# Assigning Function Tags with a Simple Model

Vasile Rus and Kirtan Desai

Department of Computer Science, The University of Memphis,
Institute for Intelligent Systems, Fedex Institute of Technology,
Memphis, TN 38120, USA
vrus@memphis.edu

**Abstract.** This paper presents a method to assign function tags based on a Naive Bayes approach. The method takes as input a parse tree and labels certain constituents with a set of functional marks such as logical subject, predicate, etc. The performance reported is promising, given the simplicity of a Naive Bayes approach, when compared with similar work.

## 1 Introduction

Syntactic structure is an important phase in a variety of language tasks since it provides important information for semantic interpretation. State-of-the-art statistical parsers, the tools that generate syntactic structures, are freely available nowadays but their output is limited to basic structures and are not able to deliver richer syntactic information such as logical subject or predicate. Most of the available statistical parsers are trained on Penn Treebank [7] and are only able to identify simple phrases such as NP, VP or S although Penn Treebank contains function tags and remote dependencies coded as traces. This paper presents a naive Bayes approach to augment the output of Treebank-style syntactic parsers with functional information.

In Section 2.2 of Bracketing Guidelines for Treebank II [7], there are 20 function tags grouped in four categories: form/function discrepancies, grammatical role, adverbials, and miscellaneous. Up to 4 function tags can be added to the standard syntactic label (NP, ADVP, PP, etc.) of each bracket. Those tags were necessary to distinguish words or phrases that belong to one syntactic category and is used for some other function or when it plays a role that is not easily identified without special annotation. We rearrange the four categories into four new categories based on corpus evidence, in a way similar to [1]. The new four categories are given in Table 1 and were derived so that no two labels from same new category can be attached to the same bracket.

We present in this paper a naive Bayes approach to build a system that automatically assigns function tags to constituents in parse trees. The function tags assignment problem is viewed as a classification problem, where the task is to select the correct tag from a list of candidate tags. The results are reported per category based on the new categories mentioned above.

Simple Bayesian classifiers have been gaining popularity lately, and have been found to perform surprisingly well [3]. These probabilistic approaches make

A. Gelbukh (Ed.): CICLing 2005, LNCS 3406, pp. 112–115, 2005.

**Table 1.** Categories of Function Tags

| Category | Function Tags |
|---|---|
| Grammatical | DTV, LGS, PRD, PUT, SBJ, VOC |
| Form/Function | NOM, ADV, BNF, DIR, EXT, LOC, MNR, PRP, TMP |
| Topicalisation | TPC |
| Miscellaneous | CLR, CLF, HLN, TTL |

strong assumptions about how the data is generated, and posit a probabilistic model that embodies these assumptions; then they use a collection of labeled training examples to estimate the parameters of the generative model. Classification of new examples is performed with Bayes' rule by selecting the class that is most likely to have generated the example. The naive Bayes classifier is the simplest of these models, in that it assumes that all attributes of the examples are independent of each other given the context of the class. This is the so-called "naive Bayes assumption". While this assumption is clearly false in most real-world tasks, naive Bayes often performs classification very well. This paradox is explained by the fact that classification estimation is only a function of the sign (in binary cases) of the function estimation; the function approximation can still be poor while classification accuracy remains high [3]. Because of the independence assumption, the parameters for each attribute can be learned separately, and this greatly simplifies learning, especially when the number of attributes is large [6].

## 2 Related Work

There has been no previous work, to our knowledge, so far that attempted to build a system that assigns function tags using a naive Bayes approach.

There was only one project detailed in [1] to address the task of function tagging. They use a statistical algorithm based on a set of features grouped in *trees*, rather than *chains*. The advantage is that features can better contribute to overall performance for cases when several features are sparse. When such features are conditioned in a chain model the sparseness of a feature can have a dilution effect of a ulterior (conditioned) one.

Previous to that, Michael Collins [2] only used function tags to define certain constituents as complements. The technique was used to train an improved parser.

Related work on enriching the output of statistical parsers, with remote dependency information, were exposed in [5], [4].

