

Part-of-Speech Guessing with Bogus Statistics

Michael Jackson
University of Neverland
Dept. of Joy and Sorrow / Hollywood, CA 10000
jackson@neverland.edu

June 2002

Abstract

In this paper, I present a statistical-based approach to the part-of-speech guessing problem. I see assigning a part-of-speech, such as *Adjective* or *Noun*, as a classification problem. My guessing framework, which relies on automated learning of a language model, is described in detail. The rich feature analysis presented is suitable for linguistic data, such as the ones observed in German. I use a large margin classifier learning algorithm to select relevant features and learn appropriate labelling. The system is evaluated using a German corpus.

1 Introduction

Part-of-speech guessing algorithms are designed to assign one or more possible part-of-speech categories, such as finite-verb, adjective or common noun, to a given word not yet contained in a given lexicon.

Such a task must be accomplished during part-of-speech-tagging, a process that assigns unambiguous part-of-speech tags to the words of a given sentence. Because a word usually has a variety of part-of-speech categories, tagging means much more than merely looking up the words in a lexicon. The actual category depends on the syntactic (and sometimes semantic) context. So, tagging performs morpho-syntactical disambiguation. Tagging algorithms usually access a lexicon to first assign ambiguity classes (several categories) to each word, thus reducing search space. Then, an optimal solution for the assignment is found with the help of a previously learned statistical language model or defined or learned rules.

In the following, I will first look at previous approaches and evaluate them on a theoretical basis with

regard to a rich inflectional system such as the one of German. This will lead us to types of features we need to identify when guessing the part-of-speech category of a word. I will then apply the Support Vector Machine technique to this problem in order to implement a supervised learning mechanism. The method is evaluated on a lexicon derived from the German NEGRA corpus.

2 Common Guessing Approaches

All guessing approaches described in the following use a form of rules, which are triggered by word affixes. *Morphological* rules refer to an entry in a lexicon. They account for morphological alternation, such as inflection or derivation. *Non-Morphological* rules do not refer to any lexicon entries. Thus, they can also cope with newly invented words or spelling mistakes.

Transformation-based guessing Eric Brill describes a transformation-based tagger and a transformation-based guessing mechanism. His tagger and guesser are error-driven, i.e. the rules employed in his algorithms revise in multiple iterations the previous assignment of arbitrarily chosen tags. For the guesser, this means that all unknown words may be, e.g., first tagged as ‘common noun’ (which is the most frequent unknown word). This category is then changed depending on certain preconditions stated in the rules. The rules are learned from a corpus; they are instantiations of the following templates:

- Deleting the prefix/suffix x , $|x| < 4$ results in a known word.

- The first/last n ($1 < n < 4$) characters of the word are x .

Learning is implemented as follows: The initial corpus is annotated using the initial algorithm. Then, the automatically assigned tags are compared to the true ones from the corpus. The learner instantiates the transformation templates to construct transformations that correct the mistakes made by the previous annotator. Then, the transformations are scored; best-scoring ones are chosen and applied. Then, the comparison to the truthful corpus data starts again, until no other successful transformation is found.

3 Particularities of German

Several linguistic phenomena in German suggest a modification of the guessing strategies described above. German morphological alternation comprises deviation, composition as well as morpho-syntactical inflection, such as case or number marking.

- Inflection via suffixes is not monotonous, meaning, it will not only add a marker to the lexical item, but do so just after removing some inflectional suffix. In other words: the pure stem form does not occur in a corpus, or, only occurs with low frequency.
- Compounds are a very productive morphological word-class in German, i.e. Germans love to create new words. In contrast to English, composition is not clearly marked (as, e.g., with a dash).

4 A Statistical Approach

The statistical algorithm I will present in the following involves a language model that can encode several relevant features. To combine these features, I will discuss the use of a set of Support Vector Machines for classification and compare my results to the ones with a much simpler voting system. My language model is acquired by supervised learning. The algorithm returns, in contrast to the approaches shown before, a vector containing a probability for the assignment of each tag.

4.1 Features in categorization

Statistical categorization, such as part-of-speech guessing, involves finding a class assumption about a class

of given data C_w . A straightforward, probabilistic approach would be to find the conditional (maximum-likelihood) probability.

