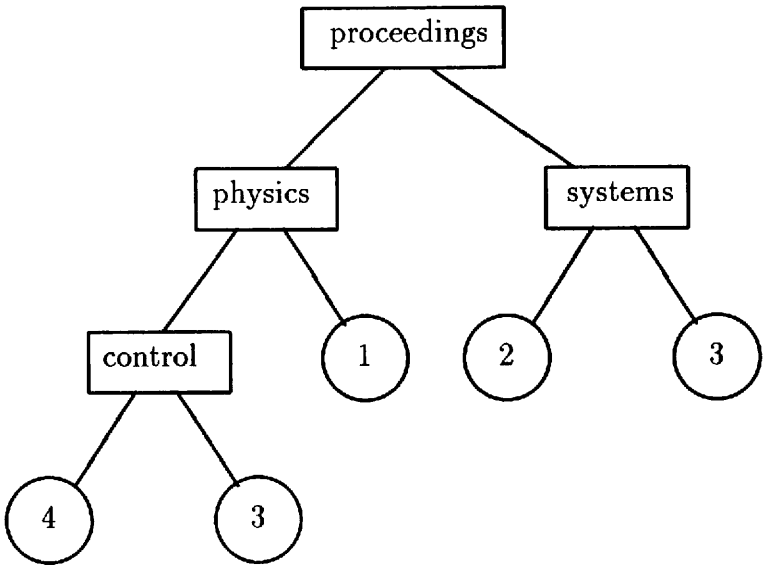

Figure 1: A text-classification tree that classifies books according to their titles.

The learning algorithm constructs a text-classification tree inductively from some already classified texts. A *classified text* means a pair of a text and a class which the text has been classified into. For example, in the case of bibliographic data, a classified text is a pair of the title of a book and the category of the book, e.g., "[Aspects of Artificial Intelligence, Information Science]". In the machine learning literature, such a classified text which is input to the learning algorithm is called an *example* and a set of examples is called a *sample*.

The specific features of our learning algorithm are as follows:

1. The algorithm constructs a text-classification tree in a top-down manner started from the root node inductively from the given sample (Top-Down Induction of Decision Tree).

2. The algorithm uses the Quinlan's (ID3) [Qui86] entropy function as the attribute selection function.

3. The algorithm is robust for classification noise contained in the sample.

4. The algorithm does not need any natural language processing technique.

The fourth feature is realized by using a simple technique of taking every substrings

of the given sample texts as candidates for keyword attributes. This feature contributes to make the system very simple and light.

The third feature is based on our theoretical work for learning decision trees in the presence of noise [Sak93]. The classification noise model proposed in [AL88] deals with a noise where errors will occur on class labels in the examples. We use the fundamental mechanism to deal with such a noise shown in [Sak93].

Now we present our algorithm for learning text-classification trees from the given sample of classified texts. An *example* is a pair (w, l), where w is a string in Σ^* and l is a class label. A *sample* is a finite set of examples. Let S be a sample, $v \in \Sigma^*$, and l be a label. We define S_1^v, S_0^v, and *Occur* as follows:

$$\begin{aligned} S_1^v &= \{(w,l) \in S \mid w \text{ contains } v \text{ as a substring}\} \\ S_0^v &= \{(w,l) \in S \mid w \text{ does not contain } v\} \\ Occur(S,c) &= |\{(w,l) \mid l = c\}| \end{aligned}$$

We say a string v is *informative* (on S) if both S_0^v and S_1^v are nonempty.

Let there be m disjoint classes and $l_1, l_2, \ldots, l_m$ be their labels. Let X be a finite set of examples. We use the following attribute selection function *Loss* in the learning algorithm:

$$I(X) = -\sum_{j=1}^{m} \frac{Occur(X,l_j)}{|X|} \log_2 \frac{Occur(X,l_j)}{|X|}$$

$$Loss(v,X) = \frac{|X_0^v|}{|X|} I(X_0^v) + \frac{|X_1^v|}{|X|} I(X_1^v)$$

The algorithm *LEARN* is shown in Figure 2.

