Tagging English by Path Voting Constraints

Gökhan Tür and Kemal Oflazer
Department of Computer Engineering and Information Science
Bilkent University, Bilkent, Ankara, TR-06533, TURKEY
{tur,ko}@cs.bilkent.edu.tr

January 21, 1998

Abstract:We describe a constraint-based tagging approach where individual constraint rules vote on sequences of matching tokens and tags. Disambiguation of all tokens in a sentence is performed at the very end by selecting tags that appear on the path that receives highest vote. This constraint application paradigm makes the outcome of the disambiguation *independent* of the rule sequence, and hence relieves the rule developer from worrying about potentially conflicting rule sequencing found in other systems. The approach can also combine statistically and manually obtained constraints, and incorporate negative constraint rules that rule out certain patterns. We have applied this approach to tagging English text from the Wall Street Journal and the Brown Corpora. Our results from the Wall Street Journal Corpus indicate that with 400 statistically derived constraint rules and about 657 hand-crafted constraint rules, we can attain an *average accuracy of 97.56%* on the training corpus and an *average accuracy of 97.12%* on the testing corpus with 11-fold cross-validation. We can also relax the single tag per token limitation and allow ambiguous tagging which lets us trade recall and precision.

1 Introduction

Tagging is one of the preliminary steps in many natural language processing systems in which the proper part-of-speech tag of the tokens comprising the sentences are disambiguated using either statistical or symbolic local contextual information. There has been a large number of studies in tagging using various techniques. Part-of-speech tagging systems have used either a statistical approach where a large corpora has been used to train a probabilistic

model which then is used to tag unseen text, (e.g., Church [4], Cutting et al. [5], DeRose [6]), or a constraint-based approach which employs a large number of hand-crafted linguistic constraints that are used to eliminate impossible tags or morphological parses for a given word in a given context, recently most prominently exemplified by the Constraint Grammar work [7, 11, 12, 13]. Brill [1, 2, 3] has presented a transformation-based learning approach, which induces tagging rules from tagged corpora.

This paper presents a novel approach to constraint-based tagging which relieves the rule developer from worrying about conflicting rule ordering requirements and constraints. The approach depends on assigning votes to constraints via statistical and/or manual means, and then letting constraints vote on matching sequences on tokens. This approach does not reflect the outcome of matching constraints to the set of morphological parses immediately as usually done in constraint-based systems. Only after all applicable rules are applied to a sentence, all tokens are disambiguated in parallel. Thus, the outcome of the rule applications is *independent* of the order of rule applications.

In the following sections we describe tagging by voting constraint rules and then present the results from tagging English.

2 Tagging by Path Voting Constraints

This section outlines our approach to constraint-based tagging where constraints vote on matching parses of sequential tokens. We assume that sentences are delineated and that each token is assigned all possible tags by a lexicon or by a morphological analyzer. We represent each sentence as a standard chart using a directed acyclic graph (DAG) where nodes represent token boundaries and arcs are labeled with ambiguous interpretations of tokens. For instance the sentence

```
I can can the can.
```

would be represented by the graph shown in Figure 1 where bold arcs denote the correct tags.

We describe constraints on the ambiguous interpretation of tokens using rules with two components

$$R = (C_1, C_2, \cdots, C_n; V)$$

where the C_i are, in general, feature constraints on a sequence of the ambiguous parses, and V is an integer denoting the vote of the rule. For English, the features that we use are:

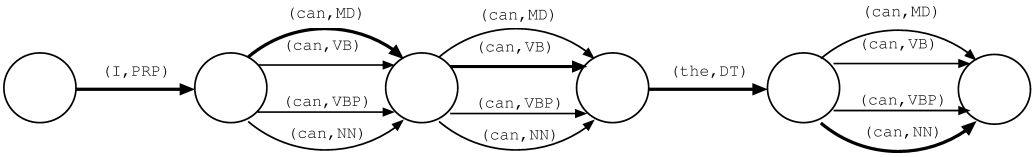

Figure 1: Representing Sentences with a directed acyclic graph

1. `LEX`: the lexical form,
2. `TAG`: the tag.

It is certainly possibly to extend the set of features used, by including features such as initial letter capitalization, any derivational information, etc. Figure 2 highlights the voting constraints paradigm.

