Automatic Acronym Recognition

Dana Dannélls
Computational Linguistics, Department of Linguistics and
Department of Swedish Language
Göteborg University
Göteborg, Sweden
cl2ddoyt@cling.gu.se

Abstract

This paper deals with the problem of recognizing and extracting acronym-definition pairs in Swedish medical texts. This project applies a rule-based method to solve the acronym recognition task and compares and evaluates the results of different machine learning algorithms on the same task. The method proposed is based on the approach that acronym-definition pairs follow a set of patterns and other regularities that can be usefully applied for the acronym identification task. Supervised machine learning was applied to monitor the performance of the rule-based method, using Memory Based Learning (MBL). The rule-based algorithm was evaluated on a hand tagged acronym corpus and performance was measured using standard measures recall, precision and f-score. The results show that performance could further improve by increasing the training set and modifying the input settings for the machine learning algorithms. An analysis of the errors produced indicates that further improvement of the rule-based method requires the use of syntactic information and textual pre-processing.

1 Introduction

There are many on-line documents which contain important information that we want to understand, thus the need to extract glossaries of domain-specific names and terms increases, especially in technical fields such as biomedicine where the vocabulary is quickly expanding. One known phenomenon in biomedical literature is the growth of new *acronyms*.

Acronyms are a subset of abbreviations and are generally formed with capital letters from the original word or phrase, however many acronyms are realized in different surface forms i.e. use of Arabic-numbers, mixed alpha-numeric forms, low-case acronyms etc.

Several approaches have been proposed for automatic acronym extraction, with the most common tools including pattern-matching techniques and machine learning algorithms. Considering the large variety in the Swedish acronym-definition pairs it is practical to use pattern-matching techniques. These will enable to extract relevant information of which a suitable set of schema will give a representation valid to present the different acronym pairs.

This project presents a rule-based algorithm to process and automatically detect different forms of acronym-definition pairs. Since machine learning techniques are generally more robust, can easily be retrained for a new data and successfully classify unknown examples, different algorithms were tested. The acronym pair candidates recognized by the rule-based algorithm were presented as feature vectors and were used as the training data for the supervised machine learning system.

This approach has the advantage of using machine learning techniques without the need for manual tagging of the training data. Several machine learning algorithms were tested and their results were compared on the task.

2 Related work

The task of automatically extracting acronym-definition pairs from biomedical literature has been studied, almost exclusively for English, over the past few decades using technologies from Natural Language Processing (NLP). This section

presents a few approaches and techniques that were applied to the acronym identification task.

Taghva and Gilbreth (1999) present the Acronyms Finding Program (AFP), based on pattern matching. Their program seeks for acronym candidates which appear as upper case words. They calculate a heuristic score for each competing definition by classifying words into: (1) stop words (”the”, ”of”, ”and”), (2) hyphenated words (3) normal words (words that don't fall into any of the above categories) and (4) the acronyms themselves (since an acronym can sometimes be a part of the definition). The AFP utilizes the Longest Common Subsequence (LCS) algorithm (Hunt and Szymanski, 1977) to find all possible alignments of the acronym to the text, followed by simple scoring rules which are based on matches. The performance reported from their experiment are: recall of 86% at precision of 98%.

An alternative approach to the AFP was presented by Yeates (1999). In his program, Three Letters Acronyms (TLA), he uses more complex methods and general heuristics to match characters of the acronym candidate with letters in the definition string, Yeates reported f-score of 77.8%.

Another approach recognizes that the alignment between an acronym and its definition often follows a set of patterns (Park and Byrd, 2001), (Larkey et al., 2000). Pattern-based methods use strong constraints to limit the number of acronyms respectively definitions recognized and ensure reasonable precision.

Nadeau and Turney (2005) present a machine learning approach that uses weak constraints to reduce the search space of the acronym candidates and the definition candidates, they reached recall of 89% at precision of 88%.

Schwartz and Hearst (2003) present a simple algorithm for extracting abbreviations from biomedical text. The algorithm extracts acronym candidates, assuming that either the acronym or the definition occurs between parentheses and by giving some restrictions for the definition candidate such as length and capital letter initialization. When an acronym candidate is found the algorithm scans the words in the right and left side of the found acronym and tries to match the shortest definition that matches the letters in the acronym. Their approach is based on previous work (Pustejovsky et al., 2001), they achieved recall of 82% at precision of 96%.

It should be emphasized that the common characteristic of previous approaches in the surveyed literature is the use of parentheses as indication for the acronym pairs, see Nadeau and Turney (2005) table 1. This limitation has many drawbacks since it excludes the acronym-definition candidates which don't occur within parentheses and thereby don't provide a complete coverage for all the acronyms formation.