In order to deal with classification noise, the algorithm *LEARN* takes two parameters, the parameter *nsrt* for noise rate and the parameter *prnrt* for pruning rate, as input. The values for these two parameters are used for the stopping conditions in the subprocedure *FINDS*. This mechanism has been developed for dealing with classification noise in [Sak93]. We can formally calculate the adequate values for two parameters *nsrt* and *prnrt* in the framework of Valiant's PAC learning model [Val84]. However, we will empirically estimate the values for these parameters at our experiences in the next section.

ALGORITHM *LEARN*
Input:

- A sample S,
- Parameters $keyl1$, $keyl2$, $prnrt$, and $nsrt$.

Output:

A decision tree T.

Procedure:

1. Calculate the following:

$$Keywords = \{v \mid keyl1 \leq |v| \leq keyl2, \; v \text{ is a substring of } w \text{ for some example } (w, l)\};$$

2. Let $T = FINDS(S, Keywords, prnrt, nsrt)$;
3. Output T and halt.

Subprocedure $FINDS(S, Keywords, prnrt, nsrt)$:

1. If $(|S| - Occur(S, l_i))/|S| \leq nsrt$ for some l_i,
   stop and return the decision tree $T = l_i$;
2. If $|S| \leq prnrt$,
   stop and return the decision tree $T = l_i$ for a largest $Occur(S, l_i)$;
3. Else
   - 3.1. Calculate $Loss(v, S)$ for all $v \in Keywords$ that is informative for S;
   - 3.2. If there is no informative keyword in $Keywords$,
     then stop and return $T =$"bad";
   - 3.3. Choose a longest keyword v_g that minimizes $Loss(v_g, S)$;
   - 3.4. Let $T_0 = FINDS(S_0^{v_g}, Keywords - \{v_g\}, prnrt, nsrt)$
     and $T_1 = FINDS(S_1^{v_g}, Keywords - \{v_g\}, prnrt, nsrt)$;
   - 3.5. Stop and return the decision tree with root labeled v_g, left subtree T_0 and right subtree T_1;

Figure 2: Algorithm for learning text-classification trees.

01!ALGEBRA VOLUME I
01!ALGEBRAIC GEOMETRY
01!ANALYSIS I
01!COMPUTER ALGEBRA
01!CONVERGENCE OF STOCHASTIC PROCESSES
02!ADVANCES IN COMPUTING AND INFORMATION
02!ADVANCES IN CRYPTOLOGY CRYPTO 89
02!ALGORITHMS AND DATA STRUCTURES
02!ANALOGICAL AND INDUCTIVE INFERENCE
02!ARTIFICIAL INTELLIGENCE
02!COMPUTATIONAL LOGIC
03!COMPUTATIONAL LINGUISTICS
04!FOUNDATIONS OF ROBOTICS
04!ROBOTICS SCIENCE
04!SYSTEMS AND CONTROL
05!MOLECULAR BIOLOGY OF THE CELL
06!MENTAL PROCESSES
07!THE PROFESSIONAL DECISION THINKER

Figure 3: A part of the data in a library.

Category	Contents
00. Reference	Handbook, Dictionary, etc.
01. Mathematics	Algebra, Probability, Numerical analysis, etc.
02. Information Science	Information science, Computer science, etc.
03. Language Science	Linguistics, Syntax, etc.
04. System Science	System theory, Control theory, Robotics, etc.
05. Biological Science	Bio-technology, DNA theory, Neural science, etc.
06. Culture Science	Philosophy, Psychology, etc.
07. Social Science	Social science, Politics, Management, etc.
08. Engineering	Electrical engineering, Mechanical engineering, etc.
09. Physics	Physical science, etc.
10. Environmental Science	Environmental science, etc.
11. Education	Pedagogy, Library science, etc.

