Soft Comput (2006)
DOI 10.1007/s00500-006-0091-5

FOCUS

Jun Xu · Yalou Huang

Using SVM to extract acronyms from text

Abstract The paper addresses the problem of extracting acronyms and their expansions from text. We propose a support vector machines (SVM) based approach to deal with the problem. First, all likely acronyms are identified using heuristic rules. Second, expansion candidates are generated from surrounding text of acronyms. Last, SVM model is employed to select the genuine expansions. Analysis shows that the proposed approach has the advantages of saving over the conventional rule based approaches. Experimental results show that our approach outperforms the baseline method of using rules. We also show that the trained SVM model is generic and can adapt to other domains easily.

Keywords Acronym · Expansion · Classification · Support vector machines

1 Introduction

People will find it helpful if we develop a system that can automatically recognize acronyms (e.g., ROM) and their expansions (e.g., *Read Only Memory*) from Text. This is because many organizations have a large number of online documents which contain many acronyms. In many cases, however, acronyms occur frequently enough to make it difficult for outsiders to comprehend text. We conclude that the correct mapping of acronyms to their expansions is very important for understanding the documents.

Manually collected acronym lists have been complied and many are available in Internet.[1] However, most of them are confined in specific domains. Moreover, with the rapid growth of acronyms, maintaining the lists is very hard.

Finding all the acronyms, along with their expansions automatically in documents is a problem of text mining that has often been tackled using ad hoc heuristics. Previous efforts fall into three categories: natural language-based approaches, rule-based approaches, and multiple alignment-based approaches. These approaches rely heavily on manually written rules and patterns.

In the paper, we propose a novel machine learning approach to extracting acronyms from text. First, all 'likely acronyms' are identified by heuristic rules. Second, expansion candidates against each likely acronym are generated from the surrounding text. Last, we employ support vector machine (SVM) model [11] to select the genuine expansions for acronyms.

Compared with conventional pattern-based approach, our machine learning approach has several advantages. First, machine learning frees human labors from the boring process of writing and tuning patterns/rules. Second, machine learning can utilize more evidences than patterns. Only strong evidences can be included into patterns for the difficulties of creating patterns. Machine learning model, however, can utilize both strong and weak evidences collectively. Finally, it is easier to conduct domain adaptation. With appropriate features, model trained in one domain can perform well in others.

Experimental results indicate that our approach performs well on real world. We show that, the built SVM model can always choose the correct expansions for acronyms from a set of candidates. We also find that the SVM model trained on one domain performs well on another. This indicates that our trained model is generic and domain adaptation can be conducted easily.

The rest of the paper is organized as follows. Section 2 introduces related work. Section 3 gives a description of acronym recognition and Sect. 4 discusses our approach in detail. Section 5 reports experimental results and Sect. 6 summarizes our work in the paper.

2 Related Work

In this section, we introduce previous work on automatic acronym extraction. Research on the topic of acronyms

J. Xu (✉) · Y. Huang
College of Software, Nankai University,
No. 94 Weijin Road, Tianjin 300071, China
E-mail: nkxj@yahoo.com.cn
E-mail: yellow@nankai.edu.cn

[1] http://www.acronymfinder.com/ and
http://www.astro.umd.edu/∼marshall/abbrev.html

extraction roughly falls into three categories: natural language-based, rule-based, and multiple alignment-based approaches.

Natural language-based approaches attempt to utilize results of natural language such as part-of-speech tagging to find expansions around acronyms. AcroMed [8] is such a system finding acronym–expansion pairs from medical documents. However, this approach is difficult in practice for the complexity of the pattern matching algorithm and the performance of part-of-speech tagging.

Acrophile [6] is an example of rule-based system. Authors of Acrophile study web documents carefully and create a large number of patterns and rules for identifying acronyms and their expansions from text. Unfortunately, due to the complexity of relationship between acronyms and expansions in web documents, precision of Acrophile is low. Other work please refer to [7,15].

As an example of multiple alignment based approaches, AFP (Acronym Finding Program) conducts extraction in an OCR environment [10]. AFP is based on an inexact pattern matching algorithm applied to text surrounding the possible acronym. Other work falls into this category can be found at [2–4]. Other work on acronym extraction please refer to [1,9,12–14].

