ELEMENTARY TREES FOR SYNTACTIC AND STATISTICAL DISAMBIGUATION

Rodolfo Delmonte, Luminita Chiran, Ciprian Bacalu
Ca' Garzoni-Moro, San Marco 3417
Università "Ca Foscari"
30124 - VENEZIA
Tel. 39-41-2578464/52/19
E-mail: delmont@unive.it
Website: http//byron.cgm.unive.it

ABSTRACT

In this paper we argue in favour of an integration between statistically and syntactically based parsing, where syntax is intended in terms of shallow parsing with elementary trees. None of the statistically based analyses produce an accuracy level comparable to the one obtained by means of linguistic rules [1]. Of course their data are strictly referred to English, with the exception of [2, 3, 4]. As to Italian, purely statistically based approaches are inefficient basically due to great sparsity of tag distribution – 50% or less of unambiguous tags when punctuation is subtracted from the total count as reported by [5]. We shall discuss our general statistical and syntactic framework and then we shall report on an experiment with four different setups: the first two approaches are bottom-up driven, i.e. from local tag combinations:
A. Statistics only tag disambiguation; B. Stastistics plus syntactic biases; C. Syntactic-driven disambiguation with no statistics; D. Syntactic-driven disambiguation with conditional probabilities computed on syntactic constituents.
The second two approaches are top-down driven, i.e. driven from syntactic structural cues in terms of elementary trees:
In a preliminary experiment we made with automatic tagger, we obtained 99% accuracy in the training set and 98% in the test set using combined approaches: data derived from statistical tagging is well below 95% even when referred to the training set, and the same applies to syntactic tagging.

1. INTRODUCTION

We assume, together with [1] that POS tagging is essentially a syntactically-based phenomenon and that by cleverly coupling stochastic and linguistic processing one should be able to remedy some if not all of the drawbacks usually associated with the two approaches, when used in isolation. However, as will be shown in detail in the following section, rather than using FSA we use Elementary Trees organized in an RTN both for training and for parsing. As to the statistical part, we don't use HMMs but only conditional probabilities on the basis of trigram information as discussed below.
Syntactic driven disambiguation is accomplished by using an RTN made up of 1700 arcs and 22 nets, which we use in a non-recursive way, as explained below. Data for the construction of the RTN were derived from the manual annotation of 60,000 token corpus suite which is then used as test set. Frequency of occurrence associated to each rewrite rule is used as organizing criteria in the ordering of the arcs contained in each node of each net. However, in the experiment, we let conditional probabilities at the level of major constituent, or net, do the choice for the best path.

Rather than flattening the Phrase Structure Grammar as [8] suggest in their shift-reduce algorithm, we only check for reachability in nonterminal symbols. So, even though the formal structure of RTN is recursive, the disambiguating algorithm does not use recursive calls and all computation is flattened down to one level, that of tags corresponding to preterminals in the RTN. The syntactic-statistical disambiguator (hence SSD) can be defined as a slightly augmented finite state transducer which works at a single level of computation and has access to higher level information when needed. For the details of the implementations the reader should look at [10].

2 .STATISTICAL VS. SYNTACTIC DISAMBIGUATION

The SSD is the final module of our syntactic tagger of Italian. Input to the SSD is the complete and redundant output of the morphological analyser and lemmatizer, IMMORTALE [10]. IMMORTALE finds all possible and legal tags for the word/token under analysis on the basis of morphological generation from a root dictionary of Italian made up of 80,000 entries and a dictionary of invariant words - function words, polywords, names and surnames, abbreviations etc. - of over 12,000 entries.

As commented by [6], the application of stochastic techniques in automatic part-of-speech tagging is particularly appealing given the ease with which the necessary statistics can be automatically acquired and the fact that very little handcrafted knowledge need to be built into the system(ibid., 152). However both probabilistic models and Brill’s algorithm need a large tagged corpus where to derive most likely tagging information. It is a well known fact that in lack of sufficient training data, sparsity in the probabilistic matrix will cause many bigrams or trigrams to be insufficiently characterized and prone to generate wrong hypotheses. This in turn will introduce errors in the tagging prediction procedure. Italian is a language which has not yet made available to the scientific community such large corpus. In lack of such an important basic resource, there are two possibilities:

3. manually building it by yourself;

4. using some automatic learning procedure which in our case corresponds to the use of a syntactic tagger.

We have been working on such a corpus of Italian with the aim of achieving the above-mentioned final goal, without having to manually build it. The algorithm that we will present in this paper is partly based on stochastic techniques: this is however coupled with linguistic processing by means of a Context Free grammar of Italian formalized as an RTN, which filters it. Statistics is usefully integrated into the syntactic disambiguator in order to reduce recursivity and allow for better predictions.