3 Methods and implementation

The method presented in this section is based on a similar algorithm described by Schwartz and Hearst (2003). However it has the advantage of recognizing acronym-definition pairs which are not indicated by parentheses.

3.1 Finding Acronym-Definition Candidates

A valid acronym candidate is a string of alphabetic, numeric and special characters such as '-' and '/'. It is found if the string satisfies the conditions (i) and (ii) and either (iii) or (iv):
(i) The string contains at least two characters. (ii) The string is not in the list of rejected words[1]. (iii) The string contains at least one capital letter. (iv) The strings' first or last character is lower case letter or numeric.

When an acronym is found, the algorithm searches the words surrounding the acronym for a definition candidate string that satisfies the following conditions (all are necessary in conjunction):
(i) At least one letter of the words in the string matches the letter in the acronym. (ii) The string doesn't contain a colon, semi-colon, question mark or exclamation mark. (iii) The maximum length of the string is min($|A|$+5,$|A|$*2), where $|A|$ is the acronym length (Park and Byrd, 2001). (iv) The string doesn't contain only upper case letters.

3.2 Matching Acronyms with Definitions

The process of extracting acronym-definition pairs from a raw text, according to the constraints described in Section 3.1 is divided into two steps:
1. Parentheses matching. In practice, most of the acronym-definition pairs come inside parentheses (Schwartz and Hearst, 2003) and can correspond to two different patterns: (i) definition (acronym) (ii) acronym (definition). The

[1]The rejected word list contains frequent acronyms which appear in the corpus without their definition, e.g. 'USA', 'UK', 'EU'.

algorithm extracts acronym-definition candidates which correspond to one of these two patterns.
2. Non parentheses matching. The algorithm seeks for acronym candidates that follow the constraints, described in Section 3.1 and are not enclosed in parentheses. Once an acronym candidate is found it scans the previous and following context, where the acronym was found, for a definition candidate. The search space for the definition candidate string is limited to four words multiplied by the number of letters in the acronym candidate.

The next step is to choose the correct substring of the definition candidate for the acronym candidate. This is done by reducing the definition candidate string as follows: the algorithm searches for identical characters between the acronym and the definition starting from the end of both strings and succeeds in finding a correct substring for the acronym candidate if it satisfies the following conditions: (i) at least one character in the acronym string matches with a character in the substring of the definition; (ii) the first character in the acronym string matches the first character of the leftmost word in the definition substring, ignoring upper/lower case letters.

3.3 Machine Learning Approach

To test and compare different supervised learning algorithms, Tilburg Memory-Based Learner (TiMBL)[2] was used. In memory-based learning the training set is stored as examples for later evaluation. Features vectors were calculated to describe the acronym-definition pairs. The ten following (numeric) features were chosen: (1) the acronym or the definition is between parentheses (0-false, 1-true), (2) the definition appears before the acronym (0-false, 1-true), (3) the distance in words between the acronym and the definition, (4) the number of characters in the acronym, (5) the number of characters in the definition, (6) the number of lower case letters in the acronym, (7) the number of lower case letters in the definition, (8) the number of upper case letters in the acronym, (9) the number of upper case letters in the definition and (10) the number of words in the definition. The 11th feature is the class to predict: true candidate (+), false candidate (-). An example of the acronym-definition pair $\langle$"$vCJD$","$variant\ CJD$"$\rangle$ represented as a feature vector is: 0,1,1,4,11,1,7,3,3,2,+.

[2]http://ilk.uvt.nl

4 Evaluation and Results

4.1 Evaluation Corpus

The data set used in this experiment consists of 861 acronym-definition pairs. The set was extracted from Swedish medical texts, the MEDLEX corpus (Kokkinakis, 2006) and was manually annotated using XML tags. For the majority of the cases there exist one acronym-definition pair per sentence, but there are cases where two or more pairs can be found.

4.2 Experiment and Results

The rule-based algorithm was evaluated on the untagged MEDLEX corpus samples. Recall, precision and F-score were used to calculate the acronym-expansion matching. The algorithm recognized 671 acronym-definition pairs of which 47 were incorrectly identified. The results obtained were 93% precision and 72.5% recall, yielding F-score of 81.5%.

A closer look at the 47 incorrect acronym pairs that were found showed that the algorithm failed to make a correct match when: (1) words that appear in the definition string don't have a corresponding letter in the acronym string, (2) letters in the acronym string don't have a corresponding word in the definition string, such as "PGA" from "glycol alginate lösning", (3) letters in the definition string don't match the letters in the acronym string.

The error analysis showed that the reasons for missing 190 acronym-definition pairs are: (1) letters in the definition string don't appear in the acronym string, due to a mixture of a Swedish definition with an acronym written in English, (2) mixture of Arabic and Roman numerals, such as "USH3" from "Usher typ III", (3) position of numbers/letters, (4) acronyms of three characters which appear in lower case letters.

