Classifying Swedish Acronyms with MBT

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

December 21, 2005

Abstract

This paper presents a supervised machine learning approach to the acronym classification problem. A task which both difficult and crucial for many NLP applications. The experiment utilizes Memory-Based Tagger (MBT) to generate a tagger suitable for tagging Swedish acronym pairs based on a set of examples generated from the MEDLEX corpus. As a part of the experiment different feature patterns were tested. The tagger achieves accuracy = 91.8%, a result which is as good as reported from earlier experiments that have dealt with the same task. The experiment shows the flexibility of MBT and proves its ability to successfully tag acronym-pairs.

1 Introduction

An acronym is a string formed from the initial letters or syllables of other words. It is a result of taking a phrase and shortening it into a new form, thus every acronym has an expanded form (a definition). One of the main challenges of the acronym recognition task is the ability to automatically classify acronyms with their related definitions. A task which is very important and useful for many natural languages processing (NLP) applications such as: information retrieval (IR), information extraction (IE), text normalization and text mining.
One of the difficulties in classifying acronym pairs is their wide acronym formation coverage i.e. acronyms may appear in any length and may be realized in different surface forms, especially in biomedical texts where the vocabulary is quickly expanding.
In more recent developments, there have been a number of attempts to apply machine learning techniques to the acronym problem, e.g. Nadeau and Turney (2005) and Tsuruokayz et al. (2005). Most of these approaches use support vector machines (SVMs) and additional features to classify acronym pairs.
Since machine learning have the advantage of being robust and easily adaptable to new data it is a promising technique ought to be examined from different angles. This project investigates an approach of using a memory-based tagger to classify acronym-definition pairs.

2 Memory-Based Learning

Memory-Based Learning (MBL) is a machine learning paradigm that is used to detect patterns and regularities in a data. The learning process is done by reading training instances into a memory and classifying test instances by extrapolating a class from the most similar instance(s) in memory. This strategy is used by several memory-based learning algorithms, of which the most common algorithm is the nearest neighbour search. Memory-based learning (also known as instance-based learning) algorithms are sometimes referred to as "lazy" learning algorithms since they delay processing until new instances must be classified (Mitchell, 1997). The advantage of these learning algorithms is that they estimate the target function locally and differently for each new instance to be classified. Another advantage of memory-based learning is that it can handle very well domains that present exceptions and sub-regularities (usually considered noise by other machine learning techniques).
Since MBL has been applied with great success to a variety of NLP tasks and have many advantages, a different approach using this technique is examined.

2.1 TiMBL: Tilburg Memory-Based Learner

TiMBL[1] is a program which includes implementation of several memory-based learning techniques. The program stores a representation of the training set explicitly in memory and classifies new cases by extrapolation from the most similar stored cases. TiMBL is optimized for fast classification by using several indexing techniques and heuristic approximations (such as IGTREE and TRIBL). It gives access to several memory-based learning algorithms and metrics, some of which are: Information Gain weighting for dealing with features of differing importance (IB1-IG) and the Modified Value Difference metric for making graded guesses of the match between two different symbolic values.
TiMBL is freely available software and can be used for different purposes, a detailed manual (Daelemans et al., 2004) and an API (van der Sloot, 2005) provide an easy access to the contained methods, allowing users to easily get started with the program.
One way of using TiMBL is to apply a learning method to a dataset and analyze its output to extract information about the data. A particular problem can be compared and evaluated using different methods, a flexibility which makes the program suitable for selecting the most appropriate method to a certain learning task.

2.2 Memory-Based Tagger

Memory-Based Tagger (MBT)(Daelemans et al., 2003) is an automatic memory-based Part Of Speech (POS) tagger-generator (Daelemans et al., 1996). Part of speech tagging is the process of classifying words in a text with their corresponding syntactic (parts of speech) categories. MBT assigns tags to a new text by extrapolation from the most similar examples (cases) in memory. The cases are indexed using IGTREE algorithm. The examples stored in the memory (generated from the annotated data) include lexical representations for the words, their syntactic categories, previous and following context within where

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

they appear. The examples are represented by a variety of features that provide information about a focus word to be tagged, its left and right context and additional information that can be useful to provide about the known and unknown words in the corpus, a detailed description of the features are specified in table 2. The features can easily modify and different feature patterns can be chosen depending on the corpus used.
The MBT consists of a tagger generator and a tagger. The tagger generator generates a lexicon, a case base for known words and a case base for unknown words, generated from the annotated corpus, these are used (by the tagger) to tag a text with. The MBT software makes use of TiMBL, this collaboration allows to modify the settings and test different algorithms for generating a tagger. The tagger generator flexible integration of information in the case representations and the ability of choosing different algorithms, allow users to construct a tagger most suitable for the learning task.