3 Properties of acronyms and expansions

Acronyms are relatively new linguistic feature in English language, first appearing in the 1940s [12]. They are often defined by preceding (or following) their first use with a textual explanation. Acronyms are used in place of their expansions, where either reader is familiar with the concepts and entities they stand for or their meaning is clear from the context. Acronyms have following special properties:

- Generally, acronyms do not appear in standard dictionaries. To make their meaning clear, authors may give their expansions at their first use, usually in form of " *ROM* (*Read Only Memory*)" or "*Read Only memory* (*ROM*)".
- Acronyms may be nested. For example, " *ROM*" is part of " *PROM*".
- Acronyms are not necessary unique. For example, "*MIT*" may stand for both "*Massachusetts Institute of Technology*" and "*Management Information Tree*". The later expansion comes from the CISCO glossary list.
- Acronyms are generally three to ten characters in length, although shorter or longer acronyms do exist. Acronyms' characters are alphabetic, numeric, and special characters such as '-', '/', '.' or '&' etc. White spaces rarely appear.

4 Using SVM to extract acronyms

Our approach consists of three steps: identifying likely acronyms, generating expansion candidates, and selecting genuine expansions. Figure 1 illustrates the outputs of each step based on a simple example text "*ROM stands for Read Only Memory*".

4.1 Identifying likely acronyms

The aim of this step is to identify all possible acronyms from original text. Just as the first step of example illustrated in Fig. 1, we are to identify acronym "*ROM*" from text "*ROM stands for Read Only Memory*".

Based on properties of acronyms listed in Sect. 3, in this paper, we consider a string of alphabetic, numeric, and special characters as likely acronym if it satisfies following conditions:

1. Its length is between two and ten characters and at most contains one white space.
2. Its first character is alphabetic or numeric. At least one letter is capital.
3. It is not a known word in dictionary. It is not person name, location name or word in a predefined list of stop words.

The first condition restricts the length of acronyms. The second and third keeps many person names, location names, and common words from being treated as likely acronyms. Based on these restrictions, we can get a list of likely acronyms from text. Obviously, these three restrictions are very loose. Many likely acronyms have no expansions at all. These false identified acronyms will not impact final result because they will be removed automatically in following steps.

4.2 Generating expansion candidates

In this step, we are to generating all expansion candidates for acronyms identified. As the example shown in Fig. 1, we generate a set of expansion candidates.

We notice that expansions always occur in surrounding text where acronyms appear in. Based on this, we generate expansion candidates for an acronym from its surrounding text in the same sentence. One of these candidates may be the genuine expansion for the acronym.

Before generating expansion candidates, sentence broken and tokenization should be conducted on text. Sentences are split and tokens in sentences are segmented by white spaces. Please note that punctuations such as ',', '.', ')', '(' and '!' etc. are considered as single tokens.

A given acronym splits the sentence into two parts: the substring that precedes the acronym (left context) and the substring that follows the acronym (right context). All of the substrings of left and right context are considered as candidates. Two parameters are used for identify an expansion candidate: *length* and *offset*, where *length* is the tokens of the candidate and *offset* is tokens between acronym and expansion. As shown by (1) and (2) in Fig. 2, in both cases, we get the genuine expansion " *Read Only Memory*" for acronym " *ROM*" whe *n offset*=2 and *length*=3. Note ',' is considered as a token.