After a first fully automatic phase, we started building BIASES which are used to correct most common errors. This second phase has taken us 3 man/months work to complete. The final result is a 95% accuracy analysis on the whole corpus. The final output has then been used to collect trigrams for the statistical tagger. Statistics and syntactic disambiguation have then been fully integrated in order to reduce recursivity and allow for better predictions and higher efficiency. Fully stochastic taggers, in case no large tagged corpora are available, may make use of HMMs. However, HMMs show some of the disadvantages present in more common Markov models: they lack perspicuity basically because they impose that the data related to tags are all treated on a par. Even though they allow for biases to be implemented – very similar to patches in Brill's tagger – they are inherently incapable of capturing higher level dependencies present in natural language, and are always prone at generating wrong interpretations, i.e. accuracy never goes higher than 96-97%. Of course it is a good statistical result, but a poor linguistic result, seen the premises, i.e. the need to use tagging information for further syntactic processing.

2.1 Tagset and Statistical Processing

Our tagset is made up of 91 tags thus subdivided: 10 for punctuation; 4 for abbreviations, titles, dates, numbers; 19 for verbs including three syntactic types of subcategorization – transitive, intransitives, copulatives – and tensed cliticized verbs; 47 for function closed class words subdivided into 18 for pronouns, clitics, determiners and quantifiers - 18 for adverbs conjunctions and prepositions - 11 for auxiliaries and modals; 11 for adjectives and nouns, including special labels for colour nouns, time nouns, factive nouns, proper nouns, person names - this list includes special labels for guessed proper nouns, foreign words and misspelled words. Twenty categories from the general tagset never occur single, so they had to be converted into distributionally equivalent ones, in the statistical table.

We refer to the tagset of LOB corpus which uses 157 tags for English: however, they include in their set special tags for plural forms, genitive forms both for nouns and verbs, and with tags for comparative and superlative forms of adjectives. In case we eliminate these duplicate forms the total number of tags is 107.

The disambiguator is made up of two separate modules: the Probabilistic Transition Table for local tag disambiguation; the syntactic transition network where the learning phase is situated. We use a Viterbi-like algorithm to find and select the best candidates in any given context, given the trigram matrix information. However, since we only computed trigram for a comparable small quantity of training data - we would need 700K trigrams for our 90 tags, but we only use 30K! - we often find no data available. In a similar way to the reductionist statistical approach proposed by [2,7] we induce the best tag from the set of available tags in the context of an unambiguous tag by recursively calling all contextually allowable combinations, from where we select the ones corresponding to the current ambiguity class: we then compute trigram conditional probabilities, according to the formula suggested in [2]. We remove low-probability candidate tags by ignoring the tail of the Viterbi output list, on the basis of a fixed threshold. In case no data are available, rather than computing zero probability we let the current procedure fail - the algorithm is implemented in Prolog - and use information coming from Elementary Trees (ETs) or Networks which can be superimposed on each tag in a given context: the most adequate ETs will be chosen in the top-down syntactically driven disambiguating procedure.

The final aim of the disambiguation is to produce information reusable by the following shallow parser, which will then be in charge of combining ETs previously assigned by the SSD.

3. SYNTACTIC CONSTITUENCY ANNOTATION

The first problem to be solved when starting work on a corpus in order to produce a syntactic structure annotation, is the choice of representation, or the syntactic annotation scheme. As with tagging, the scheme must be consistent, it could be used as gold standard for parser testing or as a basis for the induction of stochastic grammars and lexical representations. The main sources of information in the field of syntactic annotation scheme are related to the Penn Treebank (hence PT) [11], which is remarkable as to extension of the coverage and documentation of linguistic phenomena. The PT uses a generativist constituency which is related to chomskian syntax of the '60s/'70s which we do not share: as a result, much of the bracketing is non comparable. In addition, syntactic constituency has been enriched with functional labels and other non standard additional labels which increased the overall number of constituents but reduced its perspicuity. As a result, PT uses 22 symbols for main constituent and 32 more for functional annotation. We also use 22 symbols for syntactic constituency but they are different from the PT's ones.