4.1.1 Identification of Feature Types

The set of possible features (*feature space*) needs to be determined before learning. I will first informally describe the types of information to be considered during the learning process. Some parameters are considered to have significant predictive capabilities. We will call them *feature types* as opposed to *features*.¹

I found statistical evidence for a good recall of the markers ‘ge’, ‘-’ (χ^2 estimate shows significant marker-tag correlation). If taken as single clue, either precision is rather low (as in ‘ge’), or they only give us information we already have other indicators for (‘-’ for nouns, which can be recognized by word capitalization). If employed in a statistical model that can deal with interdependent features, markers do improve overall performance. The performance with a small subset of the training/test-corpus and a support vector model went up from .69-.56 (precision/recall) to .76-.68 with marker feature. A prominent German example of this are infinite verbs with ‘zu’, which can be recognized much better with the ‘zu’ marker.

4.1.2 Filtering features

The frequency of a word (if available in the training data) may serve as a filter. Filtering out frequent items has proven to increase model performance. I explain this phenomenon by a tendency of languages to lexicalize forms with high frequency; thus these forms are not as regular and won’t generalize well. From the NEGRA corpus, only samples with a maximal frequency of 2 are learned.

4.2 Category selection

It seemed quite helpful to combine some classes into one, if we know a priori that features are not distributed over these classes in any way that would serve a good prediction. For example, this seems to be the case for different types of adjectives in German: Attributive, predicative and adverbial adjectives have empty markers in German (or go unmarked, depending on your favorite linguistic theory.) Other tags were considered to

¹The feature type ‘suffix’ subsumes instances like -en, -st, -chen, which we call *features* following machine learning terminology.

be of closed-classes (such as modal verbs or prepositions) and thus covered entirely by a lexicon with no need for guessing. In case they are needed, they may be expanded after the guessing process.

4.3 Learning

I understand learning as the process of finding parameters (i.e. the language model M) such that the return value of some imaginary error function $e(G_M, R)$ is minimal. This function compares the performance of two functions: G_M , my guessing function using the model M , and R , the function delivering the ‘good’ standard. W be the set of words, C be the set of categories:

$$G : W \rightarrow \{\vec{v} | \vec{v} \text{ is a vector of length } |C|\}$$

$$R : W \rightarrow \{c^* | c \in C\}$$

The error function depends on the specific algorithm used to realize learning.

I adopt a supervised learning algorithm: it expects an annotated corpus, i.e. a lexicon with a set of part-of-speech tags assigned to each entry. The learning algorithm creates a data structure (*Language Model*) to be used in guessing.

Learning is a two-step process: First, a set of possible features is identified. In particular, affixes are identified. Then, each sample from the training corpus is analyzed in terms of values of present features. Out of a relatively high number of binary features (in P.O.S. guessing, every affix is seen as feature), only a few are present, thus get the value 1. Written as (sparse) vector, each sample describes a point in a k-dimensional space, where k is the number of possible features. used as input to the learning algorithm.

In the following, we describe the use of Support Vector Machines as models for training in comparison to a simpler voting mechanism.

4.3.1 Voting

Observed features were first selected using a frequency threshold. The same features as shown in ?? were used, however, they were scalar instead of binary, and the feature vectors were of length $|C|$. Feature weights in the feature vector for a given sample and a certain class contained $P(C|F_n)$. Each feature observed in a word

may be seen as its own classifier with a confidence measure for each class. A combination algorithm lets these classifiers vote: This combination algorithm tries to estimate $P(C|F_1 \cap F_2 \cap \dots F_n)$:

$$P(C|\bigcap_j F_j) \approx P'_C = \frac{1}{\sum_j \lambda_j} \sum_j \lambda_j \hat{P}(C|F_j)$$

The λ weights are set manually. For the resulting P'_C , we still need a decision function to get a binary result: Does w belong to C ? The decision predicate applies iff $P' > \theta_C$. θ_C as separation constant (or: threshold) is estimated automatically: The average value of P'_C is calculated among the positive and the negative examples. θ_C is the mean of those two average values.

4.3.2 Support Vector Classifiers

Instead of describing the intricacies of Support Vector Machines (Vapnik 1995), I will simply refer to a nice colorful diagram (Figure 1).