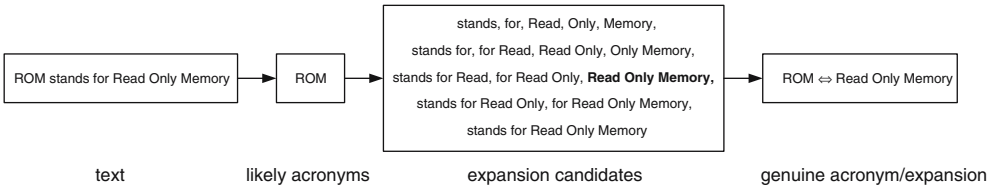

Fig. 1 An example flowchart of acronym extraction

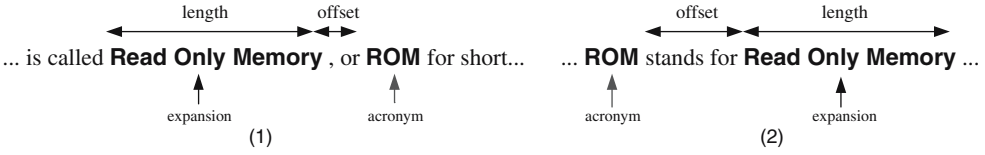

Fig. 2 Generating expansion candidates

Theoretically, all substrings within the sentence can be enumerated as expansion candidates. However, this is not necessary. We notice that the genuine expansions always appear near to acronyms. Lengths of genuine expansions (by tokens) always approximately equal to the lengths of their acronyms (by characters).

Thus we define two parameters for dramatically reducing the number of candidates generated. The first parameter, *maxoffset*, controls the maximum distance between acronyms and their expansions. In our experiments, we set *maxoffset* to ten tokens. It is large enough for generating all genuine expansions. The second parameter, *maxlength*, controls the maximum length of expansions. We assume that, for long acronym, the length of expansion (by tokens) is shorter than the length of acronym (by characters) plus 5; for short acronym, it is shorter than twice of the acronym length. Thus we get Eq. (1) for calculating *maxlength*:

$$\begin{aligned}\text{maxlength} = \min\,&(\text{length(acronym)} \\ &+5, \text{length(acronym)} \times 2),\end{aligned} \tag{1}$$

4.3 Selecting genuine expansions

In the step, a SVM model is used to select the genuine expansions for acronyms from candidate set. As the last step of the example shown in Fig. 1, we are to select candidate "*Read Only Memory*" as the genuine expansion for "*ROM*".

Acronym, expansion candidate, and the context are represented by a set of features. We employ a SVM model to determine whether the candidate is the genuine expansion for the acronym, based on the features (c.f. Fig. 3).

4.3.1 SVM Model

We take a statistical machine learning classifier to address the problem. A set of acronym/expansion candidates is labeled in advance. They are used for training SVM model and testing performance. Let us describe the problem more formally. Given a training dataset $D = \{x_i, y_i\}_{i=1}^{n}$, we construct a model that can minimize error in prediction of y given x (generalization error). Here $x_i \in X$ represents an acronym/expansion candidate pair and $y_i \in \{+1, -1\}$ presents a label indicating whether the candidate is a genuine expansion for the acronym. When applied to a new instance x, the model predicts the corresponding y and outputs the score of prediction. We employ SVM as the classifier model to assign prediction scores. Given an instance x, SVM assign as score to it based on

$$f(x) = w^{\mathrm{T}}x + b \tag{2}$$

where w denotes a vector of weights and b denotes an intercept. The sign of $f(x)$ is used. If it is positive, x is classified into the positive category, otherwise into the negative category. The construction of SVM needs labeled training data. Details of the learning algorithm can be found in [11]. In a few words, the learning algorithm creates the hyperplane in Eq. (2), such that the hyper plane separates the positive and negative instances in the training data with the largest 'margin'.

In some cases, an acronym may be mapped to more than one candidate within one sentence. (SVM model assigns more than one positive score to different candidates within a sentence for one acronym.) We choose the one has highest prediction score as the final genuine expansion. In some other cases, no candidate is identified as expansion for an acronym. (SVM model assigned all generated candidates negative scores.) The acronyms are automatically discarded.

4.3.2 Features

In SVM model, we utilize both binary and real valued features as described below. These features are created to characterize the likely acronyms, expansion candidates, and context they appear in.

- Features characterizing acronym and expansion
  Length of acronym, lower case, numeric, special characters, and white spaces in acronym are important features to determine whether it is a good acronym.
  Length of expansion, words with initial capital letters in expansion, first/last expansion words, and preposition/