## 3 The Model

Our approach is to map the function tags assignment task into a classification task and then use a naive Bayes model to build a classifier for it. Classifiers are

programs that assign a class from a predefined set to an instance or case under consideration based on the values of attributes used to describe this instance. Naive Bayes classifiers use a probabilistic approach, i.e. they try to compute a conditional distribution of classes and then predict the most probable class.

## 4 Experimental Setup and Results

We trained our model on sections 1-21 from Wall Street Journal (WSJ) part of Penn Treebank. The set of attributes/features used was automatically extracted from trees together with their classification. In those experiments punctuation was mapped to a unique tag PUNCT and traces were left unresolved and replaced with TRACE. We used a set of features inspired from [1] that includes the following: label, parent's label, right sibling label, left sibling label, parent's head pos, head's pos, grandparent's head's pos, parent's head, head. We did not use the alternative head's pos and alternative head (for prepositional phrases that would be the head of the prepositional object) as explicit features but rather modified the phrase head rules so that the same effect is captured in pos and head features, respectively. A simple add-one smoothing method was used.

To generate the training data, we only considered nodes with functional tags, ignoring constituents unlabeled with such tags. Since a node can have several tags a training example is generated for each tag. There are two types of experiments we played with: (1) each instance is assigned a single tag from the joint set of all categories (2) each instance is assigned a tag from each of four categories. While the first type is more convenient the second is similar to what Treebank does, i.e. assigning tags from multiple categories.

### 4.1 Results

The results in Table 2 were obtained by testing the Naive Bayes classifier on section 23 from WSJ in Treebank 2. The performance measure reported is precision, defined as the number of correctly tagged test instances divided by the number of attempted instances. Since the input was a perfectly parsed tree the results are an accurate measurement of the actual potential of Naive Bayes for the function tags assignment task. Blaheta [1] parses the test section 23 using a state-of-the-art parser and considers only correct constituents in the output when reporting results of the functional tags assignment classifier. Our method provides state-of-the-art results. Another advantage of our method is its simplicity. For the Topicalisation category the Naive Bayes approach provides perfect tagging (100% accuracy) for the second type of experiments due mainly to the fact that the *Topicalisation* category contains a single possible tag. The precision is considerably higher for the second type of experiment (see last column in the table).

**Table 2.** Performance Measures

| Category | Performance (%) | |
|---|---|---|
| | Exp 1 | Exp 2 |
| All Categories | 94.12 | - |
| Grammatical | 97.04 | 97.91 |
| Form/Function | 51.97 | 59.22 |
| Topicalisation | 1.87 | 100 |
| Miscellaneous | 66.93 | 93.67 |

## 5 Conclusion

We presented in this paper a Naive Bayes approach to the task of assigning function tags to constituents in parse trees. Our experiments show that the method is robust enough to offer competitive performance for the Grammatical and Miscellaneous categories of function tags. The results reported are on perfectly parsed trees.

## References

1. Blaheta, D., Johnson, M. Assigning Function Tags to Parsed Text Proceedings of the 1st Annual Meeting of the North American Chapter of the Association for Computational Linguistics, Seattle, May 2000, pp. 234-240
2. Collins, M. Three Generative, Lexicalised Models for Statistical Parsing Proceedings of the Thirty-Fifth Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics
3. Friedman, J. On bias, variance, 0/1 - loss, and the curse-of-dimensionality. Data Mining and Knowledge Discovery, 1:55-77, 1997
4. Jijkoun, V. de Rijke, M. Enriching the Output of a Parser Using Memory-Based Learning. Proceedings of the ACL 2004
5. Johnson, M. A simple pattern-matching algorithm for recovering empty nodes and their antecedents Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics
6. McCallum, A., Nigam, K. A Comparison of Event Models for Naive Bayes Text Classification Workshop on Learning for Text Categorization, AAAI (1998)
7. Bies, A., Ferguson, M., Katz, K., MacIntyre, R. Bracketing Guidelines for Treebank II Style. Penn Treebank Project.