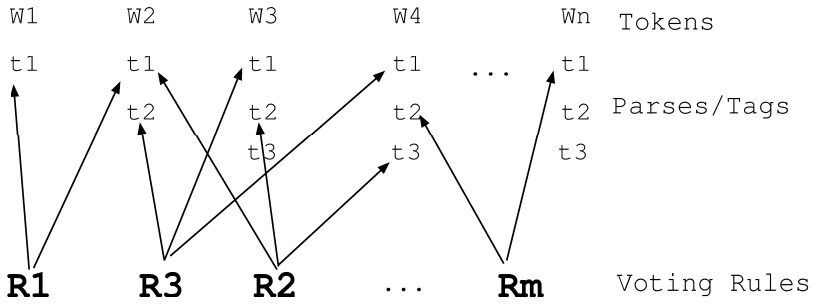

Figure 2: Voting Constraint Rules

The following examples illustrate some rules:

1. `([TAG=MD], [TAG=VB]; 100)` and `([TAG=MD], [TAG=RB], [TAG=VB]; 100)` are two constraints with a high vote to promote modal followed a verb possibly with an intervening adverb.
2. `([TAG=DT ,LEX=that], [TAG=NNS]; -100)` demotes a singular determiner reading of `that` before a plural noun.
3. `([TAG=DT, LEX=each], [TAG=JJ, LEX=other]; 100)` is a rule with a high vote that captures a collocation [8].

The constraints apply to a sentence in the following manner: Assume for a moment all possible paths from the start node to the end node of a

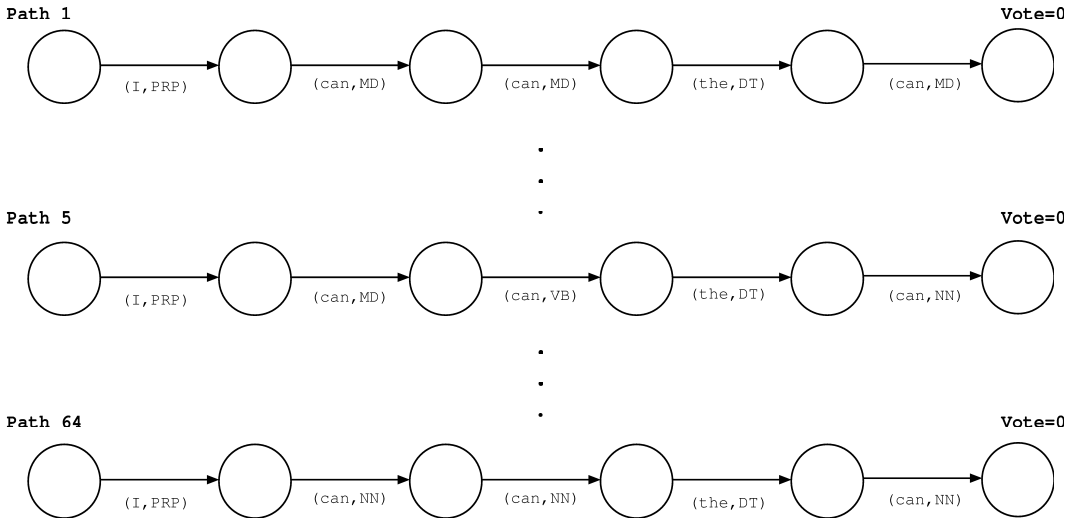

Figure 3: All paths in the Sentence DAG

sentence DAG are explicitly enumerated, and that after the enumeration of these paths, paths are augmented by a vote. In Figure 3, these initial votes are shown to be 0, they in fact would be initialized to the sum of lexical votes for the token/tag combinations on the path, extracted from a training corpus, as explained later.

For each path, we apply each constraint to all possible sequences of token parses. For instance let $R = (C_1, C_2, \cdots, C_n; V)$ be a constraint and let $w_i, w_{i+1}, \cdots, w_{i+n-1}$ be a sequence of token parses labeling sequential arcs of the path. We say R matches this sequence of parses, if $w_j, i \leq j \leq i+n-1$ is subsumed by the corresponding constraint $\mathtt{C}_{j-i-1}$. When such a rule matches, the vote of the path is incremented by V. When all constraints are applied to all possible sequences in all paths, we select the path(s) with the maximum vote. If there are multiple paths with the same maximum vote, the tokens whose parses are different in these paths are assumed to be left ambiguous.