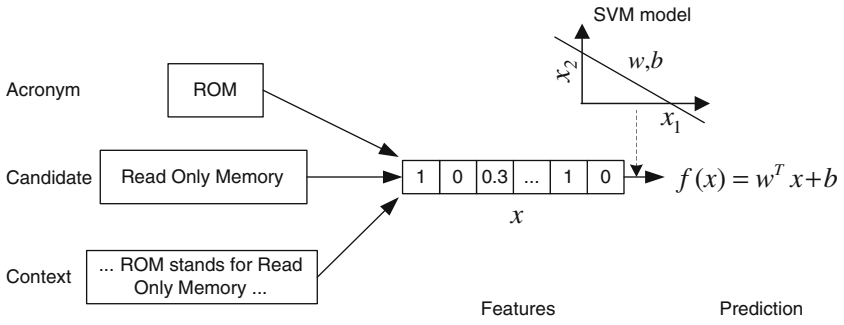

Fig. 3 Selecting genuine expansions from candidate set

conjunction in expansion are good indicators to tell whether it is a good expansion.
Relationship between acronym and expansion are considered seriously as features. E.g., if the acronym letters match with the first letters of words in expansion, it is likely the candidate is a genuine expansion for the acronym.

- Features characterizing context
We also rely upon context features. These features are, for example, most acronyms are surrounded by parenthesis and appear immediately after expansions. If acronyms already appear in preceding text, the probability that the candidate is a genuine expansion is low.

5 Advantages of SVM based approach

Heuristics rules are immediate and obvious approach to acronym extraction. However, creating and tuning patterns is boring and time consuming. Also, writing rules manually confines the usage of information. Only strong evidences can be considered in hard rules. Machine learning based approach, however, can overcome these difficulties in a natural way.

Machine learning model is built based on a collection of labeled examples. Labeling data is much easier and cheaper than writing patterns/rules.

Moreover, machine learning models can utilize variant kinds of evidences easily. In machine learning, acronyms, expansions and context they co-occur in are represented by features. Some features present strong evidences. For example, the feature defined as: whether the first characters of expansion words constitute the acronym. This kind of features can be used by both rule based methods and machine learning based approaches. Some other features represent weak evidences. For example, the feature defined as: whether the acronym appears in the preceding text. Since acronyms are often defined at their first appearance in text, the feature indicates that perhaps the candidate is not a genuine expansion. However, counterexamples do exist in real world. For example, "*The SVG DOM conforms to key aspects of Document Object Model (DOM) Level 1*." This kind of features can be utilized only by machine learning based approaches. That's why the approach proposed in the paper can outperform well for the task.

6 Experiments

We conducted experiment to verify the effectiveness of our proposed approach. Particularly, we investigated whether the problem of acronym extraction can be solved by the proposed SVM based approach. Also, we experimentally proved that the trained model is generic.

6.1 Baseline and measures for evaluation

As baseline method, we used a pattern-based approach. It is similar to AFP [10].

We made use of *precision* and *recall* for measuring performance. Equations (3) and (4) give the details.

$$\text{precision} = \frac{\#\text{ of correct acronym expansions found}}{\text{total }\#\text{ of acronym expansions found}}, \quad (3)$$

$$\text{recall} = \frac{\#\text{ of correct acronym expansions found}}{\text{total }\#\text{ of acronym expansions in documents}}. \quad (4)$$

6.2 Extracting acronyms from UCI dataset

We used the UCI dataset '*NSF Research Awards Abstracts 1990–2003*' [5] in this experiment. The dataset consists of 129,000 abstracts describing NSF awards for basic research. Acronyms and expansions appear often in these abstracts. We randomly selected 1000 documents. All occurrences of acronyms and their expansions were labeled manually. The final labeled dataset consists of 639 times of acronym/expansion co-occurrences. It was used for training and evaluation.

The dataset was split into two parts randomly: one was used for training the SVM model and another was used for testing performance. Results reported in Table 1 are performance on test set.

From Table 1, we see that our approach outperforms baseline on both precision and recall. The results indicate that the

Table 1 Performance on UCI dataset

	Precision	Recall
Baseline	0.8157	0.8201
Our approach	0.9086	0.8413

Table 2 Performance on W3C dataset

	Precision	Recall
Baseline	0.8493	0.8212
Our approach	0.8936	0.8344

proposed SVM approach is effective for the task of acronym extraction.

It is not surprising that our approach performs better than baseline. In Sect. 5, we have shown the advantages of our approach over rule-based approaches. Machine learning models can utilize both *strong* and *weak* evidences collectively. In pattern based approach, however, much *weak* but useful information is ignored.