3 The Problem

The acronym classification task can be framed in terms of a learning problem[2]. The concept we want to learn [T] is a pair (A, D) made of an acronym A (a single token) and a definition D (a sequence of one or more consecutive tokens). Given a sequence S of n tokens, S = ($s_1 \ldots s_n$), from which we wish to extract a pair (A,D). Performance [P] improves when the percent of correctly pairs classified increases. The training experience [E] is a sequence of tokens with given classified pairs.

4 The Data

The data used in this experiment consists of manually annotated set of 861 acronym-definition pairs. The set was extracted from Swedish medical texts, the MEDLEX corpus (Dimitrios, 2004). Since the tagger generator assume an input data in form of two white space separated columns, the data set was retagged. Three tags were used: 'A' for an acronym word, 'D' for a definition word and 'W' for the remaining words including punctuation numbers and other symbols which don't fall into the two first mentioned categories.
An example of a sentence taken from the original data: "...delas in i invasiv mola , koriokarcinom och $\langle def \rangle$ placental site trophoblastic tumor $\langle /def \rangle$ ($\langle acr \rangle$ PSTT $\langle /acr \rangle$) .". The sentence was tagged as follows:

delas	W
in	W
i	W
invasiv	W
mola	W
,	W
koriokarcinom	W

[2] "A well-posed learning problem is defined as follows: A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience P",(Mitchell, 1997).

och	W
placental	D
site	D
trophoblastic	D
tumor	D
(	W
PSTT	A
)	W
.	W

⟨*utt*⟩

5 Experiment and Results

To perform the experiment, the data set was divided into two files: (1) 90%, used as the training data and (2) 10%, used as the test data. In MBT, a tagger is generated by providing information about the context and the form of the words to be tagged. This is done by different feature patterns for known (p) and unknown (P) words that are given as input when generating the tagger. Feature patterns are built up as combinations of symbols that can be used for known respectively unknown words, a representation of the symbols that can be provided to the tagger generator are specified in table 2. The different feature patterns that were tested and their results are given in table 1.

Words	Features	Accuracy (%)	Correct Words	Accuracy (%)
known	ddfa	94.2	1834	89.7
unknown	dsdchFasss	80.0		
known	ddwfWa	94.6	1843	90.2
unknown	chndFasss	80.3		
known	dwdwfWaw	94.6	1851	90.6
unknown	dddFawpsss	81.7		
known	ddfWa	94.8	1856	90.8
unknown	dddFawpsss	81.9		
known	dwdwfWaw	94.7	1859	90.9
unknown	dsssFawpsss	82.8		
known	ddddfwWaaaa	95.6	1875	91.8
unknown	ddsssFawpsss	83.3		

Table 1: Memory-Based algorithm results

In total 2043 words were processed, of these 1407 detected as known words and 636 as unknown words. The best result shows accuracy = 91.8%, 1345 known words and 530 unknown words were correctly classified.
The tagging results show there were only a specified number of acronyms failed to correctly classify. Most of the errors concerned definition words, such as preposition (the, for) and other known words that during training were tagged as words (W). Wrong classification of a few acronyms occured due to the fact that these acronyms appear in the corpus more then once and are not always introduce with their definitions, in such cases when an acronym appears in the

corpus without its definition it is tagged as normal word.
The cases where the words to the left and right of each context tag where combined (i.e. dwdwfWaw) didn't seem to improve tagging. Patterns combined with the features: 'c' has-capital, 'h' has-hyphen, 'n' has-number, for unknown words didn't gave such good results.
Table 1 shows that modifications of a case representation (for known or unknown words) effects the other, such as in the third and fifth columns, where the same case for known words was used and in fourth and fifth columns, where the same case for unknown words was used.
In the experiment different frequencies and thresholds where tested, neither of those changes the results. Modifying the algorithms, the feature weighting and the distance metrics didn't lead to any improvement, in fact accuracy decreased.