Given that each token has on the average more than 2 possible tags, the procedural description above is very inefficient for all but very short sentences. However, the observation that our constraints are localized to a window of a small number of tokens (say at most 5 tokens in a sequence), suggests a more efficient scheme originally used by Church [4].

Assume our constraint windows are allowed to look at a window of at most size k sequential parses. Let us take the first k tokens of a sentence

and generate all possible paths of k arcs (spanning $k+1$ nodes), and apply all constraints to these "short" paths. Now, if we discard the first token and consider the $(k+1)^{st}$ token, we only need to consider and extend only those paths that have accumulated the maximum vote among paths whose last $k-1$ parses are the same. The reason for this is that since the first token is now out of the context window, it can not influence the application of any rules, hence only the highest scoring (partial) paths need to be extended, as lower scoring paths can not later accumulate votes to surpass the current highest scoring paths.

We can describe the procedure in a more formal way as follows: Let $w_1, w_2, \cdots, w_s$ is a sequence of sentence tokens, $amb(w_i)$ be the number of ambiguous tags for token w_i, and k be the maximum context window size (determined at run time). The procedure then is:

1. $P = \{$ all $\prod_{j=1}^{k-1} amb(w_j)$ paths of the first $k-1$ tokens $\}$
2. $i = k$
3. `while` $i \leq s$
4. `begin`

   4.1) Create $amb(w_i)$ copies of each path in P and extend each such copy with one of the distinct tags for token w_i.

   4.2) Apply all constraints to the last k tokens of every path in P, updating path votes accordingly. (The rules are indexed by the tag of their last constraint for fast access!)

   4.3) Remove from P any path p if there is some other path p' such that $vote(p') > vote(p)$ and the last $k-1$ tags of path p are same as the last $k-1$ tags of p'.

   4.4) $i = i + 1$

   `end`

3 Results from Tagging English

We have evaluated our approach using 11-fold cross validation on the Wall Street Journal Corpus and 10-fold cross validation on a portion of the Brown Corpus from the Penn Treebank CD.

We used two classes of constraints: (i) we extracted a set of tag k-grams from a training corpus and used them as constraint rules with votes assigned

as described later below, and (ii) we hand-crafted a set rules mainly incorporating negative constraints (demoting impossible or unlikely situations), or *lexicalized* positive constraints. These were constructed by observing the failures of the statistical constraints on the training corpus and fixing them.

Rules derived from the training corpus For the statistical constraint rules, we extracted tag k-grams from the tagged training corpus for $k = 2$, and $k = 3$. For each tag k-gram, we computed a vote which is essentially very similar to the rule strengths used by Tzoukermann et al.[9] except that we do not use their notion of genotypes exactly in the same way. Given a tag k-gram $t_1, t_2, \ldots t_k$, let $n = count(t_1 \in Tags(w_i), t_2 \in Tags(w_{i+1}), \ldots, t_k \in Tags(w_{i+k-1}))$ for all possible i's in the training corpus, be the number of possible places the tags sequence *can* possibly occur. Here $Tags(w_i)$ is the set of tags associated with the token w_i. Let f be the number of times the tag sequence $t_1, t_2, \ldots t_k$ actually occurs in the tagged text, that is $f = count(t_1, t_2, \ldots t_k)$. We smooth f/n by defining $p = \frac{f+0.5}{n+1}$ so that neither p nor $1-p$ is zero. The uncertainty of p is given as $\sqrt{p(1-p)/n}$ [9]. We then compute the vote for this k-gram as

$$Vote(t_1, t_2, \ldots t_k) = (p - \sqrt{p(1-p)/n}) * 100.$$

This formulation thus gives high votes to k-grams which are selected most of the time they are "selectable." And, among the k-grams which are equally good (same f/n), those with a higher n (hence less uncertainty) are given higher votes.

After extracting the k-grams as described above for $k = 2$ and $k = 3$, we ordered each group by decreasing votes and did an initial set of experiments with these, to select a small group of constraints performing satisfactorily. We selected the first 200 (with highest votes) of the 2-gram and the first 200 of the 3-gram constraints, as the set of statistical constraints. It should be noted that the constraints obtained this way are purely constraints on tag sequences and do not use any lexical or genotype information.