Figure 4: Categories for books.

4 Experimental results

This section describes our experiment on the problem of learning text-classification trees for classifying books according to their titles from a given sample. Our data consist of a set of pairs of the title of a book and its category in some library. Figure 3 shows a part of the data, where the format is "category no. ! the title of a book". Figure 4 shows the list of categories' numbers, names, and their explanations.

We divide this data into two parts: a training sample and a test set. By feeding the training sample to our learning algorithm *LEARN*, we get a text-classification tree as the output of the learning algorithm. Then we check the following two items:

- What kind of keywords can be appeared on the internal nodes of the tree,

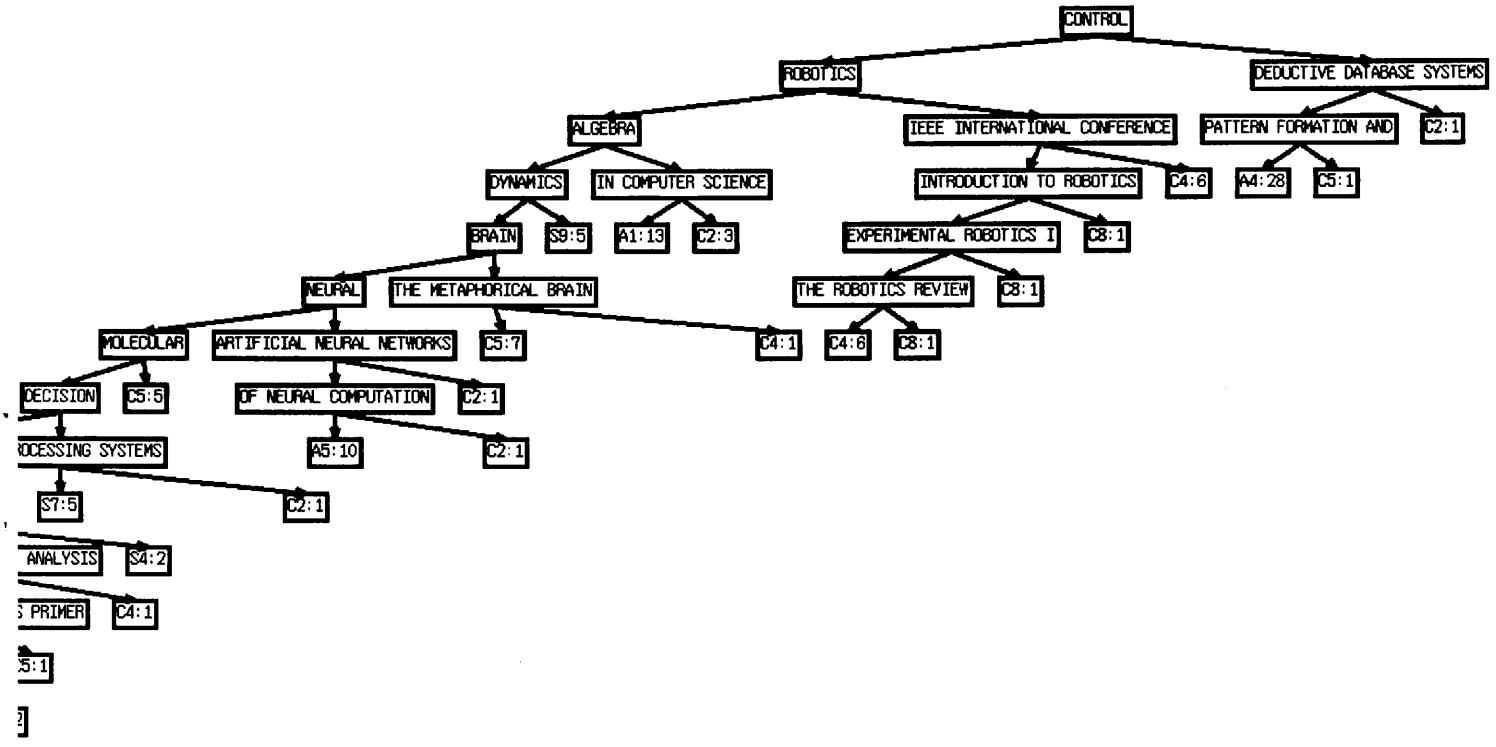

Figure 5: A part of the text-classification tree constructed by *LEARN*.