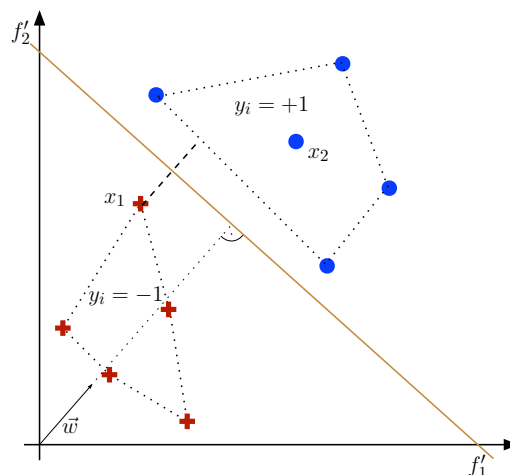


Figure 1: That’s a piece of cake, said the support vector machine and defined a beautiful function to separate all these weird data.

4.4 Evaluation

The SVM approach and the voting method were evaluated against each other with a lexicon with 30.900 entries extracted from NEGRA, a manually annotated

Tag	Baseline	Voting	SVM
Noun	.42-1.0-.42	.98-.99-.98	1.0-1.0-1.0
Adj	0-0-.56	.88-.98-.93	.93-.94-.94
Verb-fin	0-0-.89	.54-.41-.89	.70-.83-.94
Verb-inf-1	0-0-.94	.78-.39-.95	.64-.59-.95
Verb-inf-2	0-0-.99	.42-.30-.99	1.0-.51-.99
Verb-part.	0-0-.96	.75-.58-.97	.75-.72-.98
Total	.42-.40-.80	.89-.88-1.00	.91-.93-.97

Table 1: Guessing performance on German NEGRA corpus (35 000 tokens). Values denote precision, recall and accuracy.

German newspaper corpus (Skut et al. 1997) and a French corpus with 20.000 entries.

I gained the language model from 85 percent of the lexicon and evaluated with 15 percent.

The French corpus run shows that SVMs and their training algorithm seem to be successful in learning feature interdependencies, but won't improve performance with very sparse vectors: In the French corpus, mostly suffixes seem to be of importance.

Compared to other work, the performance of SVMs on German P.O.S. guessing can compete with published results for different languages. Daciuk et al. (1998) states up to 91.64 percent precision, 93.92 percent recall on a French corpus. Daciuk's reimplementation of Mikheev (1997) (a method that resembles my approach) showed 88.27 percent precision, 92.47 percent recall on the same corpus.

The algorithms presented perform worse on the same French corpus. This may be due to the fact that I did only little parameter-tuning for the corpora. Also, my (French) corpus might have been smaller than the one used in Daciuk's experiments. Most importantly, SVMs cannot take advantage of their feature-integration capabilities.

The results show that guessing difficulty differs greatly among languages. My initial understanding that German is much harder in P.O.S. analysis is confirmed.

Training times for the model based on SVMLight was 1m37s, for SVMTorch it was 0m50s.²

Contribution of single features My experiments with disabling some of the feature types show that, for SVMs, the single letter feature could basically do a

²On an Athlon 1.4GHz, 2GB RAM machine running Linux & GCC.

huge part of the job. The learning algorithm recognizes combined features for itself. Prefixes and suffixes do not contribute significantly to the performance of the guessers. Morphological features, however, do add valuable information.

5 Conclusion

I described the use of advanced statistical models in part-of-speech guessing. They improve guessing performance for German significantly. SVMs showed the properties described in the literature. The major drawback of SVMs is a bad time complexity in training. But besides the ability to handle highly-dimensional features spaces, their performance did not decrease significantly when adding non-relevant features. This way, they can successfully integrate features that are only relevant to certain classes. Thus, support vector models make an excellent classifier in every natural language classification task, where diverse features need to be integrated.

References

- Brill, E. (1993), A Corpus-Based Approach to Language Learning, PhD thesis, Philadelphia, PA.
- Daciuk, J., Watson, B. W. & Watson, R. E. (1998), Incremental construction of minimal acyclic finite state automata and transducers, *in* L. Karttunen, ed., 'FSMNLP'98: International Workshop on Finite State Methods in Natural Language Processing', Association for Computational Linguistics, Somerset, New Jersey, pp. 48–55.
- Mikheev, A. (1997), 'Automatic rule induction for unknown-word guessing', *Computational Linguistics* **23**(3), 405–423.
- Skut, W., Krenn, B., Brants, T. & Uszkoreit, H. (1997), 'An annotation scheme for free word order languages'.
- Vapnik, V. N. (1995), *The nature of statistical learning theory*, New York: Springer.