Hand-crafted rules In addition to these statistical constraint rules, we introduced 657 hand-crafted constraint rules. Most of the hand-crafted constraints imposed negative constraints (with large negative votes) to rule out certain tag sequences that we encountered in the Wall Street Journal Corpus. Another set of rules were lexicalized rules involving the tokens as well as the tags. A third set of rules were for idiomatic constructs and collocations

was also used. The votes for negative and positive hand-crafted constraints are selected to override any vote the statistical constraints may have.

Initial Votes To reflect the impact of lexical frequencies we initialize the total vote of each path with the sum of the lexical votes for the token and tag combinations on it. These lexical votes for the parse $t_{i,j}$ of token w_i are obtained from the training corpus in the usual way, i.e., as $count(w_i, t_{i,j})/count(w_i)$ and then are normalized to between 0 and 100.

Experiments on WSJ and Brown Corpora We tested our approach on two English Corpora from the Penn Treebank CD. We divided a 5500 sentence portion of the Wall Street Journal Corpus into 11 different sets of training texts (with 5000 sentences and more than 118,500 words on the average), and corresponding testing texts (with 500 sentences and more than 11,800 sentences on average), and then tagged these texts using the statistical rules and hand-crafted constraints. The hand-crafted rules were obtained from only one of the training text portions, and not from all, but for each experiment the 400 statistical rules were obtained from the respective training set.

We also performed a similar experiment with a portion of the Brown Corpus. We used 4000 sentences (about 100,000 words) with 10-fold cross validation. Again we extracted the statistical rules from the respective training sets, but the hand-crafted rules were the ones developed from the Wall Street Journal training set. For each case we measured the *accuracy* by counting the correctly disambiguated tokens.

Table 1 presents a set of tagging results for this case from the 11-fold experimentation on the Wall Street Journal Corpus. These results are the average of the 11 experiments done on 11 different training and testing texts.

Table 2 presents the tagging results from the experimentation on the Brown Corpus.

We feel that the results in the last rows of the Tables 2 and 1 are quite satisfactory and warrant further extensive investigation. On the Wall Street Journal Corpus, our tagging approach is on par or even better than stochastic taggers making closed vocabulary assumption. Weischedel et al. [14] report a 96.7% accuracy with 1,000,000 words of training corpus. Our results are very close to that of Brill's transformation-based tagger which can reach 97.2% accuracy with closed vocabulary assumption and 96.5% accuracy with open vocabulary assumption with no ambiguity [3]. Our tagging

Constraint Set	Train. Set Accuracy	Test Set Accuracy
1	95.47	94.30
1+2	96.35	95.48
1+3	96.28	95.14
1+2+3	96.53	95.73
1+4	96.86	96.24
1+2+4	97.52	97.06
1+3+4	97.30	96.66
1+2+3+4	97.56	97.12

`(1) Lexical Votes (2) 200 2-grams (3) 200 3-grams (4) 657 Manual Constraints`

Table 1: Results from tagging the Wall Street Journal with both statistically and manually derived voting constraints rules

speed is also quite high. With over 1000 constraint rules (longest spanning 5 tokens) loaded, we can tag at about 1600 tokens/sec on a Ultra Sparc 140, or a Pentium 200.

It is also possible for our approach to allow for some ambiguity. In the procedure given earlier, in item 4.3, if one selects all (partial) paths whose accumulated vote is within p $(0 < p \leq 1)$ of the (partial) path with the largest vote, then a certain amount of ambiguity can be introduced, at the expense of a slowdown in tagging speed and and an increase in memory requirements. In such a case, instead of accuracy, one needs to use *ambiguity*, *recall* and *precision* defined as follows:[10]:

$$Ambiguity = \frac{\#Parses}{\#Tokens}$$

$$Recall = \frac{\#Tokens\ Correctly\ Disambiguated}{\#Tokens}$$

$$Precision = \frac{\#Tokens\ Correctly\ Disambiguated}{\#Parses}$$

Table 3 presents the results from tagging one of the Wall Street Journal test sets using the same set of constraints but with p ranging from 95% to 99%. These compare quite favorably with the k-best results of Brill[3], but reduction in tagging speed is quite noticeable, especially for lower p's. Any improvements in single tag per token tagging (by additional hand crafted constraints) will certainly be reflected to these results also.