- How precisely the learned text-classification tree can classify the test set.

We use 80% of the whole data for the training sample and the remains for the test set.

Figure 5 shows a part of the learned text-classification tree output by our learning algorithm *LEARN*. The format of a label attached to a leaf node is "⟨stopping status⟩ ⟨category no.⟩ : ⟨the number of examples reached to this node⟩", where at the stopping status, "A" means almost examples (more than the ratio *nsrt*) reached to this node have the same label (category no.), "C" means all examples have the same label, and "S" means the number of examples reached to this node is very small (less than the threshold *prnrt*). We can see that several interesting and useful keywords have been extracted by the learning. These extracted keywords reflect the features of the library.

We have run six experiments with three different pairs of training sample and test set and two different values for the parameters *nsrt* and *prnrt*. Figure 6 shows the ratios on test sets correctly classified by text-classification trees learned from those training

	$nsrt = 0.20$, $prnrt = 5$	$nsrt = 0.05$, $prnrt = 3$
Data 1	72%	74%
Data 2	67%	67%
Data 3	69%	71%

Figure 6: Correctly classified ratios by the learned trees.

samples.

These results are not so high and good performances in machine learning literature. We consider this result of our learning is because of the following reasons:

- The number of the data is too small. The data more than thousand will be needed to produce significant statistics.
- A book often has its own keywords in the title.
- It is hard to classify books according to only their titles. Availabilities of abstracts or contents of books will improve the accuracy of the classifications.

5 Application to automatic extraction of keywords

As we have seen in the above sections, our learning algorithm can be used to extract keywords from already classified texts automatically. We employ our learning algorithm to produce a text-classification tree from already classified texts, and extract important keywords which appear on internal nodes of the learned tree. These extracted keywords are important and useful in the following sense: These keywords are extracted to distinguish texts (i.e., classifying into different classes is same as distinguishing) and information retrieval is fundamentally a task to distinguish necessary texts (information) from unnecessary ones. This provides a new view of information retrieval.

Thus our method of using a learning algorithm for text-classification trees has such a distinguished feature from usual methods for keyword extractions based on techniques of natural language processing or using frequency in appearance of a word.

6 Concluding remarks

We are now proceeding to work on the following problems:

1. Since our learning algorithm takes every substrings of the given sample as candidates for keyword attributes, it takes a large amount of time to process them. These attributes of substrings contains many unnecessary attributes, that is, irrelevant features, and it will be more effective if we eliminate such irrelevant features before learning. We will be able to use several techniques presented in [AD91] to do so.

2. We can consider more expressive and powerful attributes for text-classification trees than just strings. We are now having experiments where the attributes are proposed to be strings which include a special symbol "$*$" like the wild-card that can match any string. For example, the string "$aa*bb$" can match any string which starts with "aa" and ends with "bb".

   We have already some experimental results on this work. Figure 7 shows a part of a text-classification tree over such attributes using the symbol "$*$" which have been learned from the same training sample as Figure 5.

3. We will consider logical "and" and logical "or" of keyword attributes.

4. We will consider some noise on attributes, that is, errors on strings. We consider three types of errors on strings, called *EDIT operation errors*. EDIT operations consist of "insertion", "deletion", and "change" of a symbol in a string. We call such a noise where the examples are corrupted by random errors of EDIT operations on strings the *EDIT noise*. Several theoretical results on this problem have already been obtained in [SS92].