4.3 Machine Learning Experiment

The acronym-definition pairs recognized by the rule-based algorithm were used as the training material in this experiment. The 671 pairs were presented as feature vectors according to the features described in Section 3.3. The material was divided into two data files: (1) 80% training data; (2) 20% test data. Four different algorithms were used to create models. These algorithms are: IB1, IGTREE, TRIBL and TRIBL2. The results obtained are given in Table 1.

Algorithm	Precision	Recall	F-score
IB1	90.6 %	97.1 %	93.7 %
IGTREE	95.4 %	97.2 %	96.3 %
TRIBL	92.0 %	96.3 %	94.1 %
TRIBL2	92.8 %	96.3 %	94.5 %

Table 1: Memory-Based algorithm results.

5 Conclusions

The approach presented in this paper relies on already existing acronym pairs which are seen in different Swedish texts. The rule-based algorithm utilizes predefined strong constraints to find and extract acronym-definition pairs with different patterns, it has the advantage of recognizing acronyms and definitions which are not indicated by parentheses. The recognized pairs were used to test and compare several machine learning algorithms. This approach does not requires manual tagging of the training data.

The results given by the rule-based algorithm are as good as reported from earlier experiments that have dealt with the same task for the English language. The algorithm uses backward search algorithm and to increase recall it is necessary to combine it with forward search algorithm.

The variety of the Swedish acronym pairs is large and includes structures which are hard to detect, for example: $\langle$"VF","$kammarflimmer$"$\rangle$ and $\langle$"CT","$datortomografi$"$\rangle$, the acronym is in English while the extension is written in Swedish. These structures require a dictionary/database lookup[3], especially because there are also counter examples in the Swedish text where both the acronym and the definition are in English. Another problematic structure is three letter acronyms which consist of only lowercase letters since there are many prepositions, verbs and determinates that correspond to this structure. To solve this problem it may be suitable to combine textual pre-processing such as part-of-speech annotation or/and parsing with the exiting code.

The machine learning experiment shows that the best results were given by the IGTREE algorithm[4]. Performance can further improve by modifying the input settings e.g test different feature weighting schemes, such as Shared Variance and Gain Ratio and combine different values of k for the k-nearest neighbour classifier[5].

On-going work aim to improve the rule-based method and combine it with a supervised machine learning algorithm. The model produced will later be used for making prediction on a new data.

[3]Due to short time available and the lack of resources this feature was not used in the experiment.

[4]The IGTREE algorithm uses information gain in a compressed decision tree structure.

[5]In the machine learning experiment default value is used, k=1.

Acknowledgements

Project funded in part by the SematicMining EU FP6 NoE 507505. This research has been carried out thanks to Lars Borin and Dimitrios Kokkinakis. I thank Torbjörn Lager for his guidance and encouragement. I would like to thank Walter Daelemans, Ko van der Sloot Antal van den Bosch and Robert Andersson for their help and support.

References

Ariel S. Schwartz and Marti A. Hearst. 2003. *A simple algorithm for identifying abbreviation definitions in biomedical texts*. Proc. of the Pacific Symposium on Biocomputing. University of California, Berkeley.

David Nadeau and Peter Turney. 2005. *A Supervised Learning Approach to Acronym Identification*. Information Technology National Research Council, Ottawa, Ontario, Canada.

Dimitrios Kokkinakis. 2006. *Collection, Encoding and Linguistic Processing of a Swedish Medical Corpus: The MEDLEX Experience*. Proc. of the 5th LREC. Genoa, Italy.

James W. Hunt and Thomas G. Szymanski. 1977. *A fast algorithm for computing longest common subsequences*. Commun. of the ACM, 20(5):350-353.

James Pustejovsky, José Castaño, Brent Cochran, Maciej Kotecki and Michael Morrella. 2001. *Automation Extraction of Acronym-MeaningPairs from Medline Databases*. In Proceedings of Medinfo.

Kazen Taghva and Jeff Gilbreth. 1999. Technical Report. *Recognizing Acronyms and their Definitions*. University of Nevada, Las Vegas.

Leah S. Larkey, Paul Ogilvie, Andrew M. Price and Brenden Tamilio. 2000. *Acrophile: An Automated Acronym Extractor and Server*. University of Massachusetts, Dallas TX.

Stuart Yeates. 1999. *Automatic extraction of acronyms from text*. Proc. of the Third New Zealand Computer Science Research Students' Conference. University of Waikato, New Zealand.

Youngja Park and Roy J. Byrd. 2001. *Hybrid Text Mining for Finding Abbreviations and Their Definitions*. IMB Thomas J. Watson Research Center, NY, USA.