The inventory we use follows the basic intuitions of the XBAR syntax, while having as its main goal that to serve as an interface as simple as possible to the following levels of representations: the functional, LFG-style, and the semantic ones. In particular, whereas PT uses Chomsky-adjunction and VP, we opted for a separated IBAR constituent with all tensed verbal constituents and its adjoined minor constituents, like negation, clitics and certain adverbials. We then qualify all verbal complements according to their lexical subcategorization frame. Seen that they only have one layer of syntactic representation, whereas we allow for two, they include all semantic information at constituent level. In particular, they introduce all possible empty categories in the syntactic constituents with coindexation. In case of discontinuous or non canonical order of constituents, they use special constituent names, like SINV (Inverted Sentence), to allow for the subject NP to be automatically recovered. We introduce no empty category at syntactic level, while leaving their computation for the functional and semantic level. . As an example we report the bracketing for "John's decision to leave":

```
(NP (NP John 's)
        decision
                (S (NP-SBJ *)
                        (VP to
                        (VP leave))))
```

compared to the Italian, "la decisione di Gino di partire"

```
SN-[la-art, decisione-n,
                SPD-[di-pd, SN-[Gino-nh] ]
                SV2-[di-pt, partire-viin] ]
```

where we can see that the level of embedding in PT is 4 brackets, whereas it is 2 brackets in our representation. We report here below the list of constituents in our representation for Italian corpora.

TABLE 1. List of Syntactic Constituents and their meaning

F	**sentence, starting with subject SN or SV2; or in case subject is missing starting with IBAR**
SN	**noun phrase, including its complements and/or adjuncts**
SA	**adjectival phrase, including its complements and/or adjuncts**
SP	**prepositional phrase**
SPD	**prepositional phrase DI / "of"**
SPDA	**prepositional phrase DA / "by,from"**
SAVV	**adverbial phrase, including its complements and/or adjuncts**
IBAR	**verbal nucleus with finite tense and all adjoined elements like clitics, adverbs and negation**
SV2	**F for infinitival clause**
SV3	**F for participial clause**
SV5	**F for gerundive clause**
FAC	**CP for sentential complement**
FC	**CP for Coordinate sentences (also ellipsed and gapped)**
FS	**CP for Subordinate sentence**
FINT	**CP for +wh interrogative sentence**
FP	**CP for punctuation marked parenthetical or appositional sentence**
F2	**CP for relative clause**
CP	**Generically for dislocated or fronted, sentential adjuncts**
COORD	**Coordination with coordinating conjunction as head**
COMPT	**Transitive/Passive/Ergative/Reflexive Complement**
COMPIN	**Intransitive/Unaccusative Complement**
COMPC	**Copulative/Predicative Complement**

4. AUTOMATIC SYNTACTIC TAGGING

Being language-dependent the tagger needs to be based on an accurate analysis of corpora with an as broad as possible coverage of genre, style and other social and communicative variables. To answer these needs we built our syntactic shallow parser on the basis of manually annotated texts for 60,000 words chosen from different corpora and satisfying the above-mentioned criteria. The annotation was carried out twelve years ago to be used for a text-to-speech system for Italian (DecTalk Italian version) with unlimited vocabulary.

We report here below the list of the 10 main constituents or net labels used by the annotators, which are a superset of our current syntactic tagset which is subsumed by it. As can be easily seen, lexical subcategorization information for verbs was not included: also, no information was available as to DI/DA (of/by-from) PPs, nor a subdivision of sentences in simplex and complex with subordination. Sequences of preterminal symbols, category labels or simply POS tags may reasonably belong to three levels of constituency: in the most desirable case, they may be part of the same constituent, e.g. NP(art, quant, noun); else, they may belong to a parent node, whose head is followed by the Complement node, any head dependent constituent in a daughter node, e.g. NP(art, noun (AP(adj)); finally, it may belong to two sibling nodes from a common higher parent node, as for instance in the case of CP(AdvP(adv, NP), IP(NP, VP)).

However, our tagset of elementary trees is different from the one used within the LTAG approach [12], where they are called Supertags: in our framework, elementary trees only belong to the syntactic constituency domain. On the contrary, in the LTAG framework they are constituted by both syntactic and functional constituent labels.

Table 2. Net Accessibility Preterminals and their Frequency

NET	TAG	FREQ	NET	TAG	FREQ	NET	TAG	FREQ	NET	TAG	FREQ
F	PK	235	SN	Q	189	SP	P	6160	SV	VG	147
F	CONG	218	SN	PRON	338	SV	V	656	SV	VPP	814
F	COSU	294	SN	ART	3792	SV	AUSA	244	SV	VSUP	518
SA	A	353	SN	DIM	117	SV	AUSE	363	SV2	P	173
SA	Q	239	SN	N	1662	SV	CLIT	388	SV2	PT	529
SAVV	AVV	1479	SP	PART	5234	SV	NEG	318	SV2	VI	217

Disambiguation proceeds as follows. Fully ambiguous cases such as the following:**Tag1=[ag, n], Tag2=[ag, n],** cannot be solved by relying on frequency of occurrence given the fact that 75% of all NP rules take the pair Noun/Adjective, and only 25% take Adjective/Noun.
We use biases which take into account a list of exceptions - ambiguous cases which prefer Prenominal position and only then to use local cues provided by the RTN.
At first we try to traverse the network by continuing in the network accessible from the left highest score tag, as explained below. Net traversal is worked out trying to proceed from the arc associated with that tag onto a following one as encoded in the RTN and extracted from the current tag-list. The arc in question is called from the pair (Net, Tag). The output is the associated arc, which is represented as follows,

- arc(Net,Category,InputNode,OutputNode).