References

[AD91] Hussein S. Almuallim and Thomas G. Dietterich. Learning with many irrelevant features. In *Proceedings of 9th National Conference on Artificial Intelligence (AAAI'91)*, pages 547–552. AAAI Press/MIT Press, 1991.

[AL88] Dana Angluin and Philip Laird. Learning from noisy examples. *Machine Learning*, 2:343–370, 1988.

[Qui86] J. Ross Quinlan. Induction of decision trees. *Machine Learning*, 1:81–106, 1986.

[RL92] Ellen Riloff and Wendy Lehnert. Classifying texts using relevancy signatures. In *Proceedings of 10th National Conference on Artificial Intelligence (AAAI'92)*, pages 329–334. AAAI Press/MIT Press, 1992.

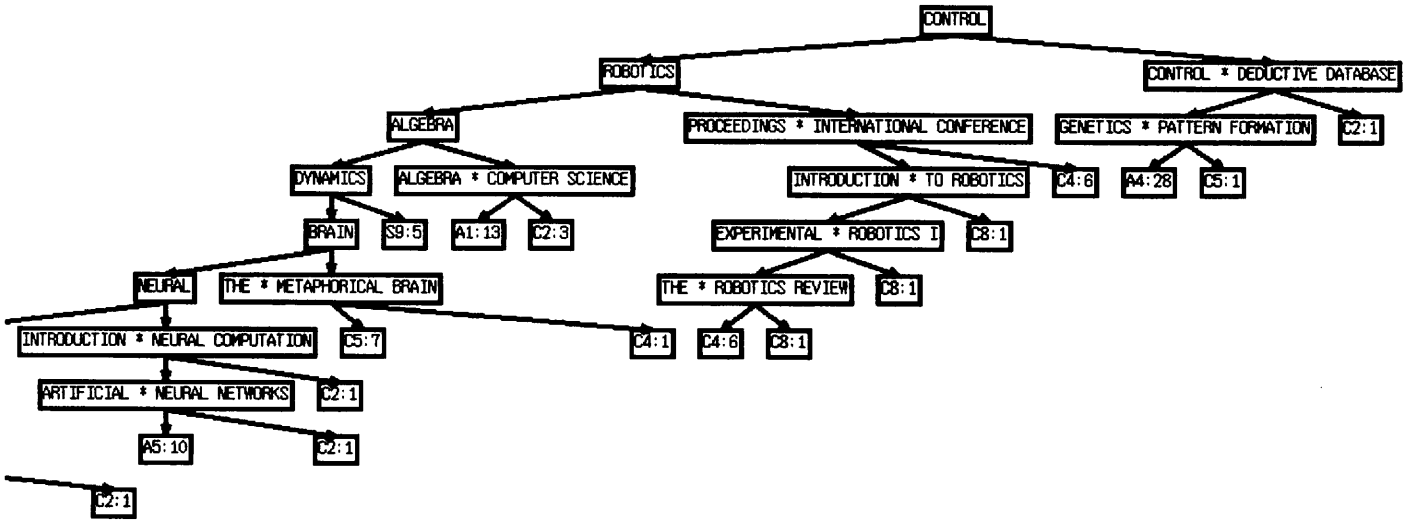

Figure 7: A part of the learned text-classification tree over attributes using "$*$".

[Sak93] Yasubumi Sakakibara. Noise-tolerant Occam algorithms and their applications to learning decision trees. *Machine Learning*, 11:37–62, 1993.

[SS92] Yasubumi Sakakibara and Rani Siromoney. A noise model on learning sets of strings. In *Proceedings of 5th Workshop on Computational Learning Theory (COLT'92)*, pages 295–302. ACM Press, 1992.

[Val84] Leslie G. Valiant. A theory of the learnable. *Communications of the ACM*, 27:1134–1142, 1984.