Constraint Set	Train. Set Accuracy	Test Set Accuracy
1	95.75	94.25
1+2	96.78	95.76
1+3	96.50	95.10
1+2+3	96.91	96.02
1+4	96.11	95.30
1+2+4	97.07	96.50
1+3+4	96.74	96.19
1+2+3+4	97.11	96.57

`(1) Lexical Votes (2) 200 2-grams (3) 200 3-grams (4) 657 Manual Constraints`

Table 2: Results from tagging the Brown Corpus with both statistically and manually derived voting constraints rules

p	Test Set		
	Recall	Precision	Ambiguity
0.99	97.97	96.22	1.018
0.98	98.35	94.76	1.038
0.97	98.61	92.70	1.063
0.96	98.77	90.67	1.089
0.95	98.87	88.95	1.111

Table 3: Recall and precision results on a test set with some tokens left ambiguous

4 Conclusions

We have presented an approach to constraint-based tagging that relies on constraint rules voting on sequences of tokens and tags. This approach can combine both statistically and manually derived constraints, and relieves the rule developer from worrying about rule ordering as removal of tags is not immediately committed but only after all rules have a say. Using positive or negative votes, we can promote meaningful sequences of tags or collocations, or demote impossible tags. Our approach is quite general and is applicable to any language. Our results from the Wall Street Journal Corpus indicate that 400 statistically derived constraint rules and about 657 hand-crafted

constraint rules, we can attain an *average accuracy of 97.56%* on the training corpus and an *average accuracy of 97.12%* on the testing corpus. We can also relax the single tag per token limitation and allow ambiguous tagging which lets us trade recall and precision. Our future work involves extending to open vocabulary case and evaluating unknown word performance.

References

[1] Eric Brill. A simple-rule based part-of-speech tagger. In *Proceedings of the Third Conference on Applied Natural Language Processing*, Trento, Italy, 1992.

[2] Eric Brill. Some advances in rule-based part of speech tagging. In *Proceedings of the Twelfth National Conference on Articial Intelligence (AAAI-94)*, Seattle, Washinton, 1994.

[3] Eric Brill. Transformation-based error-driven learning and natural language processing: A case study in part-of-speech tagging. *Computational Linguistics*, 21(4):543–566, December 1995.

[4] Kenneth W. Church. A stochastic parts program and a noun phrase parser for unrestricted text. In *Proceedings of the Second Conference on Applied Natural Language Processing*, Austin, Texas, 1988.

[5] Doug Cutting, Julian Kupiec, Jan Pedersen, and Penelope Sibun. A practical part-of-speech tagger. In *Proceedings of the Third Conference on Applied Natural Language Processing*, Trento, Italy, 1992.

[6] Steven J. DeRose. Grammatical category disambiguation by statistical optimization. *Computational Linguistics*, 14(1):31–39, 1988.

[7] Fred Karlsson, Atro Voutilainen, Juha Heikkilä, and Arto Anttila. *Constraint Grammar-A Language–Independent System for Parsing Unrestricted Text.* Mouton de Gruyter, 1995.

[8] Beatrice Santorini. Part-of-speech tagging guidelines fro the penn treebank project. Available at `http://www.ldc.upenn.edu/`, 1995. 3rd Revision, 2nd Printing.

[9] Evelyne Tzoukermann, Dragomir R. Radev, and William A. Gale. Combining linguistic knowledge and statistical learning in french part-of-speech tagging. In *Proceedings of the ACL SIGDAT Workshop* From

Texts to Tags: Issues in Multilingual Language Analysis, pages 51–57, 1995.

[10] Atro Voutilainen. Morphological disambiguation. In Fred Karlsson, Atro Voutilainen, Juha Heikkilä, and Arto Anttila, editors, *Constraint Grammar-A Language–Independent System for Parsing Unrestricted Text*, chapter 5. Mouton de Gruyter, 1995.

[11] Atro Voutilainen. A syntax-based part-of-speech analyzer. In *Proceedings of the Seventh Conference of the European Chapter of the Association of Computational Linguistics*, Dublin, Ireland, 1995.

[12] Atro Voutilainen, Juha Heikkilä, and Arto Anttila. *Constraint Grammar of English.* University of Helsinki, 1992.

[13] Atro Voutilainen and Pasi Tapanainen. Ambiguity resolution in a reductionistic parser. In *Proceedings of EACL'93*, Utrecht, Holland, 1993.

[14] Ralph Weischedel, Marie Meteer, Richard Schwartz, Lance Ramshaw, and Jeff Palmucci. Coping with ambiguity and unknown words through probabilistic models. *Computational Linguistics*, 19(2):359–382, 1993.