An interpretation of some case representations:
The case for known words where the features are 'ddddfwWaaaa', means four disambiguated tags to the left of the focus word (dddd), the ambiguity class of the focus word (f),the corresponding word of the right tag (w), the focus word itself (W) and four ambiguous tags to the right (aaaa).
The case for unknown words where the features are 'ddsssFawpsss', means two disambiguated tags to the left (dd) of the focus word, three last letters of the word to be tagged (sss), the position of the word (F), the ambiguity class of the focus word (a), the right neighbouring word (w), first letter of the unknown word to be tagged (p) and its three suffix letters (sss).
The case for known words where the features are 'ddwfWa', means two disambiguated tags to the left (dd) of the focus word, the corresponding word of the left tag (w), the ambiguity class of the focus word (f), the focus word itself (W) and one ambiguous tag to the right (a).
The case for unknown words where the features are 'chndFasss', means has-capital (c), has-hyphen (h), has-number (n), one disambiguated tag to the left (d) of the focus word (F), one ambiguous tag to the right (a) and three suffix letters (sss).

6 Conclusions

The results show the importance of including information about the context and morphological information for the unknown words. The "winning" patterns include large information about the context, the tags surrounding the word in focus and the morphological structures of the unknown words. Considering the nature of the Swedish language it is not surprising that this information is relevant during tagging.
It is clear that rules are not learned about the relationship between an acronym and its definition, a fact that leads to wrong classification, mostly among the definitions. It might be appropriate to deal with this problem by classifying acronyms without their definitions and combine rule based methods to relate these together.
The results are just as good as reported in earlier experiments that have dealt with the same task and I believe the results can further be improve by testing different feature patterns and training the tagger on a larger corpus so that more examples can be learned and tested.

References

Walter Daelemans, Jakub Zavrel, Peter Berck, and Steven Gillis. Mbt: A memory-based part of speech tagger-generator. In *Proceedings of the Fourth Workshop on Very Large Corpora*, Eva Ejerhed and Ido Dagan, editors, pages 14–27, 1996. Copenhagen.

Walter Daelemans, Jakub Zavrel, and Antal van den Bosch. Mbt: Memory-based tagger. Technical Report version 2.0, Computational Linguistics Tilburg University, November 2003. ILK Technical Report - ILK 03-013.

Walter Daelemans, Jakub Zavrel, Ko van der Sloot, and Antal van den Bosch. Timbl: Tilburg memory-based learner. Technical Report version 5.0.1, Computational Linguistics Tilburg University, December 2004. URL `http://ilk.uvt.nl/timbl/`. ILK Technical Report - ILK 04-02.

Kokkinakis Dimitrios. Medlex. Technical Report version 1.0, Språkdata, University of Göteborg, April 2004. URL `http://demo.spraakdata.gu.se/svedk/pbl/MEDLEX_work2004.pdf`.

Tom M. Mitchell. *Machine Learning*. MIT Press and The McGraw-Hill Companies,Inc, 1997. Singapore.

David Nadeau and Peter Turney. A supervised learning approach to acronym identification. *In Proceedings 18th Conference of the Canadian Society for Computational Studies of Intelligence*, LNCS 3501, 2005. Victoria, BC, Canada.

Yoshimasa Tsuruokayz, Sophia Ananiadou, and Jun ichi Tsujiii. A machine learning approach to acronym generation. In *Proceedings of the ACL-ISMB Workshop on Linking Biological Literature, Ontologies and Databases*, Mining Biological Semantics, pages 25–31, 2005. Japan.

Ko van der Sloot. Timbl: Tilburg memory-based learner. Technical Report version 5.1, Computational Linguistics Tilburg University, January 2005. API Reference Guide.

Appendix

Table 2: Symbols that are used to build up feature patterns.

Usage	Symbol	Definition
-p or -P	d	Left context (tag)
-p or -P	a	Right context (ambitag)
-p or -P	w	Left or right context (word)
-p	f	Focus (ambitag for known words)
-p	W	Focus (word)
-P	F	Focus (position of the unknown word)
-P	c	The focus contains capitalized characters
-P	h	The focus word contains a hyphen
-P	n	The focus word contains numerical characters
-P	p	Character at the start of the word
-P	s	Character at the end of the word