In case the current tag(list) is accepted by the RTN no further computation is needed: the associated network will be used for further processing.

2. In case of failure, we execute in turn the following procedures:

a. The two tags belong to two separate networks which are in an inclusion relation;

b. The two tags belong to non inclusive networks.

Case a. is further expanded as follows:

Tag 1 belongs to a network which includes the network to which Tag 2 belongs. Network for Tag 2 is then simply asserted as the first network that Tag 2 may be a proper starting category for.

This information is recovered from a Network Accessibility Table Lookup (NATL) as indicated in Table 2, where all category symbols are cross-tabulated against the network they may provide access for. NATLs are compiled at runtime and are encoded as sets of starting symbols for each network with a given probability.

Match for tags is a simple membership check.

```
Tag1/Tag2 => Network1/Tag1 ⊇
                    Network2/Tag2
```

Tag 1 and Tag 2 belong to two separate networks which are both included in another network. Whereas in i. above it was between terminal and nonterminal, this time, the inclusive relationship is between nonterminals,. Network for Tag1 and network for Tag2 are both included in the set of Networks accessible from a higher Network. NATLs used in this case are for nonterminals.

```
        Tag1/Tag2=>
{Network1/Tag1,Network2/Tag2} ⊇
        HigherNetwork
```

Tag1 and Tag2 cannot be regarded a legal continuation as can be computed from the available grammar encoded in the RTN. The parsing process is reverted from Top-down to Bottom-up. The first network associated to Tag2 as recovered from NATLs.

5.THE EXPERIMENT

As said above, we took two subparts in order to check the effect of training separately. The benchmark test corpus was constituted by a segment from the School-Administration corpus which amounts to approximately 10,000 tokens and is not included in the training set.

We constrained the choice of the statistical tagger by the matrix of actually occurring combinations as determined by the syntactic disambiguator. Thus the training set should have granted similar results, but as Table 4. clearly show, this is not the case. Some improvements are obtained by the addition of Biases, which in one case that advantage of local syntactic accessibility information.

Table 3. EXPERIMENTAL RESULTS

	Tot. Toks	%Tot	Syntax Only	Syntax + Biases	Trigrams only	Trigr. + Biases	Syntax + Statistics
Culture	28,000	62%	90%	97.5%	93.5%	96.5%	98.97
Politics	7,000	16%	87%	95.5%	92.5%	94.5%	98.05
S.Admin.	10,000	22%	86,5%	93.5%	90.5%	93.5%	97.85
Total	45,000	100%					

6. REFERENCES

[1] P. Tapanainen and Voutilainen A.(1994), Tagging accurately - don't guess if you know, ***Proc. of ANLP '94***, pp.47-52, Stuttgart, Germany.

[2] Brants T. & C.Samuelsson(1995), Tagging the Teleman Corpus, in ***Proc.10th Nordic Conference of Computational Linguistics***, Helsinki, 1-12.

[3] Lecomte J.(1998), Le Categoriseur Brill14-JL5 / WinBrill-0.3, INaLF/CNRS,

[4] Chanod J.P., P.Tapanainen (1995), Tagging French - comparing a statistical and a constraint-based method". ***Proc. EACL'95***, pp.149-156.

[5] Delmonte R., E.Pianta(1999), Tag Disambiguation in Italian, ***in Proc.Treebanks Workshop ATALA***, Paris, pp.18-25.

[6] Brill E. (1992), A Simple Rule-Based Part of Speech Tagger, in ***Proc. 3rd Conf. ANLP***, Trento, 152-155.

[7] Voutilainen A. and P. Tapanainen,(1993), Ambiguity resolution in a reductionistic parser, ***in Sixth Conference of the European Chapter of the ACL***, pp. 394-403. Utrecht.

[8] Pereira F., R.Wright, (1991), Finite-State Approximation of Phrase Structure Grammars, in ***Proc. 29th ACL***, Berkeley, 246-255.

[10] Delmonte R., E.Pianta(1996), "IMMORTALE - Analizzatore Morfologico, Tagger e Lemmatizzatore per l'Italiano", in ***Atti V Convegno AI*IA***, Napoli, 19-22.

[11] Marcus M. et al.(1993), Building a Large Annotated Corpus of English: The Penn Treebank, ***Computational Linguistics***, Vol.19.

[12] Bangalore S. & A.K.Joshi(1998), Supertagging: An Approach to Almost Parsing, in Computational Linguistics, 22.