6.3 Domain adaptation

In this experiment, we tested whether a generic model (SVM model) can be constructed for recognizing acronyms. As training data, we used the training set used in Sect. 6.2. To create the testing data, we randomly selected 100 documents from TREC W3C corpus,[2] WWW scope. All of the co-occurrences of acronym/expansion pairs in the 100 documents are labeled and the final test dataset consists of 151 times of acronym/expansion co-occurrences.

From results reported in Table 2, we see that our approach still outperforms the baseline method, though the SVM model is trained in another domain. In Sect. 4.3.2, we have described the features used in SVM. The features are domain independent. That is why we can create a domain independent generic model.

7 Conclusions

In this paper, we proposed to address the issue of acronyms extraction from free format text using SVM model. We have proposed a novel approach based on machine learning. Specifically, we first identify likely acronyms and then generate expansion candidates from text surrounding acronyms. Last, we use a SVM model to select the genuine expansions. We have shown the advantages of the proposed approach, especially that it can utilize variant evidences easily and effectively.

Experimental results show that our proposed method outperforms the baseline method of using patterns. Experimental results also show that our proposed method has good abilities of conducting domain adaptation.

[2] http://research.microsoft.com/users/nickcr/w3c-summary. html

Acknowledgements We would like to express deep gratitude to Yunhua Hu from Xi'an Jiaotong University, Weijian Ni and Yongmei Gao from Nankai University for their valuable comments. Supported by the Ministry of Education under the grant 02038, and the Asia Research Center at Nankai University under the grant AS0405.

References

1. Adar E (2004) SaRAD: a simple and robust abbreviation dictionary. Bioinformatics 20:527–533
2. Bowden PR, Automatic (1999) Glossary construction for technical papers. Department Working Paper, Nottingham Trent University
3. Bowden PR, Halstead P, Rose TG (2000). Dictionaryless English plural noun singularisation using a corpus-based list of irregular forms. In: Proceedings of the 17th international conference on English Language Research on Computerized Corpora, Rodopi, Amersterdam, The Netherlands, pp 130–137
4. Chang JT, Schutze H, Altman RB (2002) Create an online dictionary of abbreviation from MEDLINE. J Am Med Inform Assoc, 9(6):612–620
5. Hettich S, Bay SD (1999) The UCI KDD Archive. [http://kdd.ics.uci.edu]. Department of Information and Computer Science, University of California, Irvine
6. Larkey LS, Ogilvie P, Price MA, Tamilio B (2000) Acrophile: An automated acronym extractor and server. In: Proceedings of the 5th ACM conference on digital libraries. ACM Press, San Antonio, pp 205–214
7. Park Y, Byrd RJ (2001) Hybrid text mining for finding abbreviations and their definitions. In: Proceedings of the 2001 conference on empirical methods in natural language processing, Pittsburgh, pp 126–133
8. Pustejovsky J, Castano J, Cochran B, Kotecki M, Morrell M (2001) Automatic extraction of acronym-meaning pairs from MEDLINE databases. Medinfo 10(Pt 1):371–375
9. Schwartz A, Hearst M (2003) A simple algorithm for identifying abbreviation definitions in biomedical text. In: Proceedings of the 2003 pacific symposium on biocomputing. World Scientific Press, Singapore
10. Taghva K, Gilbreth J (1999) Recognizing acronyms and their definitions. Technical Report, ISRI (Information Science Research Institute), UNLV
11. Vapnik VN (1995) The nature of statistical learning theory. Springer, Berlin Heidelberg New York
12. Yeates S (1999) Automatic extraction of acronyms from text. In: Proceedings of the 3rd new zealand computer science research students' conference, University of Waikato, Hamilton, pp 117–124
13. Yeates S, Bainbridge D, Witten IH (2000) Using compression to identify acronyms in text. In: Proceedings of data compression conference, IEEE Press, New York, pp 582
14. Yoshida M, Fukuda K, Takagi T (2000) PNAD-CSS: a workbench for constructing a protein name abbreviation dictionary. Bioinformatics 16:169–175
15. Yu H, Hripcsak G, Friedman C (2002) Mapping abbreviations to full forms in biomedical articles. J Am Med Inform Assoc 9:262–272